Universidad de las Ciencias Informáticas

Facultad 3.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título:

Desarrollo del módulo de interoperabilidad del Sistema de Gestión de Archivos Arkheia.

Autora: Ayle Morales Basulto.

Tutores: Ing. Yanet del Risco Batista.
Ing. Yaidel Ferrales Obregón.

Ciudad de la Habana, Junio de 2013

Año 55 de la Revolución

Los grandes conocimientos engendran las grandes dudas.

Declaración de Autoría

Declaro ser la autora de la presente tesis, reconociendo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de manera exclusiva.

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

Ayle Morales Basulto

_____

Firma del Autor

Yanet del Risco Batista

_____

Firma del Tutor

Yaidel Ferrales Obregón

_____

Firma del Tutor

Agradecimientos

A mis padres por su amistad, apoyo incondicional y por ser un ejemplo a seguir en todas las etapas de mi vida.

A mi hermano por ser mi buen amigo.

A mi novio por ser mi guía y la luz que me iluminó durante toda la carrera, por su paciencia, por compartir conmigo momentos difíciles y de alegría, por ser mi confesor, amigo y amante. Te amo.
A toda mi familia, por la preocupación y apoyos brindados en todo momento.

A todas las personas que compartieron conmigo desde primer año, sepan que me llevo un grato recuerdo de todos, en especial de Liliana, Danay, Katy, Jiubel, Pompi, Carlos y Coqui.

A los que no estuvieron conmigo desde primer año, pero fueron como una familia más, al grupo 3308 en especial Gisel, Analie, Joceline, Yanet y Yanerkis.

A mi tutora Yanet por no considerarme como su tesista sino como su hija adoptiva, muchas gracias por darme la oportunidad de ser tu tesista y por toda la ayuda brindada durante el desarrollo de este trabajo.

A todas aquellas personas que no formaron parte de mi grupo docente pero sí de mis buenos amigos: Leandro, Rene, Yasel, Zulay, Apa, Dariel.

A todas las personas maravillosas que he conocido en estos cinco años y que me han dado fuerzas para seguir adelante…
Gracias

Dedicatoria

A mis padres, mi hermano y mi novio por ser mis fuentes de inspiración durante estos cinco años.

Resumen

Los archivos históricos son la memoria colectiva de una nación, región o localidad; testimonios que evidencian la experiencia humana. Una forma de facilitar el manejo de la información es a través de la utilización de los Sistemas de Gestión de Documentos de Archivos (SGDA). Un elemento esencial en el contexto de estos sistemas 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. La interoperabilidad entre SGDA evita la duplicidad de los datos y permite contar con la información y el intercambio de la misma en el momento oportuno. El producto Xabal-Arkheia desarrollado por la UCI para la gestión de documentos en los archivos históricos no cuenta con los mecanismos que permitan intercambiar información entre los sistemas de archivos. Esto provoca que los investigadores tengan que trasladarse hasta las instituciones para saber si existe un determinado documento que ellos necesiten. Además estos no pueden realizar búsquedas de información en diferentes instituciones desde una misma ubicación, lo que se traduce en un servicio incompleto al cliente y un mal aprovechamiento de los recursos disponibles.

Con el presente trabajo se obtuvo el módulo Interoperabilidad, el cual garantiza el intercambio de información entre el sistema Xabal-Arkheia y otros SGDA mediante el protocolo OAI-PMH evitando la duplicidad de información en los Archivos, garantizando un mayor aprovechamiento de los recursos y un mejor servicio al cliente.

Palabras claves: duplicidad, intercambio, interoperabilidad, SGDA.

Índice

INTRODUCCIÓN

Toda sociedad necesita conservar sus documentos como argumento eficaz de las actividades desarrolladas, pues sin ellos se podría considerar como una sociedad sin historia, sin experiencia y la misma sería desconocida por futuras generaciones. Al lugar destinado al almacenamiento de estos documentos se denomina archivo.

Los archivos remontan sus orígenes al surgimiento de la escritura. Diferentes culturas como la griega, la romana y la egipcia contaron con importantes archivos. Estos como institución fueron creados para servir de depósito de colecciones de documentos que poseían valor administrativo, legal, fiscal, científico, económico, político, cultural y/o histórico, para de esta forma facilitar su tratamiento, servicio y control.

La finalidad de los archivos es gestionar, atesorar, conservar y difundir el patrimonio documental. Pueden almacenar documentos históricos recibidos por donación, depósito, transferencia y adquisición (1). Una forma de facilitar el manejo de la información es a través de la utilización de los Sistemas de Gestión de Documentos de Archivos (SGDA).

Un elemento esencial en el contexto de los SGDA 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. Los sistemas de información interoperables deben implementar determinada norma o estándar que permita dicha comunicación de manera rápida y transparente (2).

La interoperabilidad entre SGDA evita la duplicidad de los datos y permite contar con la información y el intercambio de la misma en el momento oportuno. Con todos los sistemas interconectados, no existe la necesidad de copiar la información cuando se puede acceder desde cualquier lugar al sistema que contenga los datos pertinentes. Varios SGDA conectados que utilicen un mismo protocolo de comunicación representa una red de sistemas.

En la Universidad de las Ciencias Informáticas, el Centro de Informatización de la Seguridad Ciudadana (ISEC), cuenta con un departamento para el desarrollo de SGDA que tiene como su principal producto el sistema Xabal-Arkheia. Este sistema cuenta con varias funcionalidades como la búsqueda de documentos, empleando los metadatos

descritos a través del sistema siguiendo la Norma Internacional General de Descripción Archivística ISAD (G).

El sistema Xabal-Arkheia no cuenta con los mecanismos que permiten intercambiar información entre los archivos en los que se encuentra instalado. Actualmente los sistemas que gestionan documentos de archivo líderes en el mercado brindan soporte a estas funcionalidades. Entre las ventajas que se pueden mencionar del intercambio de información entre documentos de archivo se encuentran:

- Contribuye a la difusión de la información custodiada en las instituciones de archivo.
- Ayuda a los investigadores en localización de los documentos, evitando que tengan que trasladarse hasta las instituciones para conocer si el documento forma parte del fondo documental en esta.

Por lo anteriormente planteado surge como **problema a resolver**:

¿Cómo garantizar que el Sistema de Gestión de Documentos de Archivo Xabal-Arkheia posibilite el intercambio de la información de los documentos descritos en las instituciones que lo utilizan?

Definiendo como **objeto de estudio:**

Los protocolos de comunicación entre aplicaciones web.

Tomando en cuenta lo anterior se plantea el **objetivo general**:

Desarrollar un módulo que posibilite el intercambio de la información de los documentos descritos en las instituciones que utilizan el sistema Xabal-Arkheia**.**

Quedando definido como **campo de acción**:

Protocolos de comunicación para aplicaciones de gestión de documentos de archivo.

La **Idea a defender** es:

El desarrollo del módulo interoperabilidad para el sistema Xabal-Arkheia garantizará el intercambio de información de los documentos descritos en las instituciones que lo utilizan.

Para dar cumplimiento al objetivo general trazado se han definido los siguientes **objetivos específicos:**

- ✓ Realizar el estudio de las aplicaciones de gestión de archivo y los protocolos de comunicación que utilizan para lograr la interoperabilidad.
- ✓ Realizar el análisis y diseño del módulo de interoperabilidad del Xabal-Arkheia.
- ✓ Implementar el módulo de interoperabilidad del Xabal-Arkheia.
- ✓ Validar el funcionamiento del módulo de interoperabilidad del sistema Xabal-Arkheia.

En aras de dar cumplimiento al objetivo planteado se traza el siguiente conjunto de **tareas de investigación**:

- ✓ Evaluación de los protocolos que permitan un intercambio de información entre sistemas de archivos.
- ✓ Elaborar estado de arte de los sistemas para la gestión de documentos de archivos existentes en el mercado.
- ✓ Análisis de la arquitectura y los patrones a utilizar en el desarrollo de la solución.
- ✓ Descripción de las principales funcionalidades del módulo a implementar.
- ✓ Elaboración de los diagramas de clases, componentes y despliegue del módulo.
- ✓ Validación del diseño propuesto.
- ✓ Realizar pruebas funcionales del módulo.
- ✓ Solución de las no conformidades detectadas durante la aplicación de la estrategia de pruebas.

Para el desarrollo de esta investigación se utilizaron diferentes métodos de investigación, los cuales se detallan a continuación.

- ✓ **Analítico – sintético**: este método fue utilizado en el proceso de análisis de la bibliografía utilizada, realizando la extracción de los elementos más importantes relacionados con los metadatos y con los protocolos de intercambio de información entre sistemas, para facilitar el resultado de la investigación.
- ✓ **Análisis histórico – lógico**: este método fue utilizado en el análisis la trayectoria de los protocolos de intercambio de datos entre sistemas de información, su condicionamiento a los diferentes periodos de la historia y las conexiones históricas fundamentales.

El documento consta de 3 capítulos además de los Anexos.

En el capítulo 1 se abordan los conceptos fundamentales a tener en cuenta para comprender los términos usados en el desarrollo del trabajo. Se realiza un estudio de los principales protocolos para lograr la interoperabilidad entre SGDA. Además, se exponen las distintas tecnologías a utilizar en el desarrollo del módulo propuesto.

En el capítulo 2 se muestra el análisis y diseño del módulo presentando el modelo de dominio, en el cual se capturan los conceptos más importantes en el contexto del sistema. Se realiza un levantamiento de los requerimientos funcionales a partir de los cuales se definen los casos de uso del sistema. Como parte del proceso de diseño se confecciona un conjunto de artefactos para un mayor entendimiento del mismo. Se explica la arquitectura que será empleada.

En el capítulo 3 se detallan las actividades definidas para la implementación y las pruebas del módulo de Interoperabilidad del sistema Xabal-Arkheia. Se especificaran los diagramas de componentes y de despliegue, así como las pruebas que se le realizaran al módulo, plasmando los resultados obtenidos en las mismas.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En este capítulo se presentan los conceptos fundamentales que constituyen la base de la investigación. Se realiza un estudio de los principales protocolos para lograr la interoperabilidad entre SGDA, seleccionando uno de ellos para darle solución al problema planteado. Además, se realiza una breve descripción de las distintas herramientas y tecnologías que serán utilizadas en el desarrollo del módulo de interoperabilidad del SGDA Xabal-Arkheia.

1.2 Conceptos fundamentales

La **archivística** es la ciencia de los archivos, está integrada por un conjunto de conocimientos y de métodos para el tratamiento de los documentos y de los archivos (3).

Según el diccionario de terminología archivística, la archivística es la disciplina que trata de los aspectos teóricos y prácticos de los archivos y el tratamiento archivístico de sus fondos documentales (4).

De manera general la archivística es la ciencia que se encarga de estudiar la naturaleza de los archivos.

Según el Consejo Internacional de Archivos (ICA/CIA), la palabra **"archivo"** tiene tres acepciones (5):

- Conjunto de documentos sean cuales sean su fecha, su forma y su soporte material, producidos o recibidos por toda persona física o moral y por todo servicio u organismo público o privado, en el ejercicio de su actividad.
- Institución responsable de la acogida, tratamiento, inventariado, conservación y servicio de los documentos.
- Edificio o parte de edificio donde los documentos son conservados y servidos.

Entre las definiciones más completas del término "archivo", según la archivera sevillana Antonia Heredia Herrera se encuentra la siguiente:

Archivo es 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 lo produce, para los ciudadanos o para servir de fuentes de historia (5).

Cruz Mundet[1], plantea 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 (3).

De manera general un archivo es un espacio donde se almacenan documentos sin importar fecha, naturaleza y soporte material que puede servir de ayuda a la persona interesada en la búsqueda de información.

Un **documento de archivo** es el testimonio material de un hecho o acto realizado en el ejercicio de sus funciones por personas físicas y jurídicas, públicas o privadas, de acuerdo con unas características de tipo material o formal (6).

La **gestión documental** es un área de la administración general que se encarga de garantizar la economía y la eficiencia en la creación, mantenimiento, uso, y disposición de los documentos administrativos durante todo su ciclo de vida. Esta se extiende al complejo ciclo de vida del documento, desde su producción hasta la eliminación final y su envío al archivo para su conservación permanente. Está dirigida a asegurar una documentación adecuada, evitar lo no esencial, mejorar la forma de cómo se organizan y recuperan los documentos (7).

La **interoperabilidad** es la capacidad para comunicar sistemas entre ellos y pasar información de ida y vuelta en un formato utilizable (8).

La **descripción archivística** es el medio utilizado por el archivero para obtener la información contenida en los documentos y ofrecerla a los interesados en ella. La descripción persigue dos objetivos: dar información a los demás y facilitar el control al archivero. El objeto de la labor descriptiva es el hacer accesibles los fondos documentales. La descripción de los documentos constituye la parte culminante del

[1] Cruz Mundet: Presidente de la asociación latinoamericana de archiveros.

trabajo archivístico y viene a coincidir exactamente en su finalidad con la de la propia documentación: informar (9).

En los últimos años un instrumento descriptivo que está ganando mucha aceptación es la norma internacional general de descripción archivística ISAD (G), la cual es una normativa internacional para la descripción de documentos.

La norma ISAD (G) se rige por las siguientes reglas (10):

- Descripción de lo general a lo específico.
- Información pertinente al nivel de descripción
- Interconexión de las descripciones
- No repetición de la información

Las reglas de la ISAD (G) se encuentran estructuradas en siete áreas de información (10):

- Área de identificación: Esta área recoge información esencial para identificar a la unidad de descripción.
- Área de contexto: Su objetivo es almacenar información acerca del origen y custodia de la unidad de descripción.
- Área de contenido y estructura: Almacena información sobre el tema principal de los documentos y la organización de la unidad de descripción.
- Área de condiciones de acceso y uso: Acumula información acerca de la disponibilidad de la unidad de descripción.
- Área de documentación asociada: Recopila información acerca de los materiales que tengan una relación importante con la unidad de descripción.
- Área de notas: Recoge información especializada que no se puede acomodar en ninguna de las otras áreas.
- Área de control de descripción: Almacena información sobre cómo, cuándo y por quién se ha preparado la descripción archivística.

Esta norma está constituida por 26 elementos que permiten la descripción archivística, de los cuales sólo 6 se consideran de carácter obligatorio para el intercambio internacional de la información descriptiva como son (10):

- Código de referencia.

- Título.
- Productor(es).
- Fecha(s).
- Extensión de la unidad de descripción.
- Nivel de descripción.

1.3 Protocolos de comunicación para aplicaciones de gestión de archivos

1.3.1 Protocolo

Un protocolo es un conjunto de normas que definen la comunicación entre sistemas. FTP (Protocolo de Transferencia de Ficheros) y HTTP (Protocolo de Transferencia de Hipertexto) son ejemplos de otros protocolos utilizados para la comunicación entre sistemas a través de Internet (11).

Para que la transferencia e intercambio de datos entre los SGDA ocurra de una manera transparente al usuario se utilizan uno o varios protocolos dentro de los cuales se pueden encontrar:

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 para la participación de los departamentos (12):

- Nivel de archivo de carácter pasivo, pues simplemente proporciona información.
- Nivel de servicio activo ya que extrae la información de los anteriores y construye un “servicio” de utilidad para los usuarios finales.

Su principal objetivo es facilitar el acceso a los resultados de investigaciones entre comunidades RePEc (Research Papers in Economics / Trabajos de investigación económicos), adaptarlo y configurarlo al sistema Xabal-Arkheia implicaría un esfuerzo adicional por lo que no será tenido en cuenta en la solución propuesta.

Protocolo Dienst: Permite la unión de un conjunto de servicios para formar una biblioteca digital. Los servicios y recursos de una aplicación que utiliza este protocolo pueden estar físicamente ubicados en cualquier lugar. Es utilizado por las bibliotecas digitales para poner sus documentos a disposición de los usuarios. Parte de cuatro tareas básicas:

búsqueda y recuperación de documentos, navegación, agregación de nuevos documentos y registro de usuarios. Propone seis categorías de servicios: repositorio, indexado, mediadores, información, colección y registro (13).

Con el paso del tiempo este protocolo ha sido sustituido casi en su totalidad por otros como el OAI-PMH, por lo cual no será tomado en cuenta para la solución del problema en cuestión.

(Protocolo Simple para la Interoperabilidad de Bibliotecas Digitales) SLIPD: Permite integrar fuentes de información heterogéneas. Fue desarrollado en conjunto con las universidades de Stanford, Berkeley, Santa Bárbara, el Centro de Supercomputadoras de San Diego y el Proyecto de Librería Digital de California. Se pueden construir implementaciones sobre transporte HTTP. (14).

Este protocolo permite comunicar con otros sistemas, pero no permite que otros sistemas busquen archivos en el sistema Xabal-Arkheia lo cual es uno de los fundamentos y base del sistema a implementar en el transcurso de la presente investigación. Por lo anteriormente planteado este protocolo no formará parte de la solución.

(Open Archives Initiative - Protocol Metadata Harvesting o Iniciativa Abierta de Archivos – Protocolo de Recolección de Metadatos) OAI-PMH: Herramienta de interoperabilidad que posibilita el intercambio de metadatos sobre cualquier material almacenado en soporte electrónico. Considerando que los metadatos a transmitir vía OAI-PMH pueden codificarse usando cualquier norma existente, además de la norma Dublin Core[2]. Esta transferencia puede realizarse desde diferentes proveedores de servicios a través de búsquedas que abarquen la información recopilada en distintos repositorios de archivos asociados (proveedores de datos).

OAI-PMH proporciona una plataforma sencilla y basada en la tecnología existente para acceder y/o difundir cualquier tipo de información. 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

[2] Dublin Core: Modelo de datos cuyas implementaciones usan XML.

describen un determinado recurso y no el recurso en sí, permitiendo que la transmisión se realice de una manera rápida (15).

El proyecto Archivo utiliza la norma ISAD (G) que puede ser adaptada a este protocolo, el mismo una vez implementado permite a cada SGDA actuar como proveedor y cliente de servicio. Mediante uno de los verbos del protocolo es posible obtener los formatos que implementa el proveedor de datos, así como realizarle peticiones en cualquiera de ellos.

Por las razones anteriormente expuestas se decide emplear OAI-PMH para poner a disposición de otros sistemas de información el patrimonio documental descrito en Xabal-Arkheia.

Flujo actual de los procesos del OAI-PMH

En la siguiente figura se puede apreciar el funcionamiento básico del protocolo OAI-PMH.

Figura 1: Funcionamiento básico del protocolo OAI-PMH.

Las peticiones realizadas en el protocolo OAI-PMH utilizan transacciones HTTP. El protocolo OAI-PMH basa su funcionamiento en dos servidores: Proveedores de Servicios (PS) y Proveedores de Datos (PD). El PS realiza una petición al PD con el objetivo de recolectar sus metadatos. En respuesta, el PD devuelve un conjunto de registros en formato XML, incluyendo identificadores de los objetos descritos en cada registro.

Este mecanismo de recolección de metadatos de los documentos de archivos preservados dentro de un repositorio o sistema de información se realiza mediante solicitudes al protocolo OAI-PMH. Estas solicitudes se realizan por medio de los métodos GET y POST del protocolo HTTP. Todas las solicitudes realizadas a un sistema de

información que haya implementado el protocolo OAI-PMH se hacen a través de una dirección de internet o URL base que es la que apunta directamente al protocolo implementado.

Dentro de este protocolo, las solicitudes o peticiones realizadas entre el PS y el PD se han definido como un conjunto de 6 verbos, mediante los cuales un PS podrá recuperar la información que contiene el PD. Las peticiones contenidas en el protocolo son (16):

- **Identify**: Esta petición se usa para que el PD conozca la información tanto técnica como administrativa del PS, con el fin de establecer cómo se va a realizar el proceso de cosechado de metadatos. La información enviada del PD al PS debe incluir una instancia de los siguientes elementos:
  - ✓ Nombre del repositorio.
  - ✓ Base Url.
  - ✓ Versión del Protocolo OAI-PMH por el cual se va a realizar la recolección de metadatos.
  - ✓ earliestDatestamp, donde se expresa la fecha del último cambio de adición, eliminación o modificación de los objetos dentro del repositorio.
  - ✓ deletedRecord, establece la forma como se realiza el proceso de eliminación de los objetos dentro del repositorio, que puede ser de 3 tipos: No se realiza, transitoria o persiste.
  - ✓ Granularidad, que define el formato de espacio y tiempo bajo el cual está soportado el repositorio, tiene que estar definida en la norma ISO-860131
  - ✓ Correo electrónico del administrador del repositorio.
- **ListRecords:** Este es el verbo que se utiliza para realizar la recolección de una agrupación de objetos hospedados en un repositorio, a fin de dar un resultado al PS coherente con las necesidades del mismo. Este verbo tiene que llevar como argumento el metadataPrefix, que es el que utiliza el protocolo, para enviar la información solicitada en un estándar de metadatos específico.

- **ListMetadataFormat:** Este verbo se usa para saber bajo qué tipos de metadatos el PS puede recuperar la información dada por el PD, por lo general este protocolo se basa en los metadatos Dublin Core. Con los resultados de esta petición, el PS

seleccionará la sintaxis de metadatos que utilizará para recuperar la información del PD.

- **GetRecord**: Este verbo se utiliza para recolectar un solo metadato asociado a un objeto dentro del repositorio, como obligatorio. Para poder recolectar el metadato asociado a un objeto mediante este verbo, se debe realizar la petición al PD con los siguientes argumentos
  - ✓ Identifier, es la llave o número único dentro del repositorio que identifica al objeto dentro del sistema.
  - ✓ MetadataPrefix, es la sintaxis de metadatos que se va a utilizar para realizar el cosechado del objeto.
- **ListIdentifiers:** Este verbo funciona de igual forma que el ListRecords pero con el mismo solo se obtienen las cabeceras de los registros que se necesitan recolectar. Los argumentos requeridos por este verbo son los mismos que se solicitan cuando se hace una petición OAI-PMH ListRecords.
- **ListSets:** Este verbo se usa para conocer cuáles son las agrupaciones de objetos que se tienen dentro de un repositorio documental, una agrupación puede ser los tipos de documento (tesis, libros, música, videos, monografías, etc.) o los estado del documento (publicados, en borrador, etc.).

1.3.2 Proveedores de datos y de servicios

La arquitectura de OAI-PMH se basa en cliente/servidor. Los PD son los archivos que proporcionan la información y los PS son los recolectores o servicios que toman los datos, con el objetivo de incorporarles algún valor añadido y presentarlos a los usuarios finales (17).

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.

Estos PD son los encargados de recibir peticiones de metadatos provenientes de PS, validarlas y decidir si son correctas o no. Al recibir una petición, si no es correcta, el PD debe ser capaz de responder con un error y en caso contrario debe realizar los procesamientos para ofrecer una respuesta apropiada a quien realizó la petición (18).

Los PS recolectan los metadatos de los PD y emplean los metadatos recolectados con el fin de proporcionar servicios (19). El PS es el responsable de recopilar los datos de los PD para proporcionar servicios de valor añadido en función de los metadatos que ha recogido. Entre el tipo de ayuda que el PS pueda proporcionar está la creación de una interfaz de búsqueda unificada en todos los repositorios.
OAI-PMH soporta cualquier formato de metadatos codificado en XML, siendo Dublin Core de obligada implementación. 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 (20).

1.3.3 Estándares de metadatos

Los metadatos pueden ser definidos como datos sobre otros datos. Es el término usado en la era de internet para la información que los bibliotecarios tradicionalmente habían puesto en los catálogos, y más comúnmente se refiere a información descriptiva sobre recursos de la Web. Un registro de metadatos consiste en un conjunto de atributos, o elementos necesarios para describir la fuente en cuestión (21).

Los metadatos permiten incrementar el acceso a los objetos si los mismos están descritos correctamente, disminuye el tráfico en la red, pues al clasificar la representación del objeto y no el objeto en sí, no se requiere demasiado ancho de banda para hacer las búsquedas o generar los índices. Estos expanden el uso de la información facilitando la difusión de versiones digitales de un único objeto. Permiten establecer claramente las restricciones de uso, condiciones de licenciamiento, informan sobre los derechos de autor, el control del todo o de una parte del objeto, el método de pago, si es comercial y el control al acceso a información restringida, entre otros (22).

Los metadatos en el protocolo OAI-PMH se emiten en formato XML. Para describir los documentos de archivos del sistema Xabal-Arkheia es utilizada la norma de descripción archivística ISAD (G), esta norma no presenta un formato XML por lo que en la solución del problema en cuestión se decidió utilizar la norma Dublin Core pues el protocolo define que esta norma es obligatoria. Adicionalmente se implementará la norma EAD (Encoded Archival Description / Descripción Archivística Codificada) pues contiene una descripción de metadatos más ampliada que la Dublin Core.

La 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, con el objetivo 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. Es una estructura de datos normalizada que reproduce en formato digital los instrumentos de descripción archivística (23).

La EAD 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 (SGML) y del Lenguaje de Marcas Extensible (XML) para codificar instrumentos de descripción (7).

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 mismo además de la información contextual y administrativa asociada, <findaid> (23).

Para utilizar la norma EAD se hace necesario realizar una transformación de los elementos de ISAD (G) a EAD. En la tabla siguiente se puede consultar dicha transformación.

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 1: Transformación de ISAD (G) a EAD

Dublin Core es una norma de descripción archivística donde los metadatos que provee se encuentran en formato XML. La semántica de esta norma ha sido establecida por un grupo internacional e interdisciplinario de profesionales de la biblioteconomía, la informática, la codificación textual, la comunidad museística, y otros campos teórico-prácticos relacionados (24).

El estándar de metadatos Dublin Core es un conjunto de elementos para describir una amplia gama de recursos de red, cada elemento es opcional y puede repetirse. La norma del Dublin Core conlleva dos niveles: Simple y cualificado. El simple presenta quince elementos (creador, título, fecha, etc.); el cualificado presenta otros elementos adicionales, como la audiencia, procedencia, titulares de los derechos, etc. Además de un grupo de elementos que refinan la semántica de los elementos de tal forma que pueden ser útiles para la recuperación/localización de recursos en internet (24).

Es necesario transformar los elementos de ISAD (G) a Dublin Core para que los metadatos contenidos en Xabal-Arkheia puedan ser enviados a los proveedores de servicios en formatos Dublin Core. Esta transformación puede ser visible en la siguiente tabla.

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 2: Transformación de ISAD (G) a Dublin Core.

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

Archivo 3000: Surge en el marco de la OdiloTID, 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 empleando el lenguaje de programación Java, lo que la convierte en una aplicación multiplataforma. Dentro de las características generales que posee se pueden encontrar las siguientes (25) :

- 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.

Este sistema es privativo por lo que no revela su código fuente, imposibilitando así la reutilización y análisis de la implementación del protocolo.

PARES: El Portal de Archivos Españoles (PARES), es un proyecto del Ministerio de Cultura Español destinado a la difusión en Internet del Patrimonio Histórico Documental. Fue desarrollado en Java y emplea tecnologías XML. Ofrece un acceso libre y gratuito para cualquier usuario que se encuentre interesado en acceder a los documentos con

imágenes digitalizadas de los archivos españoles. Funciona sobre una base de datos Oracle. Este portal les brinda la posibilidad a los usuarios de realizar búsquedas simples y avanzadas de los archivos históricos a través de una interfaz amigable y sencilla, además permite la interconexión con otros archivos (26).

En este sitio resulta imposible reutilizar su código fuente puesto que el mismo es propietario.

ArchiVenHIS: Este sistema fue realizado en la UCI. El mismo permite realizar la conformación del cuadro de clasificación de la documentación y la descripción de los niveles según la Norma Internacional General de Descripción Archivística ISAD (G). Se posibilita la definición de la estructura física donde se almacenan los documentos y la asociación de estos con sus representaciones digitales. Se permite además la localización de los documentos con base en los metadatos descritos con el propósito de brindar servicios (préstamos, consulta digital, digitalización, fotocopias, transcripciones) y permitir el control de los préstamos internos (27)**.** Esta solución fue realizada con una tecnología incompatible con la utilizada por el proyecto Archivo, imposibilitando la reutilización de la implementación del protocolo.

1.5 Tecnologías y herramientas utilizadas en el desarrollo

El entorno de desarrollo, metodología, herramientas de modelado y herramientas de desarrollo-, definido para la construcción del producto Xabal-Arkheia, fue producto de un estudio realizado por el equipo de arquitectura, y establecido como políticas del proyecto, por lo que la selección de los mismos queda fuera del alcance del presente trabajo. Solo se brindará una breve descripción de cada herramienta a utilizar. Las especificaciones del por qué se usaron estas herramientas se pueden encontrar en el documento de arquitectura del proyecto Archivo (28).

1.5.1 Metodología de desarrollo

Rational Unified Process (RUP): Define un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto.

Es la metodología de desarrollo de software más usada y probada en el mundo de soluciones informáticas y su objetivo es lograr un software de eficiencia y calidad, que

satisfaga las necesidades de usuarios finales. Es muy factible para el desarrollo de soluciones informáticas complejas y extensas en cronogramas de ejecución, producto de las características que posee. Cada iteración añade funcionalidades al producto o mejora las existentes (29).

1.5.2 Lenguaje de modelado

UML (Unified Modeling Language / Lenguaje unificado de modelado): Lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software. Es un sistema notacional destinado a los sistemas de modelado que utilizan conceptos orientados a objetos (30).

1.5.3 Lenguajes de desarrollo

HTML (Hyper Text Markup Languaje / Lenguaje de Marcado de Hipertexto): Es un lenguaje de marcas orientado a la publicación de documentos en internet. La mayoría de las marcas son semánticas. HTML es un lenguaje extensible, al que se le pueden añadir nuevas características, marcas y funciones. Los documentos HTML están formados por una serie de bloques de texto con una entidad lógica (titulares, párrafos, listas, entre otros). La interpretación de estas entidades se deja al navegador, lo cual da una gran flexibilidad a la presentación del documento, que puede ser mostrado, por ejemplo, en terminales gráficos o de texto (31).

JavaScript: Se caracteriza por ser un lenguaje basado en prototipos, con entrada dinámica y con funciones de primera clase. Todos los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM. Su principal importancia radica en que tradicionalmente, se venía utilizando en páginas web HTML, para realizar operaciones y en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se ejecuta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML (32).

Groovy: Es un lenguaje dinámico que se ejecuta en la máquina virtual de Java. Usa los mismos tipos de datos que Java, lo que le convierte en el complemento perfecto del lenguaje líder en desarrollo de software empresarial. Groovy y Java constituyen la plataforma de nueva generación para el desarrollo de software empresarial. Cualquier

implementación con Groovy es posible también con Java, y viceversa. Es el lenguaje que utiliza el marco de trabajo y es de muy fácil adopción para programadores Java (33)

1.5.4 Marco de trabajo

Grails 2.1.1: Marco de trabajo para aplicaciones web, libre, desarrollado sobre el lenguaje de programación Groovy. Grails pretende ser un marco de trabajo altamente productivo siguiendo paradigmas tales como: (Convención sobre Configuración / Convention Over Configuration) CoC o (No te Repitas / Don't Repeat Yourself) DRY, proporcionando un entorno de desarrollo estandarizado y ocultando gran parte de los detalles de configuración al programador (34).

Sus principales características son:

- **Alta productividad:** Grails tiene tres características que intentan incrementar su productividad comparándolo con los marcos de trabajo de Java tradicionales:
  1. Inexistencia de configuración XML.
  2. Entorno de desarrollo preparado para funcionar desde el primer momento.
  3. Funcionalidad disponible mediante métodos dinámicos.
- **Persistencia:** El modelo de datos en Grails se almacena en la base de datos utilizando Grails Object Relational Mapping (GORM).

1.5.5 Herramientas de modelado

Visual Paradigm For UML 8.0: Es una herramienta de modelado que se caracteriza por su flexibilidad ya que es multiplataforma, es decir, tiene la capacidad de ejecutarse sobre diferentes sistemas operativos. Soporta el ciclo de vida completo del desarrollo de software: Análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Utiliza UML como lenguaje de modelado, ofreciendo soluciones de software que permiten a las organizaciones desarrollar las aplicaciones de forma rápida y satisfactoria. Es fácil de usar y presenta un entorno gráfico agradable para el usuario (35).

Permite:

- Realizar el modelado de base de datos, el modelado de requerimientos, la generación de documentación y la integración con herramientas de desarrollo.

- Dibujar todos los tipos de diagramas de clases, código inverso, y generar código desde diagramas.

1.5.6 Herramientas de Desarrollo

Spring Source Tool Suite (STS) 2.5: Provee un entorno de desarrollo para la construcción de aplicaciones empresariales basadas en Spring. Spring Source Tool Suite presenta soporte para Groovy y Grails. Posee un servidor web Apache Tomcat integrado para el despliegue de aplicaciones web. También nos da la posibilidad de lanzar comandos Grails, posee un soporte de depuración mejorado, y mejoras en el tipo de inferencia y soluciones rápidas en el editor de Groovy (36).

TortoiseSVN 1.7: Es un software de sistema de control de versiones. Está desarrollado sobre software libre bajo una licencia de tipo Apache/BSD. Basa su funcionamiento en la creación de repositorios de ficheros a los cuales se accede mediante la red, lo que le permite ser usado por personas que se encuentran en distintos ordenadores. Su principal importancia radica mantener el código en una ubicación externa copia de seguridad (37).

Apache Tomcat 7.0.30: Es un contenedor web escrito en Java, por lo que funciona en cualquier sistema operativo que disponga de la Máquina Virtual de Java (JVM) y es desarrollado en un ambiente participativo y abierto.
Puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era solo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad (38).

1.5.7 Sistema Gestor de Base de Datos

PostgreSQL 9.1: Sistema de gestor de bases de datos objeto-relacional y con su código fuente disponible. Utiliza un modelo cliente/servidor. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. PostgreSQL permite grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema; es libre y multiplataforma. Permite la herencia entre tablas (39).

1.5.8 Tecnologías

(Java 2 Enterprise Edition) J2EE: Es una plataforma de programación, forma parte de la Plataforma Java para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de N capas distribuidas. Se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. (40).

Bootstrap: Es una colección de herramientas de software libre para la creación de sitios y aplicaciones web. Contiene plantillas de diseño basadas en HTML y CSS con tipografías, formularios, botones, gráficos, barras de navegación y demás componentes de interfaz, así como extensiones opcionales de JavaScript. Bootstrap proporciona un conjunto de hojas de estilo que proveen definiciones básicas de estilo para todos los componentes de HTML. Esto otorga una uniformidad independiente del navegador, lo que da una apariencia moderna para el formateo de los elementos de texto, tablas y formularios (41).

1.6 Conclusiones del capítulo

En el presente capítulo se estudiaron los diferentes SGDA y los protocolos candidatos a implementar para lograr la interoperabilidad, obteniendo las conclusiones siguientes:

- Los SGDA analizados que implementan el protocolo OAI-PMH son sistemas privativos por lo que no revelan su código fuente, poseen tecnologías no compatibles con las empleadas en la construcción del sistema para el cual se está realizando esta solución por lo que se hace imposible analizar y reutilizar la implementación del protocolo.
- De los protocolos analizados se escogió para la solución el OAI-PMH pues cumple y satisface con las necesidades existentes en el producto Xabal-Arkheia.

Capítulo 2: Características del Sistema

2.1 Introducción

El presente capítulo refleja las principales características del módulo Interoperabilidad y la descripción de los principales artefactos generados de acuerdo con la metodología empleada. Se detallan los requisitos funcionales y no funcionales que debe cumplir el sistema, así como la descripción de la arquitectura utilizada.

2.2 Modelo de Dominio

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del negocio. Los objetos del dominio representan las cosas que existen o los eventos que suceden en el entorno en el que trabaja el sistema.

El modelo del dominio se describe mediante diagramas de UML (especialmente mediante diagramas de clases). Estos diagramas muestran a los clientes, usuarios, revisores y otros desarrolladores las clases del dominio y como se relacionan unas con otras mediante asociaciones (42).

En este trabajo se realiza el modelo de dominio debido a que no existe un negocio definido, por lo cual, no se pueden determinar los procesos y roles del proceso de negocio, haciéndose engorroso y poco exacto la descripción de los mismos. Por tanto se describirán los conceptos relacionados a las clases del dominio.

2.2.1 Descripción de los principales conceptos

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: 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.

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 dependencia 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, la misma debe tener una sintaxis XML.

Proveedor de datos: Recibe las peticiones enviadas por el proveedor de servicios, las analiza y en dependencia del resultado obtenido en dicho análisis, genera una respuesta para dicha solicitud.

Documento: Contiene las descripciones de los documentos físicos.

En la siguiente figura se muestra el modelo de dominio correspondiente al módulo interoperabilidad del proyecto Archivo.

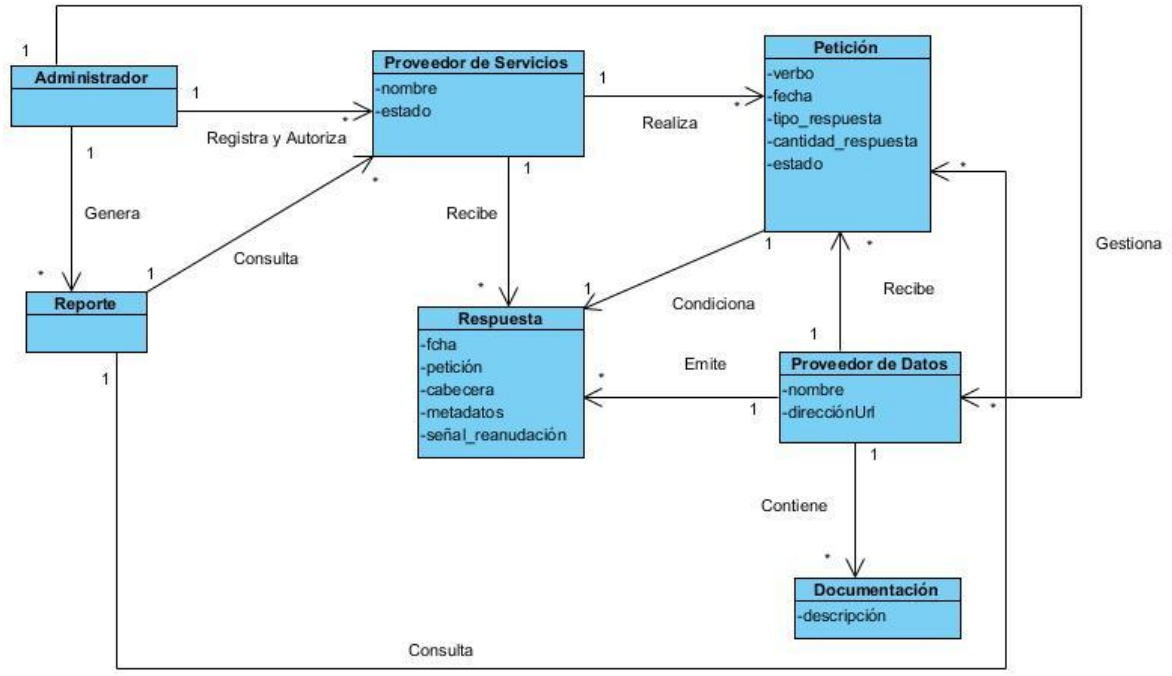

Figura 2: Modelo de Dominio del módulo Interoperabilidad.

El sistema Xabal-Arkheia cuenta con un rol administrador que gestiona varios PD para que un PS pueda realizar una determinada búsqueda. El administrador registra y autoriza

un PS a realizar peticiones, estas son recibidas por el PD el cual contiene la descripción de los documentos, este emite una respuesta condicionada por la petición realizada. El PS recibe las respuestas emitidas por el PD. El administrador genera un reporte el cual se basa en la información contenida en las peticiones y en los PS.

2.3 Requisitos

Uno de los primeros pasos en la realización de un proyecto o trabajo informático es la captura de requisitos. Estos son los aspectos con los que el sistema debe cumplir, es lo que el sistema debe de hacer, representa que es lo que desea el cliente y que información puede controlar la aplicación a desarrollar (42). En los requisitos también se define bajo que circunstancia o propiedades el sistema debe ser implementado y presentado. Una buena comprensión de los mismos puede contribuir a realizar un mejor producto que satisfaga las necesidades de las personas interesadas en el fruto del trabajo a desarrollar.

2.3.1 Técnicas de captura de requisitos

Durante el proceso de captura de requisitos se utilizan técnicas de captura para lograr una buena comunicación con los interesados en el producto. Entre las técnicas para llevar a cabo este proceso se encuentran: Las Entrevistas, Estudio de la documentación, Técnicas para facilitar las especificaciones de una aplicación (TFEA), Tormenta de ideas (brainstorming), Introspección, entre otras.

Para la captura de requisitos del módulo de interoperabilidad del producto Xabal-Arkheia se utilizaron las siguientes técnicas (43):

- **Estudio de documentación**

El estudio de documentación consiste en realizar una lectura en profundidad basada en documentos sobre el dominio del problema en el sistema a desarrollar.

Para la aplicación de esta técnica se consultó la documentación relacionada con los archivos y con la implementación del protocolo OAI-PMH, por ejemplo: Administración de documentos y archivos. Textos fundamentales (3), Gestión documental y organización de archivos (7), Protocolos de transferencia y recuperación de información (15), Guía de implementación del protocolo OAI-PMH (44). También fueron consultados los trabajos de diplomas realizados durante la creación del sistema ArchivenHis (2) (45).

- **Tormenta de ideas (brainstorming)**

Es una técnica de reuniones en grupo cuyo objetivo es la generación de ideas en un ambiente libre de críticas o juicios. Puede ayudar a generar una gran variedad de vistas del problema y a formularlo de diferentes formas, sobre todo al comienzo del proceso de captura, cuando los requisitos no se encuentran bien definidos (43).

Para el desarrollo del módulo interoperabilidad se realizaron varias reuniones donde intervinieron los arquitectos de sistema, los principales analistas y el jefe del proyecto donde en las mismas se determinaron el 70% de los requisitos existentes.

2.4 Especificación de Requisitos

En la especificación de requisitos, se registran las características definidas que debe cumplir cada requisito funcional.

Requisitos Funcionales

Los requisitos funcionales describen las funcionalidades del sistema. Los requisitos del módulo interoperabilidad del producto Xabal-Arkheia se encuentran organizados por los módulos que intervienen en la solución, proveedores de datos y proveedores de servicios.

Para el módulo **proveedor de datos** se encuentran:

RF_1 - Recolectar metadatos.

El sistema debe permitir que cualquier proveedor de servicios autorizado establezca una conexión para recolectar información.

RF_2- Proveer metadatos en formato Dublin Core.

El sistema debe brindar la posibilidad de devolver los metadatos recolectados en formato Dublin Core.

RF_3- Proveer metadatos en formato EAD.

El sistema debe brindar la posibilidad de devolver los metadatos recolectados en formato EAD.

.**RF_4- Validar petición.**

El sistema debe evaluar cada petición para así determinar si la misma está correcta o no.

RF_5- Controlar el flujo de la transmisión de metadatos.

El sistema debe ser capaz de dividir y controlar las respuestas en caso de que las mismas sean extensas (más de 50 resultados).

RF_6- Gestionar proveedores de servicios.

El sistema debe permitir que el administrador gestione los datos de los proveedores de servicios que podrán recolectar metadatos.

RF_7- Controlar acceso a los datos.

El sistema debe ser capaz de verificar a través de la dirección ip si un proveedor de servicios tiene acceso a la información almacenada.

RF_8- Describir servidor.

El sistema debe permitir realizar una descripción de sí mismo para responder a las peticiones de tipo Identify.

Para generar diferentes **reportes** se encuentran los requisitos:

RF_9- Mostrar cantidad de peticiones recibidas atendiendo al verbo.

El sistema debe permitir que se consulte la cantidad de peticiones recibidas atendiendo al verbo de la petición.

RF_10- Mostrar cantidad de metadatos que ha recolectado cada proveedor de servicios.

El sistema debe permitir mostrar la cantidad de metadatos que cada proveedor de servicios ha recolectado en el sistema.

RF_11- Exportar reporte a formato PDF.

El sistema debe permitir que los reportes realizados sean exportados en formato pdf.

Por el módulo **proveedor de servicios** se encuentran:

RF_12- Gestionar proveedores de datos.

El sistema debe permitir que el administrador del sistema gestione los proveedores de datos en los que el proveedor de servicios podrá realizar una búsqueda.

RF_13- Buscar documentos en los proveedores de datos.

El sistema debe permitir realizar búsquedas sobre las descripciones de los documentos contenidos en proveedores de datos externos.

RF_14- Mostrar resultados de la búsqueda.

El sistema debe permitir mostrar los resultados de la búsqueda realizada.

RF_15- Visualizar descripción.

El sistema debe permitir mostrar una descripción ampliada de cada uno de los resultados obtenidos en las búsquedas.

RF_16- Describir archivos históricos.

El sistema debe registrar datos de un archivo histórico para mostrarlos como un valor agregado en los resultados obtenidos de las búsquedas en proveedores de datos.

RF_17- Modificar datos de la descripción de archivos históricos.

El sistema debe ser capaz de modificar la información registrada de un archivo determinado.

RF_18- Ver detalles de los archivos históricos.

El sistema debe ser capaz de mostrar toda la información registrada de un archivo determinado.

Requisitos no Funcionales

Los requerimientos no funcionales, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes del mismo (42).

RNF_1. Requerimientos de Hardware.

El sistema debe contar con dos servidores, uno para la base de datos y otro para la aplicación. Estos deben tener 2 GB de RAM, un micro a una frecuencia de 2.3 GHZ, 100 GB de disco duro el de base de datos y 50 GB el de aplicaciones.

RNF_2. Requerimientos de Software.

Usuario:

En las computadoras de los usuarios solo se requiere de un navegador Web Mozilla Firefox 17 o Google Chrome 23 además de un lector de pdf Adobe Reader 10 o en sus versiones superiores.

Servidor:

En el caso del servidor de aplicaciones web se deberá tener instalado cualquier versión igual o superior al Apache Tomcat 7.0.30 y en el servidor de base de datos se debe tener instalado PostgreSQL 9.1, además de una Máquina Virtual de java 7 y como sistema operativo Windows o Linux.

RNF_3. Requerimientos de apariencia o interfaz externa.

El sistema debe tener una apariencia profesional y un diseño gráfico sencillo, con la utilización de las tonalidades de los colores rojo, blanco y gris fundamentalmente.

El texto incluido en las páginas de la aplicación debe ser de tipo Arial 11.

Debe mostrar mensajes de información tanto para el éxito o fallido de una operación.

RNF_4. Requerimientos de Seguridad.

- **Confidencialidad.**

Sólo pueden acceder a la información los usuarios que posean permisos.

- **Integridad.**

Sólo pueden modificar información los usuarios que posean permisos para realizar esta acción.

- **Disponibilidad**

El sistema debe estar disponible las 24 horas del día por los 7 días de la semana.

RNF_5. Requerimientos de rendimiento.

El tiempo promedio de respuesta por transacción no debe exceder de 1 min, debido a que el sistema es una herramienta web que realiza transferencias de documentos.

RNF_6. Requerimientos de portabilidad.

El sistema podrá ser usado bajo cualquier sistema operativo ya sea Linux o Windows.

2.5 Técnicas de validación

Luego de realizar la descripción de los requisitos, los mismos necesitan ser validados. La validación de requisitos tiene como misión demostrar que la definición de estos realmente posean las características que tendrá el sistema y lo que el cliente realmente desea Existen diferentes técnicas para este proceso como son: Auditorías, Reviews o Walk-throughs, Prototipos. Las utilizadas para el desarrollo de este trabajo fueron (46):

- **Reviews o Walk-throughs**

Esta técnica consiste en la lectura y corrección de la completa documentación o modelado de la definición de requisitos. Con ello solamente se puede validar la correcta interpretación de la información transmitida.

Esta técnica fue llevada a cabo por los analistas del proyecto Archivo. Con esta se detectaron algunas inconveniencias como fueron:

- ✓ Presencia de errores ortográficos y de concordancia.
- ✓ Ocurrencia de palabras ambiguas.

Se realizaron algunas recomendaciones de cambios de textos para que se entendiera y facilitara la implementación al desarrollador. Todas estas irregularidades fueron solucionadas y aceptadas permitiendo así la obtención de una correcta documentación para lograr una mayor interpretación de la descripción de las funcionalidades del sistema.

- **Prototipos**

El prototipo de interfaz de usuario es una técnica de representación aproximada de la interfaz de usuario de un sistema, permite a los clientes y usuarios entender más fácilmente la propuesta de los requisitos para resolver sus problemas de negocio**.**

Con la utilización de esta técnica fueron aceptados el 90% de los requisitos propuestos realizándoles modificaciones al resto. Las modificaciones ejecutadas fueron:

- ✓ Modificación de nombres de los componentes de la interfaz.
- ✓ Cambios de tipos de componentes.
- ✓ Eliminación y adición de botones.

Esta técnica facilitó la construcción de las vistas del sistema y el entendimiento de la propuesta de requisitos por parte de los analistas, arquitectos y jefe de proyecto.

Los prototipos correspondientes al módulo Interoperabilidad se pueden encontrar en los anexos de la investigación.

2.6 Casos de uso

Los casos de uso son muy útiles para explicar el funcionamiento del sistema. Existen varios motivos por los cuales los mismos se han hecho populares y se han adoptado universalmente. Las dos razones fundamentales son (42) :

- Proporcionan un medio sistemático e intuitivo de capturar requisitos funcionales centrándose en el valor añadido para el usuario.
- Dirigen todo el proceso de desarrollo debido a que la mayoría de las actividades como el análisis, diseño y prueba se llevan a cabo partiendo de los casos de uso. El diseño y las pruebas pueden también planificarse y coordinarse en términos de casos de uso.

2.6.1 Diagrama de casos de uso

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar.

Su ventaja principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente (47).

El diagrama de casos de uso que se corresponde con el desarrollo de este trabajo se muestra a continuación.

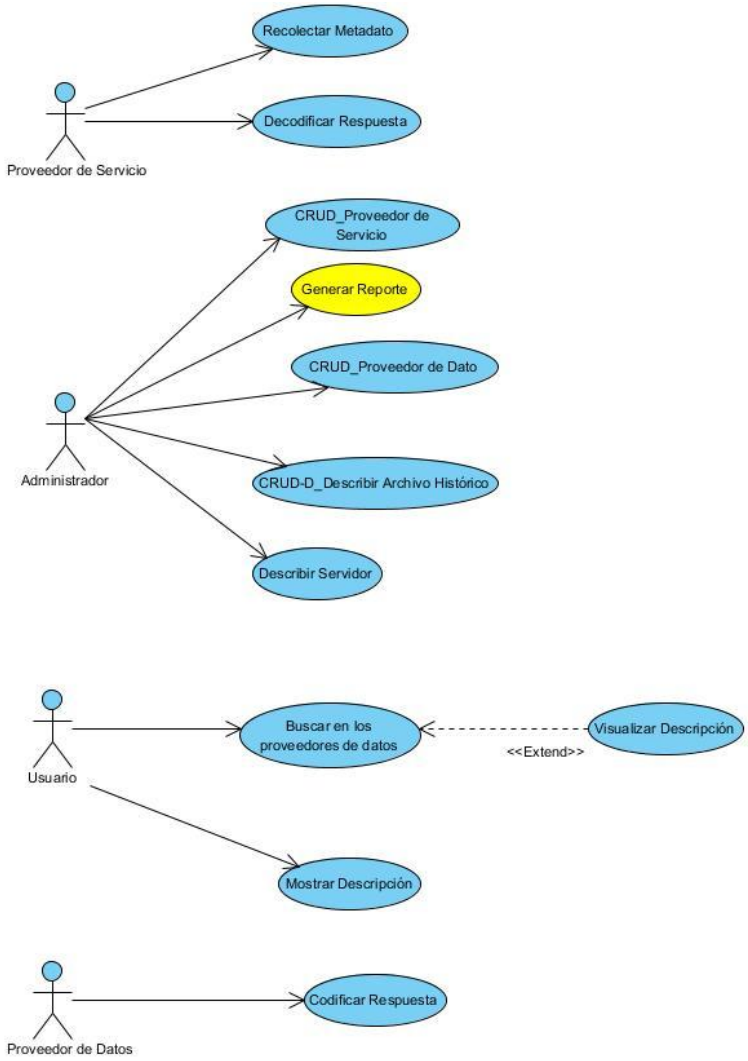

Figura 3: Diagrama de Casos de Usos.

2.7 Patrones de Casos de Uso

Un patrón puede ser entendido como un conjunto de elementos interrelacionados tales como problema, contexto y solución. El hecho de encontrar un patrón adecuado para un problema dado depende en gran medida de una correcta identificación del problema en sí y del contexto en que este ocurre (48).

Los patrones de casos de uso son comportamientos que deben existir en el sistema, ayudan a describir qué es lo que el sistema debe hacer, es decir, describen el uso del sistema y cómo este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen como debería ser estructurados y organizados los casos de uso. Son patrones que capturan mejores prácticas para modelar casos de uso. Los patrones utilizados para identificar los casos de uso fueron (49):

CRUD (Creating, Reading, Updating, Deleting)

Se basa en la fusión de casos de uso simples para formar una unidad conceptual.

- **Completo**

Este patrón consta de un caso de uso, llamado Información CRUD o Gestionar que modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio, y estos a su vez son cortos y simples.

- **Parcial**

Una de las alternativas del caso de uso puede ser modelada como caso de uso independiente. Este patrón es preferible cuando uno de los flujos alternativos del caso de uso es más significativo, muy largo o mucho más complejo que el patrón completo.

Extensión Concreta

Consiste en una relación extendida entre dos casos de uso. Esta relación se utiliza cuando el caso de uso extendido no es de obligatoria ejecución en el caso de uso al que el mismo extiende.

Nombres que revelan la intención (Intention Revealing Name)

El nombre debe reflejar la intención del caso de uso y reflejar un único objetivo e intención que el actor está intentando lograr. Se debe asignar un nombre apropiado que facilite el manejo del caso de uso, permitiendo tener una vista general del trabajo en su conjunto.

Escenario + Fragmentos (Scenario Plus Fragments)

El flujo principal debe describir cómo el actor logra su propósito. Debe contener todos los procedimientos o pasos a realizar para que se ejecute la funcionalidad descrita en el caso de uso.

Alternativas Exhaustivas, Integras (Exhaustive Alternatives)

Este patrón establece que un caso de uso puede tener varias alternativas, identificándose en cada caso el flujo normal de eventos y capturando los posibles fallos.

2.8 Descripción de Casos de Uso del Sistema

La descripción de casos de uso permite al desarrollador un mayor entendimiento de la funcionalidad que va a implementar. En el desarrollo de este trabajo se tomaron de estudio dos casos de uso, los cuales se consideran de alta prioridad para el desarrollo del módulo Interoperabilidad. La descripción del resto de los casos de uso puede consultarse en los anexos de esta investigación.

Caso de Uso: **Recolectar metadatos**

Objetivo	Recolectar metadatos de interés.
Actores	Proveedor de Servicio
Resumen	El sistema permite que cualquier proveedor de servicios autorizado establezca una comunicación con el proveedor de datos, con el objetivo de recolectar sus metadatos.
Complejidad	Media
Prioridad	Alta
Precondiciones	El proveedor de servicios debe estar registrado.
Postcondiciones	

Flujo de eventos		
Flujo básico Recolectar metadato		
	Actor	**Sistema**
1.	Realiza una petición al sistema.	
2.		Verifica que el Proveedor de Servicios tiene permisos para recolectar metadatos.
3.		Verifica que la petición cumpla con el formato del protocolo OAI-PMH.
4.		Obtiene los metadatos de los documentos que coincidan con el criterio de búsqueda.
5.		Codifica los metadatos obtenidos en formato XML. **Ver CU Codificar Respuesta.**
6.		Devuelve los datos de la petición en formato XML.
7.		Termina el caso de uso.
Flujo alterno		
2ª Acceso prohibido		
	Actor	Sistema
1.		Muestra un mensaje indicando que no tiene acceso a los datos.
2.		Termina el caso de uso.
3ª Petición incorrecta		
	Actor	Sistema
1.		Muestra un mensaje indicando que la petición no corresponde con el formato establecido.
2.		Termina el caso de uso.
Relaciones	**CU Incluidos**	No aplica
	CU Extendidos	No aplica
Requisitos no funcionales		
Asuntos pendientes		

Tabla 3: Descripción del CU Recolectar metadatos

Caso de Uso: **Buscar en los proveedores de datos**

Objetivo	Buscar en los proveedores de datos metadatos de interés.
Actores	Usuario
Resumen	El proveedor de servicios le envía una petición a los proveedores de datos seleccionados por el usuario con el fin de que estos recolecten los metadatos correspondientes al criterio de búsqueda introducido por el usuario.
Complejidad	Alta
Prioridad	Alta

Precondiciones	Debe estar registrado y descrito al menos un proveedor de datos.	
Postcondiciones		
Flujo de eventos		
Flujo básico Aprobar Descripción		
	Actor	**Sistema**
1.	Selecciona la opción Buscar.	
2.		Muestra un formulario con los parámetros ➢ Título. ➢ Nombre de los productores. Proveedores de datos y la opción "Buscar".
3.	Introduce los parámetros de búsqueda y selecciona la opción "Buscar".	
4.		Comprueba que se ha seleccionado al menos un proveedor de datos.
5.		Comprueba que se han introducido los datos obligatorios.
6.		Realiza la recolección de metadatos en cada proveedor de datos seleccionado a partir de los criterios de búsqueda introducido por el usuario. **Ver CU Recolectar metadatos.**
7.		Decodifica los datos. **Ver CU Decodificar Respuesta.**
8.		Muestra los resultados y las opciones Visualizar Descripción y Mostrar descripción del Archivo.
9.		Termina el caso de uso
Flujos alternos		
4a Seleccionar proveedor		
	Actor	**Sistema**
4a.1		Muestra un mensaje indicando que se debe seleccionar al menos un proveedor de datos para realizar a búsqueda.
4a.2		Ir a la acción 3.
5a Datos obligatorios		
	Actor	**Sistema**
5a.1		Muestra un mensaje indicando que debe introducirse al menos un criterio de búsqueda.
5a.2		Ir a la acción 1.
8a Visualizar Descripción		
	Actor	**Sistema**
1.	Selecciona la opción "Visualizar Descripción"	
2.		Ver **CU Visualizar Descripción de Documentos de Archivo.**
8b Mostrar descripción del Archivo		
	Actor	**Sistema**

1.	Selecciona la opción “Mostrar descripción del Archivo”	
2.		Ver **CU Mostrar descripción del Archivo.**
Relaciones	**CU Incluidos**	
	CU Extendidos	CU Visualizar Descripción
Requisitos no funcionales		
Asuntos pendientes		

Tabla 4: Descripción del CU Buscar en los proveedores de datos.

2.9 Arquitectura del sistema

La arquitectura del sistema está basada en la propuesta por Grails. La misma es una arquitectura N-Capas, se basa en el patrón Modelo-Vista-Controlador (MVC). El mismo establece que los componentes de un sistema de software deben organizarse en 3 capas distintas según su misión (50):

- Modelo, o capa de datos: Contiene los componentes que representan y gestionan los datos manejados por la aplicación. En el caso más típico los objetos encargados de leer y escribir en la base de datos.
- Vista o capa de presentación: Los componentes de esta capa son responsables de mostrar al usuario el estado actual del modelo de datos, y presentarles las distintas acciones disponibles.
- Capa de control: Contendrá los componentes que reciben las órdenes del usuario, gestionan la aplicación de la lógica de negocio sobre el modelo de datos, y determinan que vista debe mostrarse a continuación.

Grails propone además que la lógica de negocios no debe ser implementada en los controladores, sino que esta se debe implementar una capa extra denominada Capa de Servicios.

- Capa de Servicios: Responsable de implementar, manejar y validar la lógica de negocios de la aplicación. Maneja directamente las clases de dominio permitiendo la inyección de métodos dinámicos (50).

2.10 Patrones de Diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Los patrones de arquitectura

expresan un esquema organizativo estructural fundamental para sistemas de software (30).

Los patrones utilizados para el diseño son:

Inversión de control

La Inversión de Control es un patrón de diseño pensado para permitir un menor acoplamiento entre componentes de una aplicación y fomentar así la reutilización de los mismos. Plantea que la creación de los objetos se debe realizar por un componente externo (30). Grails implementa este patrón por defecto para la capa de los servicios, mediante la implementación de un contenedor que se encarga de administrar el ciclo de vida de las instancias. Para utilizarlo solo es necesario declarar las variables de tipo def y con el nombre exactamente igual al del servicio y Grails se encarga de hacer el resto (50).

Patrones GRASP

Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP es el acrónimo para General Responsibility Assignment Software Patterns (Patrones Generales de Software para Asignar Responsabilidades) (30).

- **Experto:** Se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad.

- **Creador:** Permite identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. La clase que implementa el patrón tiene la información necesaria para realizar la creación del objeto, usa directamente las instancias creadas del objeto y almacena o maneja varias instancias de la clase.

- **Alta cohesión y bajo acoplamiento:** Los conceptos de cohesión y acoplamiento están íntimamente relacionados. Un mayor grado de cohesión implica un menor de acoplamiento.

    - ✓ **Alta Cohesión:** Define que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con

la clase. Asigna una responsabilidad de forma tal que la cohesión siga siendo alta.

- ✓ **Bajo Acoplamiento**: Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

- ➢ **Controlador:** Sirve de intermediario entre una determinada interfaz y las operaciones del sistema.

Singleton

Consiste en la existencia de una instancia única de una clase en el contexto de la aplicación, así como la creación de un mecanismo de acceso global a dicha instancia (30). Grails utiliza este patrón por defecto en cada uno de los servicios, ya que al ejecutar la aplicación una instancia de cada servicio será creada y almacenada en memoria, dicha instancia será destruida solo al finalizar la ejecución de la aplicación. Este comportamiento se puede alterar si al servicio se le define la variable scope, la cual indica cuando será creada y destruida cada instancia de dicho servicio (50).

Agente Remoto

El patrón Agente Remoto sugiere crear una clase de software local que represente el servicio externo y asignarle la responsabilidad de contactar el servicio real donde el principal problema es que el sistema debe comunicarse con un componente situado en el espacio de otra dirección (30).

Estrategia

Define una familia de algoritmos, encapsula uno de ellos y los hace intercambiables. Permite que un algoritmo varíe independientemente de los clientes que lo usan. Este patrón se utiliza cuando varias clases relacionadas sólo difieren en su comportamiento. Permite configurar a una clase con uno de entre varios comportamientos se necesitan variantes del mismo algoritmo, que se implementan como una jerarquía de clases (51).

2.11 Diagramas de clases

Los diagramas de clase muestran un conjunto de clases, interfaces y colaboraciones y las relaciones entre estos; los diagramas de clases muestran el diseño de un sistema desde un punto de vista estático (42).

2.11.1 Diagramas de clases del diseño

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos (42).

A continuación se representan los diagramas de clases del diseño con estereotipos web de los CU Recolectar metadatos y Buscar en los proveedores de datos.

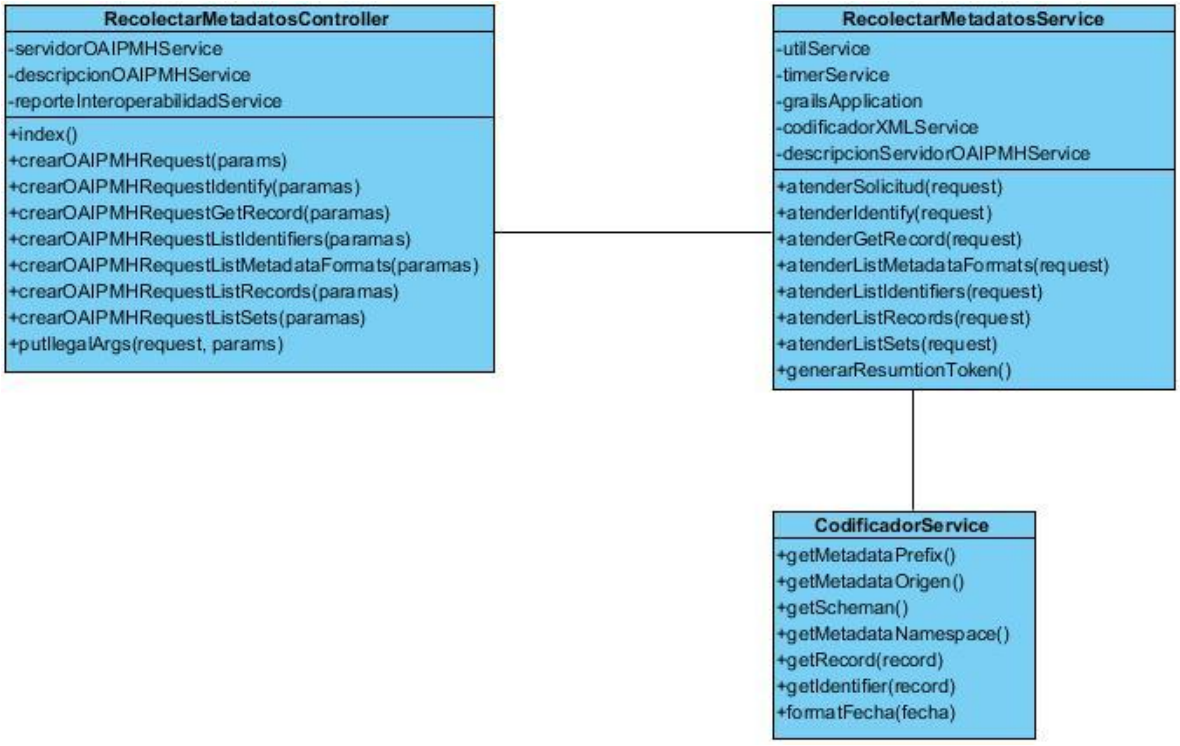

Figura 4: Diagrama de clases del diseño CU Recolectar metadatos.

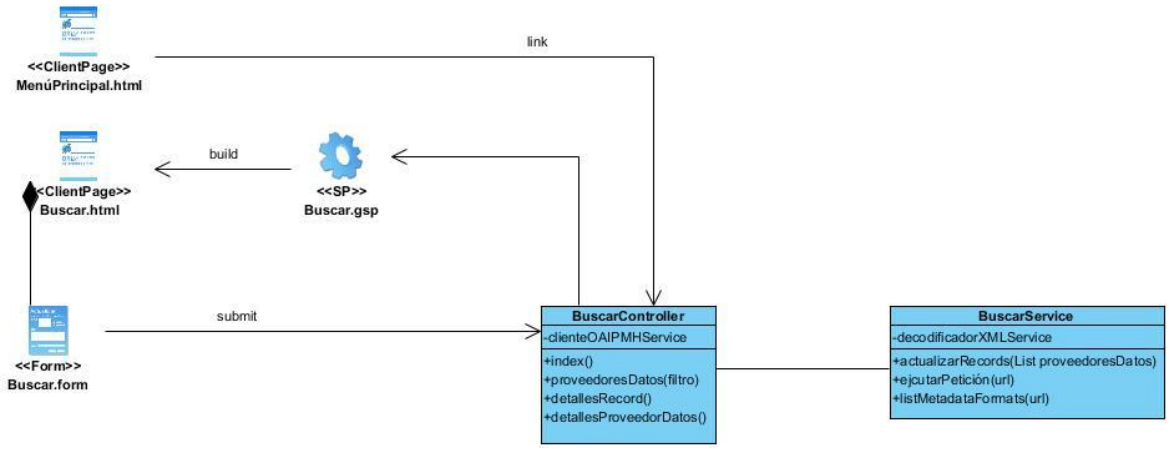

Figura 5: Diagrama de clases del diseño CU Buscar en los proveedores de datos.

2.11.2 Aplicación de patrones en el diagrama de clases del diseño

El patrón Experto se pone de manifiesto en los servicios pues son los que contienen la información necesaria para ejecutar la lógica de negocio de la aplicación.

El patrón Controlador se evidencia en los controladores pues son los encargados de gestionar el flujo de las peticiones realizadas por el usuario.

El patrón Alta Cohesión se demuestra entre las clases controladoras y los servicios pues las mismas colaboran entre sí para realizar una determinada funcionalidad.

El patrón Bajo Acoplamiento se pone de manifiesto en la interacción de todas las clases del sistema, pues el mismo está divido en capas, con lo cual se logra que la interacción entre las clases implementadas sea mínima.

El Creador se pone de manifiesto en los servicios pues los mismos son los encargados de crear los objetos del sistema como parte de la lógica de negocio.

La explicación del resto de los patrones: Inversión de Control, Singleton, Agente Remoto y Estrategia se encuentran en el siguiente capítulo, pues estos se explican con imágenes pertenecientes a la implementación del módulo.

2.12 Diagramas de interacción

El diagrama de interacción, consistente en un conjunto de objetos y sus relaciones, incluyendo los mensajes que pueden ser enviados entre ellos para denotar la relación

existente entre los mismos. Estos diagramas se utilizan para mostrar la vista dinámica de un sistema, entre ellos se encuentran: Los diagramas de colaboración y secuencia (42).

2.12.1 Diagramas de secuencia

El diagrama de secuencia es un esquema conceptual que permite representar el comportamiento de un sistema, para lo cual emplea la especificación de los objetos que se encuentran en un escenario y la secuencia de mensajes intercambiados entre ellos, con el fin de llevar a cabo una transacción del sistema (52).

A continuación se representan los diagramas de secuencia de los CU Recolectar metadatos y Buscar en los proveedores de datos. El resto de los diagramas se pueden encontrar en los anexos de la investigación.

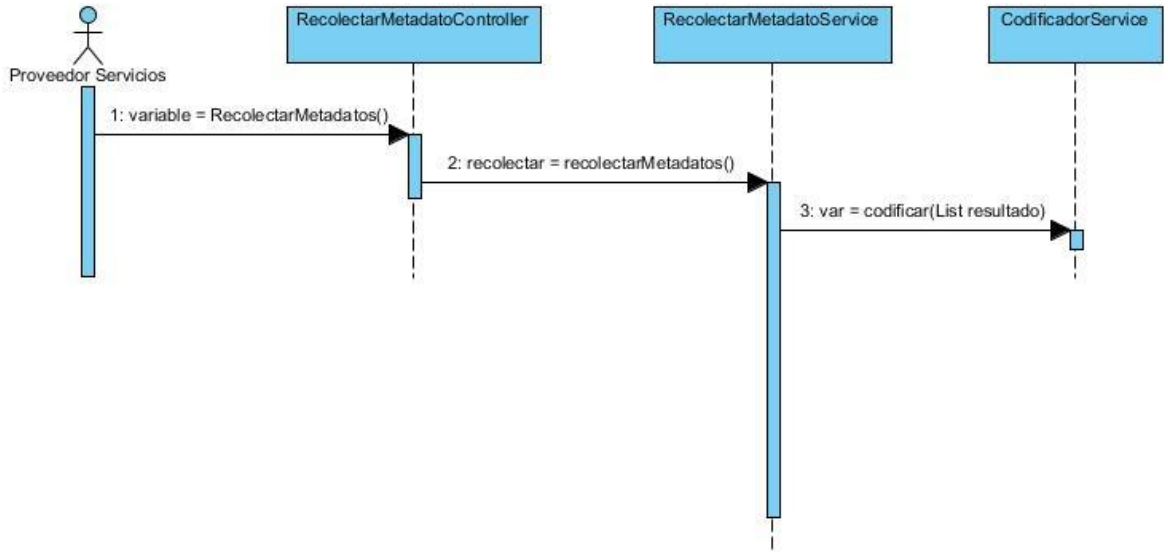

Figura 6: Diagrama de Secuencia del CU Recolectar metadatos.

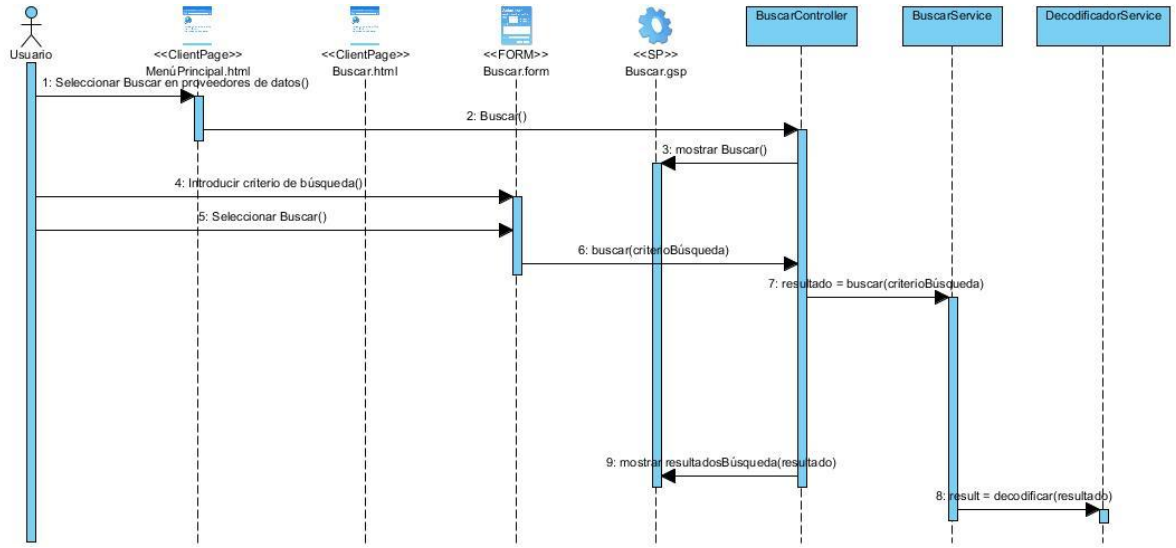

Figura 7: Diagrama de Secuencia del CU Buscar en los proveedores de datos.

2.13 Validación del diseño propuesto

Una métrica es un instrumento que cuantifica un criterio y persigue comprender mejor la calidad del producto, estimar la efectividad del proceso y mejorar la calidad del trabajo realizado al nivel del proyecto (53).

Para la evaluación de la calidad del diseño propuesto para el sistema se hizo un estudio de las métricas básicas inspiradas en la calidad del diseño orientado a objeto, en el mismo se abarcan atributos de calidad que permiten medir la calidad del diseño propuesto. Dentro de estos se encuentran (53):

- **Responsabilidad:** consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- **Complejidad de implementación:** consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
- **Reutilización:** consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- **Acoplamiento:** consiste en el grado de dependencia o interconexión de una clase o estructura de clase con otras, está muy ligada a la característica de Reutilización.
- **Complejidad del mantenimiento:** consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error

de un diseño de software. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.

- **Cantidad de pruebas:** consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad del producto diseñado.

Las métricas concebidas como instrumento para evaluar la calidad del diseño y su relación con los atributos de calidad definidos son las siguientes:

TOC (Tamaño Operacional de Clase)

Se refiere al número de métodos pertenecientes a una clase. La siguiente tabla muestra los atributos que forman parte de esta métrica y el modo en que se afectan.

Atributo que afecta	Modo en que lo afecta
Responsabilidad	Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un aumento del TOC implica una disminución en el grado de reutilización de la clase.

Tabla 5: Tamaño operacional de clase.

Esta métrica está determinada por los atributos: Responsabilidad, Complejidad de implementación y la Reutilización, existiendo una relación directa con los dos primeros e inversa con el último antes mencionado.

La tabla que se muestra a continuación contiene el rango de valores para la evaluación técnica de los atributos de calidad (Responsabilidad, Complejidad de Implementación y Reutilización). La variable Prom indica el promedio de operaciones por cada clase.

Atributos	Categoría	Criterio
Responsabilidad	Baja	< =Prom.
	Media	Entre Prom. Y 2* Prom
	Alta	> 2* Prom
Complejidad de implementación	Baja	< =Prom
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom
Reutilización	Baja	> 2*Prom
	Media	Entre Prom. y 2* Prom
	Alta	<= Prom

Tabla 6: Rango de valores.

La siguiente tabla muestra los umbrales de la métrica TOC.

Tamaño operacional de la clase	Criterio
Pequeña	<= Prom.
Media	Entre Prom. y 2 * Prom.
Grande	> 2* Prom.

Tabla 7: Umbrales para el TOC.

Resultados del instrumento de evaluación de la métrica TOC

Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos, (ver figura 8).

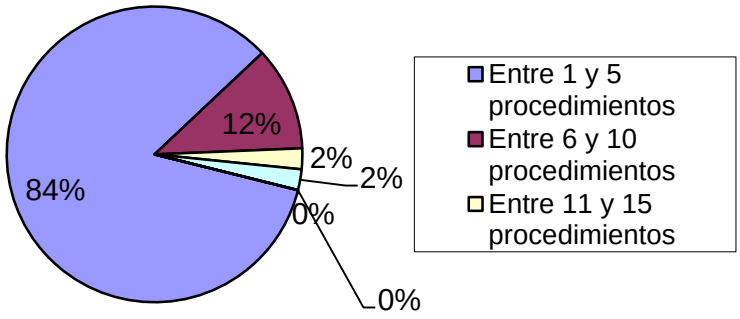

Figura 8: Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad, (ver figura 9**)**.

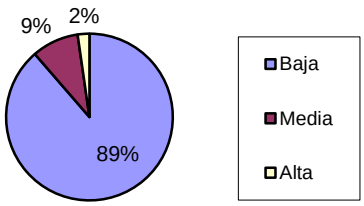

Figura 9: Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad.

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Complejidad de implementación, (ver figura 10).

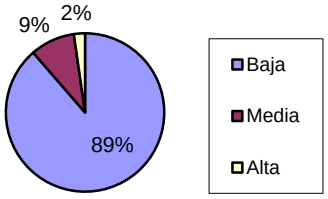

Figura 10: Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Complejidad de implementación

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Reutilización, (ver figura 11).

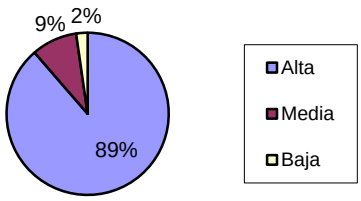

Figura 11: Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Reutilización.

Al analizar los resultados obtenidos luego de aplicar el instrumento de medición de la métrica TOC, se puede concluir que el diseño propuesto para el sistema es simple y tiene una calidad aceptable, teniendo en cuenta que la mayoría de las clases (84%) posee menos cantidad de operaciones que la media registrada en las mediciones. Los atributos de calidad se encuentran en un nivel satisfactorio en el 89% de las clases, de manera que se puede observar cómo se fomenta la Reutilización (elemento clave en el proceso de desarrollo de software) y cómo están reducidas en menor grado la Responsabilidad y la Complejidad de implementación.

RC (Relaciones entre Clases)

Esta métrica está dada por el número de relaciones de uso de una clase. La siguiente tabla muestra los atributos y pertenecientes a esta métrica y el modo en que se afectan.

Atributo que afecta	Modo en que lo afecta
Acoplamiento	Un aumento del RC implica un aumento del Acoplamiento de la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
Cantidad de pruebas	Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.
Reutilización	Un aumento del RC implica una disminución en el grado de reutilización de la clase.

Tabla 8: Relaciones entre clases (RC).

Esta métrica está determinada por los atributos: Acoplamiento, Complejidad de mantenimiento, Cantidad de pruebas y Reutilización, existiendo una relación directa con los tres primeros e inversa con el último antes mencionado.

La siguiente tabla muestra el Rango de valores para la evaluación técnica de los atributos de calidad (Acoplamiento, Complejidad de mantenimiento, Reutilización y Cantidad de Pruebas) relacionados con la métrica RC. La variable Prom indica el promedio de relaciones entre clases.

Atributos	Categoría	Criterio
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	2
	Alto	>2
Complejidad de mantenimiento	Baja	< =Prom
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom
Reutilización	Baja	> 2*Prom
	Media	Entre Prom. y 2* Prom
	Alta	<= Prom
Cantidad de Pruebas	Baja	< =Prom
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom

Tabla 9: Rango de valores para la evaluación técnica de los atributos de calidad (Acoplamiento, Complejidad de mantenimiento, Reutilización y Cantidad de Pruebas) relacionados con la métrica RC.

Resultados del instrumento de evaluación de la métrica Relaciones entre Clases

Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos, (ver figura 12).

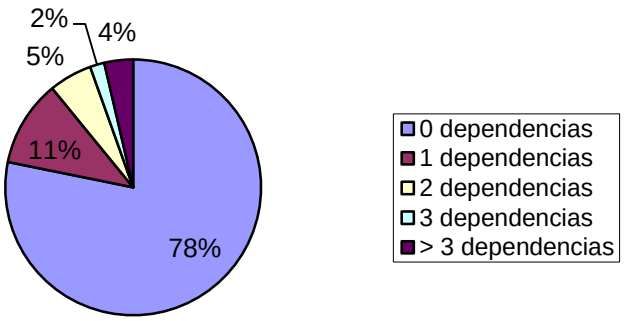

Figura 12: Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Acoplamiento, (ver figura 13).

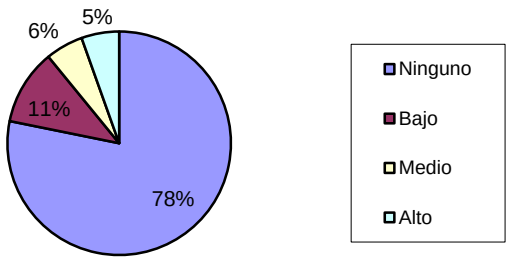

Figura 13: Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Acoplamiento.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Complejidad de mantenimiento, (ver figura 14).

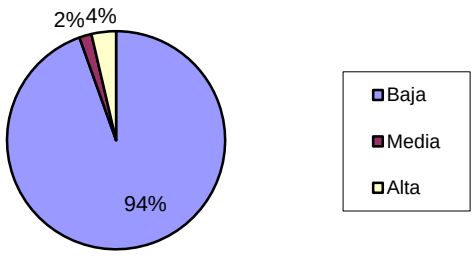

Figura 14: Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Complejidad de mantenimiento.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Cantidad de pruebas, (ver figura 15).

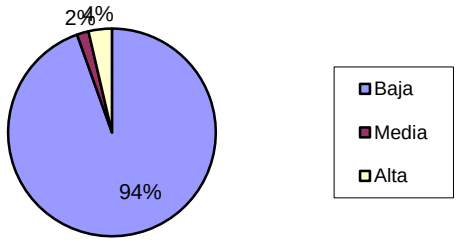

Figura 15: Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Cantidad de pruebas.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Reutilización, (ver figura 16).

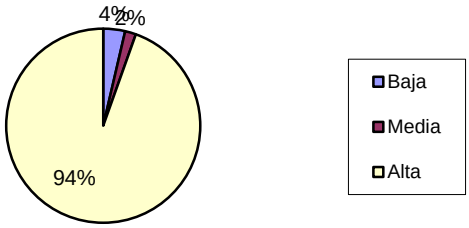

Figura 16: Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Reutilización.

Al analizar los resultados obtenidos luego de aplicar el instrumento de medición de la métrica RC, se puede concluir que el diseño propuesto para el sistema es simple y tiene una calidad aceptable, teniendo en cuenta que la mayoría de las clases (94%) poseen 2 o menos dependencias respecto a otras. Los atributos de calidad se encuentran en un nivel satisfactorio, en el 89% de las clases el grado de acoplamiento es mínimo, la Complejidad

de mantenimiento, la Cantidad de pruebas y la Reutilización se comportan favorablemente para un 94% de las clases.

2.14 Base de datos

Grails para la generación de la base de datos se basa en las clases de dominio. Estas representan las entidades persistentes que son mapeadas hacia una tabla de la base de datos. Las formas en que se generarán las tablas de la base de datos dependen de las relaciones que se establezcan entre ellas, por ejemplo:

- Una relación de uno a uno se define creando en una clase de dominio un atributo de la otra como se puede evidenciar en la (figura7) en la cual la clase DProveedorDatos tiene un atributo de tipo DINormaISADIAH la cual generará una llave foránea en la tabla DProveedorDatos que hace referencia a la tabla DINormaISADIAH.

```
class DProveedorDatos extends Entidad {

    DINormaISDIAH descripcion
    String urlProveedor
    DMetadataFormat metadataFormat
    Date ultimaActualizacion

    static constraints = {...}

    static DProveedorDatos clone(DProveedorDatos origen) {...}

    def beforeUpdate = {
        DProveedorDatos proveedorDatos = DProveedorDatos.get(id)
        if (proveedorDatos) {
            descripcion = proveedorDatos.descripcion
        }
    }
}
```

Figura 17: Clase de dominio donde se evidencia la relación de uno a uno.

- La relación de uno a muchos se representa a través del atributo hasMany como se evidencia en la (figura8). Donde la clase DIAreaControl representa con el hasMany su relación con las clases DIRegla, DILengEsc y DIFuente, lo cual genera una tabla intermedia entre las tablas DIAreaControl y cada una de las anteriores. Estas tablas intermedias contienen dos llaves foráneas, una que representa a la tabla DIAreaControl y otra que representa a las tablas DIRegla, DILengEsc y DIFuente respectivamente.

```
package arkheia.domain.interoperabilidad.isdiah.areaControl
import ...

class DIAreaControl extends Entidad implements Serializable {

    String identidicador
    String idenInstitucion
    String estElaboracion
    String nivelDetalle
    Date fechaCreacion
    Date ultimaRevision
    String notaMantenimiento

    Collection reglas
    Collection lengsEsc
    Collection fuentes

    static hasMany = [reglas: DIRegla, lengsEsc: DILengEsc, fuentes: DIFuente]
    static belongsTo = [norma:DINormaISDIAH]

    static constraints = {...}
}
```

Figura 18: Clase de dominio donde se representa la relación de uno a muchos.

2.14 Conclusiones parciales

- En este capítulo se presentó un modelo de dominio para un mejor entendimiento del contexto del sistema, puesto que no existe un negocio definido.
- Se especificaron los requisitos del módulo quedando estos validados por el equipo de analistas y directivos del proyecto Archivo.
- Teniendo en cuenta la arquitectura descrita se diseñó el módulo interoperabilidad donde se generaron los artefactos para los casos de uso Recolectar metadatos y Buscar en los proveedores de datos.
- Fueron expuestas las métricas para validar el diseño, las cuales arrojaron resultados satisfactorios sobre el diseño realizado, demostrando que este era simple y los atributos de calidad alcanzaban niveles favorables.

El desarrollo de todas las actividades descritas permitió adquirir una mejor visión de los principios que guiaron la implementación.

Capítulo 3: Implementación y Prueba

3.1 Introducción

La implementación del sistema comienza luego de realizar el análisis de la solución y el modelado de los artefactos que genera la metodología. Como parte de la implementación se muestran los diagramas de componentes y de despliegue. Además, se definen las pruebas del software como elemento crítico para la garantía de la calidad del sistema.

3.2 Implementación

En esta disciplina los elementos del diseño, fundamentalmente las clases, se implementan en términos de componentes, como ficheros de código fuente, ejecutables, librerías, entre otros. También se describe como dependen los componentes unos de otros.

3.2.1 Implementación de las clases del Dominio

Las clases del dominio se encuentran ubicadas en la carpeta grails-app/domain, formando el modelo de datos de la aplicación.

3.2.1.1 Operaciones sobre el modelo de datos

Las entidades de la aplicación poseen métodos dinámicos que facilitan el trabajo con las mismas.

- save (): Para insertar el registro en la base de datos o para actualizarlo si ya existe. Usando el argumento (flush: true) los datos se envían a la base de datos en el mismo momento en que se invoca el método save ().
- delete (): Para eliminar un registro.
- Dynamic Finders: Son un conjunto de métodos dinámicos que inserta grails para el acceso a los datos (get, list, findBy, findAllBy).
  - ✓ get (): Obtener un objeto por el id.
  - ✓ List (): Listar todos los objetos de la clase correspondiente.
  - ✓ findBy (): Obtener un objeto a partir del atributo que se especifique.
  - ✓ findAllBy (): Obtener todos los objetos a partir del atributo que se especifique.

A continuación se muestra un ejemplo de la implementación de las clases del dominio pertenecientes al módulo Interoperabilidad.

```
package arkheia.domain.interoperabilidad.isdiah.areaControl
import ...

class DIAreaControl extends Entidad implements Serializable {

    String identidicador
    String idenInstitucion
    String estElaboracion
    String nivelDetalle
    Date fechaCreacion
    Date ultimaRevision
    String notaMantenimiento

    Collection reglas
    Collection lengsEsc
    Collection fuentes

    static hasMany = [reglas: DIRegla, lengsEsc: DILengEsc, fuentes: DIFuente]
    static belongsTo = [norma:DINormaISDIAH]

    static constraints = {...}
}
```

Figura 19: Implementación de la clase dominio OAIPMHRequestGetRecord.

La clase OAIPMHRequestGetRecord.groovy pertenece a la implementación del CU Recolectar metadatos, es la encargada de obtener información a partir del criterio de búsqueda introducido por el usuario. Los atributos que componen esta clase son:

- **Identifier:** Atributo que almacena la identificación del objeto del cual se desea obtener información.
- **MetadataPrefix:** Atributo que almacena bajo que norma el proveedor de servicios desea obtener la información que se requiere.

Estos atributos no pueden ser nulos puesto que para obtener datos de una petición utilizando este verbo es necesario conocer su identificación y el formato en el que se desea recolectar los metadatos pertinentes. Esta clase hereda de OAIPMHRequest para acceder a la información de la misma, contiene el método getArgumentos para obtener las propiedades de sus atributos de la forma (propiedades: valor).

3.2.2 Implementación de las clases Controladoras

Las clases controladoras se encuentran ubicadas en la carpeta grails-app/controllers, y son las responsables de recibir las solicitudes del usuario, gestionar la aplicación de la lógica del negocio y decidir la vista que se debe mostrar a continuación.

A continuación se muestra un ejemplo de la implementación de la clase controladora correspondiente al CU Recolectar metadatos.

```
class ServidorOAIPMHController {

    def servidorOAIPMHService
    def descripcionServidorOAIPMHService

    def index() {
        def oaiRequest = crearOAIPMHRequest(params)
        String respuesta = servidorOAIPMHService.atenderSolicitud(oaiRequest)
        response.setContentType("text/xml")
        render respuesta
    }

    private def crearOAIPMHRequest(def params) {...}

    private OAIPMHRequest _crearOAIPMHRequest(def params) {...}

    private OAIPMHRequestIdentify _crearOAIPMHRequestIdentify(def params) {...}

    private OAIPMHRequestGetRecord _crearOAIPMHRequestGetRecord(def params) {...}

    private OAIPMHRequestListIdentifiers _crearOAIPMHRequestListIdentifiers(def params) {...}

    private OAIPMHRequestListMetadataFormats _crearOAIPMHRequestListMetadataFormats(def params) {...}

    private OAIPMHRequestListRecords _crearOAIPMHRequestListRecords(def params) {...}

    private OAIPMHRequestListSets _crearOAIPMHRequestListSets(def params) {...}

    private void putIlegalArgs(OAIPMHRequest request, def params) {...}
}
```

Figura 20: Implementación de la clase controladora ServidorOAIPMHController.

En esta clase se atienden las peticiones que realizan los usuarios. Contiene el método index () que es el encargado de recibir las peticiones y un conjunto de métodos para crear instancias de las peticiones.

Los objetos servidorOAIPMHService y descripcionServidorOAIPMHService se utilizan para atender las peticiones y obtener la descripción del servidor respectivamente.

3.2.3 Implementación de las clases Servicios

Según la convención que sigue Grails, un servicio es una clase cuyo nombre termina en Service y se encuentran ubicados en la carpeta grails-app/services. Estas clases manejan la lógica de negocios del sistema.

A continuación se muestra un ejemplo de la implementación de la clase servicio correspondiente al módulo Interoperabilidad.

```
class ServidorOAIPMHService {

    def codificadorXMLService
    def describirServidorOAIPMHService

    String atenderSolicitud(OAIPMHRequest request) {
        if (!request.validate()) {
            if (request.error == ErrorOAIPMH.badVerb) {
                return codificadorXMLService.codificarRespuesta(codificadorXMLService.badVerb(), request)
            } else if (request.error == ErrorOAIPMH.badArgument) {
                return codificadorXMLService.codificarRespuesta(codificadorXMLService.badArgument(), request)
            }
        }
        request.path = describirServidorOAIPMHService.get().baseURL
        return this."_atender${request.verb}"(request)
    }

    private String _atenderIdentify(OAIPMHRequest request) {...}

    private String _atenderGetRecord(OAIPMHRequest request) {...}

    private String _atenderListMetadataFormats(OAIPMHRequest request) {...}

    private String _atenderListIdentifiers(OAIPMHRequest request) {...}

    private String _atenderListRecords(OAIPMHRequest request) {...}

    private String _atenderListSets(OAIPMHRequest request) {...}
}
```

Figura 21: Implementación de la clase ServidorOAIPMHService.

La clase ServidorOAIPMHService.groovy es uno de los servicios utilizados en la implementación del CU Recolectar metadatos, esta clase es la encargada de atender una determinada petición.

Se utiliza el objeto codificadorXMLService para codificar la respuesta de la búsqueda realizada y el objeto describirServidorOAIPMHService para obtener la url del servidor.

3.2.4 Internacionalización

La internacionalización es la codificación de forma tal que los mensajes que se muestra en la aplicación puedan aparecen en diferentes idiomas sin la necesidad de reprogramar toda la aplicación (50).

Grails tiene soporte para la internacionalización por medio de archivos de recursos almacenados en la carpeta grails-app/i18n. Los archivos de recursos son ficheros de propiedades en los que se guardan las distintas versiones de cada mensaje en los idiomas deseados. Cada idioma está definido en un fichero distinto que incluye en su nombre el código del idioma, por ejemplo: messages_en.properties para los mensajes en inglés y messages_es.properties para los mensajes en español internacional.

A continuación se muestra un ejemplo del fichero de internacionalización:

```
dINombre.title=Nombre
dINombre.nombre.label=Nombre

dProveedorServicios.title=Proveedor de Servicios
dProveedorServicios.nombre.label=Nombre
dProveedorServicios.estadoPS.label=Estado
dProveedorServicios.correoElectronico.label=Correo electrónico
dProveedorServicios.telefono.label=Teléfono

estadoPS.title=Estado del Proveedor de Servicios
estadoPS.Activo.label=Activo
estadoPS.Deshabilitado.label=Deshabilitado
```

Figura 22: Fichero de internacionalización para el módulo Interoperabilidad

Luego de tener especificadas las claves en el fichero de internacionalización, se define en la interfaz de usuario una etiqueta en la que se hace referencia a una de las claves contenidas en el fichero, esta etiqueta mostrará el valor que tiene asignado la clave utilizada (figura 12) el código resaltado en azul.

```
<f:textField label="${message(code: "dProveedorServicios.nombre.label")}" name="nombre
<f:textField label="${message(code: "dProveedorServicios.correoElectronico.label")}" n
<f:textField label="${message(code: "dProveedorServicios.telefono.label")}" name="tele
<f:select label="${message(code: "dProveedorServicios.estadoPS.label")}"class="requeri
```

Figura 23: Utilización del fichero de internacionalización en las vistas.

3.2.5 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y las relaciones entre ellos. Nos permiten observar con más facilidad la estructura general de un sistema de software y el comportamiento de los servicios que estos componentes proporcionan. Además, permiten especificar componentes con interfaces bien definidas y en él se muestran los elementos de diseño del sistema. Proveen una vista arquitectónica de alto nivel y ayudan a los desarrolladores a visualizar el camino de la implementación, así como a tomar decisiones sobre ella (53).

3.2.5.1 Descripción de Componentes

A continuación se representa el diagrama de componentes correspondiente al módulo Interoperabilidad. En el mismo se puede apreciar la relación de uso que tiene dicho módulo con otros componentes del sistema Xabal-Arkheia.

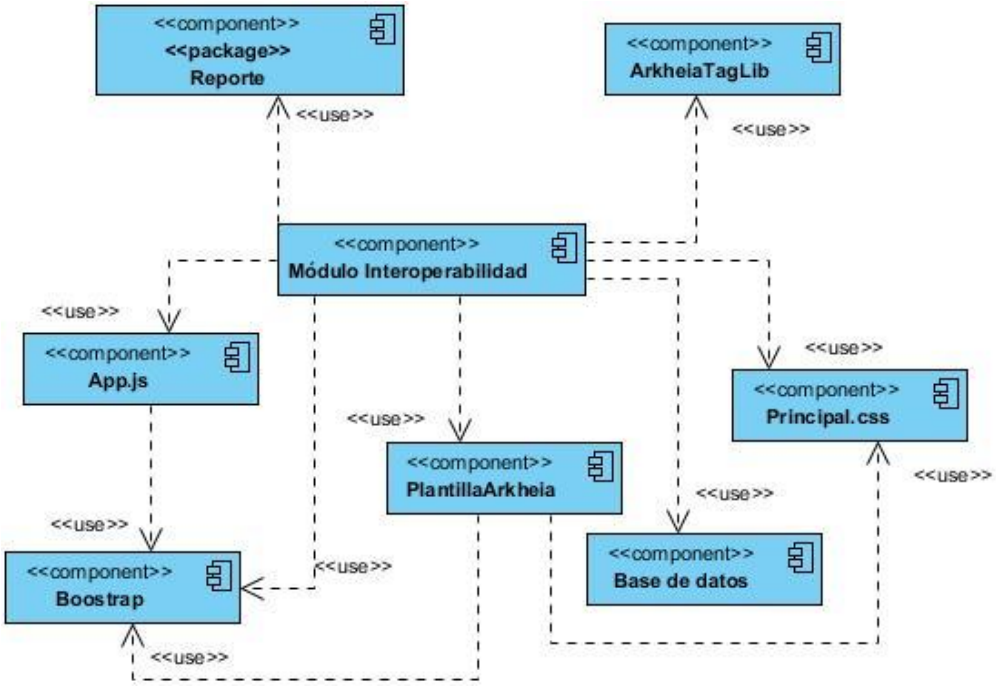

Figura 24: Componentes relacionados con el módulo Interoperabilidad.

Para la generación de reporte es utilizado el componente Reporte. La librería de etiquetas personalizadas ArkheiaTagLib es un componente utilizado para aumentar la reutilización de código y disminuir la cantidad del mismo.

En el componente App.js se encuentran todas las funcionalidades javascript comunes para toda la aplicación, por ejemplo: Funciones para controlar los componentes de tipo fecha, la ventanas emergentes, entre otras.

Se hace uso del componente Principal.css pues es donde se encuentran los estilos del sistema. El componente PlantillaArkheia se utiliza para evitar la redundancia de código, esta utiliza la librería javascript Boostrap para la creación y animación de los componentes de interfaz de usuario y el componente Base de datos para el almacenamiento y gestión de los datos.

La siguiente figura muestra cómo se relacionan algunos de los componentes que forman parte del módulo.

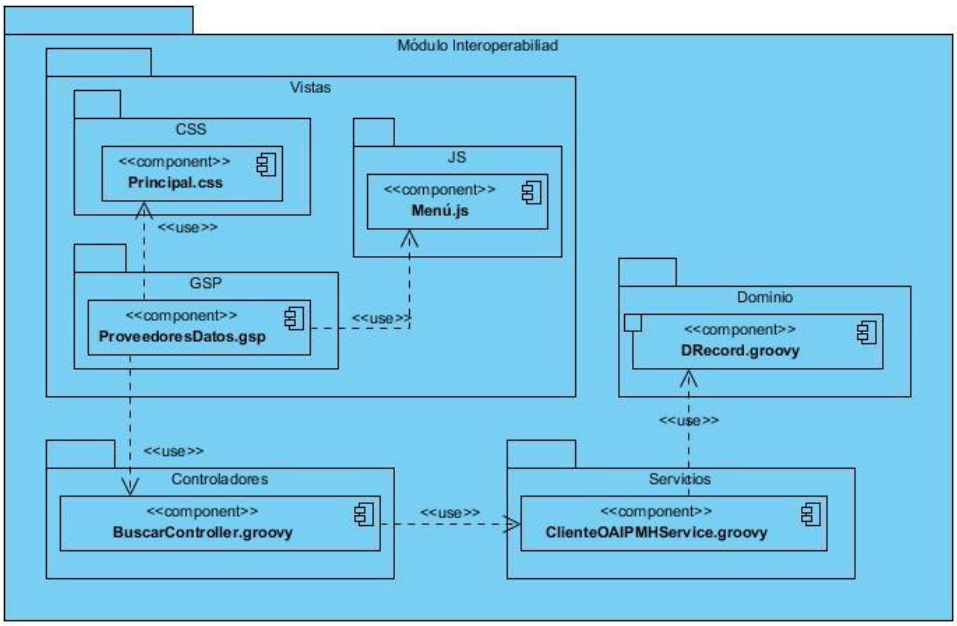

Figura 25: Diagrama de Componentes del módulo Interoperabilidad.

El módulo está compuesto por un paquete vistas, controladores, servicios y dominios. El paquete Vistas pertenece a la Capa Web de la arquitectura, dentro se encuentran los ficheros de tipo GSP (las vistas) que representan la interfaz de usuario del sistema el cual usa los ficheros CSS para los estilos del sistema y los JS que son ficheros JavaScript.

El paquete Controladores pertenece a la Capa Web de la arquitectura, dentro se encuentran las clases controladoras que son las encargadas de atender las peticiones

que proceden de las vistas, así como también de brindar los datos necesarios para satisfacer al usuario.

El paquete Servicios pertenece a la Capa Servicios de la arquitectura. Dentro se encuentran los servicios, que son las clases encargadas de almacenar, extraer y consultar los datos para responder a los pedidos de los controladores.

El paquete Dominio pertenece la Capa de Datos de la arquitectura. Dentro se encuentran las clases del dominio, las que se corresponden con las entidades, donde se guarda toda la información de la aplicación.

3.2.6 Tratamiento de errores

El tratamiento de errores es la detección de los mismos a través de la utilización del bloque de instrucciones try-catch para evitar la interrupción de la ejecución de la aplicación. Este método se utiliza en todos los fragmentos de código que puedan generar errores.

```
Verbo verb
try {
    verb = Verbo."${params.verb}"
} catch (Exception e) {}
```

Figura 26: Utilización del bloque de instrucciones try-catch.

La figura anterior muestra la obtención de un enumerador a partir de la cadena de texto almacenada en “params.verb”, esta operación de no existir un valor del enumerador igual al contenido en la cadena de texto puede generar un error por lo que se utiliza dentro del bloque de instrucciones try-catch.

3.2.7 Diagrama de despliegue

Los diagramas de despliegue muestran a los nodos procesadores la distribución de los procesos y de los componentes (30).

El despliegue de la solución para el módulo se realizará según como se muestra en el siguiente diagrama.

Figura 27: Diagrama de despliegue para el módulo Interoperabilidad.

3.2.8 Aplicación de patrones

En la siguiente figura se pone de manifiesto el patrón Inversión de Control.

```
class ServidorOAIPMHService {

    def utilService
    def timerService
    def grailsApplication
    def codificadorXMLService
    def descripcionServidorOAIPMHService


    private String[] caracteres = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N"
    private int[] valores = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

    String atenderSolicitud(DOAIPMHRequest request) {...}

    private String _atenderIdentify(DOAIPMHRequest request) {...}

    private String _atenderGetRecord(DOAIPMHRequestGetRecord request) {...}

    private String _atenderListMetadataFormats(DOAIPMHRequestListMetadataFormats request) {...}

    private String _atenderListIdentifiers(DOAIPMHRequestListIdentifiers request) {...}

    private String _atenderListRecords(DOAIPMHRequestListRecords request) {...}

    private String _atenderListSets(DOAIPMHRequestListSets request) {...}

    private String generarResumtionToken() {...}
}
```

Figura 28: Ejemplo del uso del patrón Inversión de Control.

El patrón **Inversión de Control** se evidencia en la clase ServidorOAIPMHService mediante la declaración de los objetos codificadorXMLService y descripcionServidorOAIPMHService. La utilización de la palabra def le indica a la aplicación la necesidad de un objeto de tipo CodificadorXMLService y DescripcionServidorOAIPMHService respectivamente. Estos objetos serán inyectados dinámicamente en esta clase por el manejador de objetos de Spring.

La siguiente figura muestra un ejemplo del uso del patrón **Singleton**.

```
class ReportesPDService {//reportesPD/cantPeticionesPorVerbo

    List cantPeticionesPorVerbo () {...}

    List cantPeticionesPorProveedor () {...}
}
```

Figura 29: Ejemplo de uso del patrón Singleton.

El patrón **Singleton** se pone de manifiesto en la clase ReportesPDService. Al ejecutar la aplicación se crea una única instancia de esta clase que es destruida cuando se detiene la ejecución de la aplicación.

La siguiente figura muestra la clase donde se evidencia el patrón **Agente Remoto**.

```
class ClienteOAIPMHService {

    def decodificadorXMLService

    void actualizarRecords (List proveedoresDatos) {...}

    List listMetadataFormtas(String url) throws Exception {...}

    private String ejecutarPeticion(String url) throws Exception {...}
}
```

Figura 30: Ejemplo de uso del patrón Agente Remoto.

El patrón **Agente Remoto** se pone de manifiesto en la clase ClienteOAIPMHController pues esta contiene una representación local de cada operación del PD.

El uso del patrón **Estrategia** se pone de manifiesto en las siguientes figuras.

```
class CodificadorISADG_EADService implements Codificador {
    @Override
    String getMetadataPrefix() {...}

    @Override
    String getMetadataOrigen() {...}

    @Override
    String getSchema() {...}

    @Override
    String getMetadataNamespace() {...}

    @Override
    String getRecord(DNorma record) {...}

    @Override
    String _getIdentifier (DNorma record) {...}

    @Override
    String listSets() {...}

    String _getRecord(DNorma record) {...}

    private String _formatFecha(Fecha fecha) {...}
}
```

Figura 31: Ejemplo de uso del patrón Estrategia.

```
class CodificadorISADG_DCService implements Codificador {
    @Override
    String getMetadataPrefix() {...}

    @Override
    String getMetadataOrigen() {...}

    @Override
    String getSchema() {...}

    @Override
    String getMetadataNamespace() {...}

    @Override
    String getRecord(DNorma record) {...}

    @Override
    String _getIdentifier(DNorma record) {...}

    @Override
    String _getRecord(DNorma record) {...}

    @Override
    String listSets() {...}

    private String _formatFecha (Fecha fecha) {...}
}
```

Figura 32: Ejemplo de uso del patrón Estrategia

El patrón Estrategia se pone de manifiesto en las clases CodificadorISADG_EADService y CodificadorISADG_DCService. Las mismas constituyen dos implementaciones diferentes de un mismo algoritmo de codificación y pueden ser utilizadas una o la otra en dependencia de las necesidades del sistema.

3.3 Seguridad

La seguridad informática es el conjunto de métodos y herramientas destinados a proteger los bienes (o activos) informáticos de una institución. La información es el activo más preciado y la seguridad en la información tiene el objetivo de garantizar (54):

- **Confidencialidad**: La información o los activos informáticos son accedidos solo por las personas autorizadas para hacerlo.
- **Integridad**: Los activos o la información solo pueden ser modificados por las personas autorizadas y de la forma autorizada.
- **Disponibilidad**: Los activos informáticos son accedidos por las personas autorizadas en el momento requerido.

La seguridad en el lado del servidor fue tratada mediante el empleo de dos técnicas fundamentales:

- Utilización de Spring Security**:** Spring Security brinda dos métodos fundamentales para evitar accesos no autorizados en las diferentes partes de la aplicación, los mismos son:
  - ✓ Filtrado de URL. Método en el cual se define el nivel de acceso a cada URL posible de la aplicación. Este método no es factible para la solución en cuestión debido a la gran cantidad de URLs existentes.
  - ✓ Anotaciones: Las anotaciones permite definir el nivel de acceso a cada una de las acciones de cada controlador, una vez definidas las funcionalidades que se encuentran en el mismo. Este tipo de restricción es el empleado en la solución y su uso se puede evidenciar en la siguiente figura.

```
@Secured("hasRole('GESTIONAR_PROVEEDOR_SERV')")
class ServidorOAIPMHController {
```

Figura 33: Utilización del Spring Security.

- Dirección ip: Esta técnica es utilizada por el proveedor de datos puesto que solo deja acceder a su información a aquel proveedor de servicios que se haya registrado previamente (uno de los campos del formulario de registro es la dirección ip).

Un punto fundamental en la seguridad de toda aplicación lo constituye la seguridad ante ataques de inyección de SQL y HTML. Grails cuenta con varios mecanismos de seguridad para evitar dichos ataques, tales como:

- El código SQL generado en GORM se escapa para evitar inyección de SQL.
- Cuando se genera HTML, los datos se escapan antes de mostrarse.

El escapado de caracteres consiste en sustituir caracteres que tienen funcionalidad por secuencias que representan el carácter pero no la funcionalidad (50).

3.4 Pruebas

Probar es una práctica habitual de todo proceso productivo, que básicamente consiste en comprobar que un producto tiene las características deseadas.

Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación, calidad, o usabilidad de un sistema informático. Básicamente es una fase en el desarrollo de software, consistente en probar las aplicaciones construidas. Para determinar el nivel de calidad se deben efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto de las especificaciones iniciales del sistema. Existen diferentes tipos y métodos de pruebas que juntos conforman la estrategia de prueba a seguir para lograr la calidad del producto final (55).

3.4.1 Método de Prueba

Los métodos de prueba son utilizados con el propósito de descubrir fallos y no para demostrar que el software funciona**.** Existen diferentes métodos de prueba como son las pruebas de caja blanca y pruebas de caja negra. Para el desarrollo de esta investigación se escogió el método de pruebas de caja negra pues es la estrategia de pruebas definida por el proyecto Archivo.

Las pruebas de caja negra son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno de un programa (56). Estas son llevadas a cabo en la interfaz del sistema. Para la realización de las mismas es necesaria la descripción de los requisitos donde se especifican los parámetros de entradas y las posibles respuestas que el sistema debe emitir al realizar una acción determinada.

3.4.2 Tipos de prueba

Como parte de la estrategia de prueba, se definió como tipo de prueba la siguiente:

- Funcionalidad: Se centra en las funciones, entradas y salidas. Estas pruebas tienen como objetivo comprobar las funcionalidades del sistema.

3.4.3 Técnica de prueba

Existen diversas técnicas de pruebas, las técnicas dinámicas y las técnicas estáticas. Las dinámicas ejecutan el software, para ello se introducen una serie de valores de entrada, y examinan la salida comparándola con los resultados esperados. Las estáticas permiten encontrar las causas de los defectos (57).

Para la validación de este trabajo se escogió la técnica dinámica basada en casos de uso.

Pruebas de casos de Uso

Las pruebas de caso de uso son una técnica que ayuda a identificar casos de prueba que ejerciten el sistema entero transición a transición desde el principio al final. Un caso de uso es una descripción de un uso particular del sistema. Sirven como fundamento para desarrollar casos de prueba. Los casos de uso describen el flujo de proceso a través de un sistema basado en su uso más probable. Esto hace que los casos de prueba obtenidos de los casos de uso sean particularmente útiles a la hora de encontrar defectos en el uso real del sistema (57).

3.4.4 Diseño de caso de prueba para el CU Recolectar metadatos

Escenario	Respuesta del Sistema	Flujo Central
EC 1.1: Recolectar metadatos.	Recolecta los metadatos encontrados dependiendo de la petición realizada por	1. Seleccionar la opción Buscar en proveedores de datos.

	el proveedor de servicios. Si este no está registrado o la petición no cumple con el formato del protocolo, el sistema muestra un mensaje de error.	2. Introducir criterio de búsqueda. 3. Seleccionar al menos un proveedor de datos. 4. Seleccionar la opción "Buscar". 5. Devuelve los metadatos recolectados

Tabla 10: Resumen del diseño de caso de prueba del CU Recolectar metadatos

3.4.5 Diseño de caso de prueba para el CU Buscar en los proveedores de datos

Escenario	Respuesta del Sistema	Flujo Central
EC 1.1: Buscar en los proveedores.	Muestra el listado de documentos obtenidos que coinciden con el criterio de búsqueda introducido. Los criterios de búsqueda son: Título y Nombre de los productores. Muestra además las siguientes opciones: • Ver Descripción del Archivo. • Ver Descripción del documento.	1. Seleccionar la opción "Búsqueda en proveedores de datos". 2. Introducir el criterio de búsqueda deseado. 3. Selecciona al menos un proveedor de datos. 4. Seleccionar la opción "Buscar".
EC 1.2: No se ha especificado ningún criterio de búsqueda.	Muestra un mensaje que debe introducir al menos un criterio de búsqueda.	1. Seleccionar la opción "Búsqueda en proveedores de datos". 2. Selecciona al menos un proveedor de datos. 3. Seleccionar la opción "Buscar".
EC 1.3: No se ha seleccionado ningún proveedor de datos.	Muestra un mensaje indicando que debe seleccionar al menos un proveedor de datos.	1. Seleccionar la opción "Búsqueda en proveedores de datos". 2. Introducir el criterio de búsqueda deseado. 3. Seleccionar la opción "Buscar".

Tabla 11: Resumen del diseño de caso de prueba del CU Buscar en los proveedores de datos

No.	Nombre del campo	Clasificación	Valor Nulo	Descripción
1.	Título	Campo de texto	Si	Cadena de caracteres
2.	Nombre de los productores	Campo de texto	Si	Cadena de caracteres

Tabla 12: Variables del CU Buscar en los proveedores de datos

3.4.6 Resultados

Al módulo se le realizaron 3 iteraciones de pruebas donde se encontraron irregularidades de tipo funcionalidad, validación y no correspondencia con el diseño de caso de prueba. En la primera iteración se encontraron 10 no conformidades, en la segunda 3 y en la última no se encontraron anomalías, corroborando así el correcto funcionamiento del sistema (figura 16).

Figura 34: Gráfico perteneciente a las no conformidades por iteraciones encontradas en el módulo Interoperabilidad.

A continuación se muestran las no conformidades encontradas en los casos de uso Recolectar metadatos y Buscar en los proveedores de datos.

No.	No Conformidad.	Ubicación	Estado
1	Al no seleccionar un proveedor de datos y seleccionar la opción Buscar, no se muestra el mensaje de error.	CU Buscar en los proveedores de datos.	Resuelta
2	Al presionar la opción Buscar, muestra el resultado encontrado pero no la opción Ver descripción del documento.	CU Buscar en los proveedores de datos.	Resuelta
3	La respuesta del XML no muestra los caracteres con tilde, el mismo	CU Recolectar metadatos.	Resuelta

	muestra símbolos extraños.		
4	Al seleccionar la opción Ver Descripción del Archivo no muestra la descripción del mismo.	CU Buscar en los proveedores de datos.	Resuelta

Tabla 13: No conformidades encontradas en los casos de uso Recolectar metadatos y Buscar en los proveedores de datos.

3.5 Conclusiones Parciales

En este capítulo se realizó la implementación y validación del sistema, lo que permitió obtener importantes resultados como:

- Se implementó el módulo Interoperabilidad comprobando que el mismo cumple con todos los requisitos y funcionalidades especificadas permitiendo la comunicación con diferentes SGDA, resolviendo así la problemática existente en el Xabal-Arkheia.
- Se realizaron pruebas de funcionalidad, basadas en casos de uso aplicando el método de caja negra, donde se obtuvo la liberación del sistema por CALISOFT luego de tres iteraciones de pruebas donde se detectaron y resolvieron las 13 no conformidades.

Conclusiones Generales

Como resultado principal de esta investigación se obtuvo la solución del módulo Interoperabilidad el cual garantiza el intercambio de información entre el sistema Xabal-Arkheia y otros SGDA para evitar la duplicidad de información en los archivos y lograr intercambiar información entre ellos para reconocer, procesar y usar esa información satisfactoriamente. Como parte de la solución:

- Se realizó el estudio de las aplicaciones de gestión de archivo y los protocolos de comunicación que se utilizan para lograr la interoperabilidad concluyéndose que no pueden ser empleados o adaptados para dar solución al problema planteado, resultando necesario la implementación del nuevo módulo.
- Se describieron las herramientas, lenguajes y la metodología de desarrollo para el trabajo en el módulo destacando las características que las hacen factibles para la propuesta de solución.
- Se analizó y diseñó el módulo, obteniéndose un conjunto de artefactos que guiaron y facilitaron la implementación del mismo.
- Se validó el diseño propuesto utilizando las métricas TOC y RC, llegándose a la conclusión de que el diseño elaborado para el sistema es simple y presenta una calidad aceptable.
- Se implementó el módulo Interoperabilidad, que garantiza la comunicación e intercambio de información con diferentes SGDA, generando los diagramas de componentes y despliegue que muestran la distribución física del módulo.
- Se validó el funcionamiento del módulo de interoperabilidad del sistema Xabal-Arkheia realizando pruebas de caja negra, corroborando así el buen funcionamiento del mismo.

Recomendaciones

Con el propósito de enriquecer la solución propuesta del módulo Interoperabilidad se recomienda:

- Implementar nuevos formato para proveer los datos.
- Recuperar metadatos en otros formatos.
- Publicar la URL a través de la cual se podrán realizar peticiones OAI-PMH al sistema.
- Implementar un sistema de indexado para mejorar el tiempo de respuesta.

Bibliografía

1. **Molinet Machuat, Yurisleivy .** Análisis y diseño de la Aplicación Web para la interconexión de archivos históricos que emplean el Sistema de Gestión de Documentos de Archivo ArchiVenHIS. 2011.

2. **Villar, Villar, Yanet.** Módulo para garantizar la prestación de servicios según el estándar OAI-PMH para el Sistema de Gestión de Documentos Históricos. La Habana : s.n., 2012.

3. **Cruz Mundet , José Ramón.** Administración de documentos y archivos. Textos fundamentales. Madrid : s.n., 2011. ISBN: 978-84-615-5150-7.

4. **Ministerio de Educación, Cultura y Deporte del Gobierno de España.** Diccionario. Diccionario de Terminología Archivística. [En línea] [Citado el: 22 de enero de 2013.] http://www.mcu.es/archivos/MC/DTA/Diccionario.html#archivo.

5. **Gavilán, César Martín.** Concepto y función de archivo. Clases de archivos. El Sistema Archivístico Español. 2009.

6. **Fernández Gil , Paloma.** Manual de organización de archivos de gestión en las oficinas municipales. Las Gabias : CEMCI, 1999. I.S.B.N.:84-88282-43-5.

7. **Mena Mugica, Mayra.** Gestión documental y organización de archivos. La Habana : Félix Varela, 2005. ISBN 959-258-950-X.

8. **Grupo de trabajo 2: Interoperabilidad del Repositorio.** El caso de Interoperabilidad para Repositorios de Acceso Abierto. 2011.

9. Descripción archivística. Automatización de archivos en internet. [En línea] [Citado el: 22 de enero de 2013.] http://www.vicentgimenez.net/curs/descrip.htm.

10. **Comité de Normas de Descripción.** ISAD (G) : Norma Internacional General de Descripción Archivística. Madrid : s.n., 2000.

11. **Carpenter, Leona.** OAI para principiantes: Introducción. OAI para principiantes - tutorial en línea del Open Archives Forum. [En línea] [Citado el: 23 de enero de 2013.] http://travesia.mcu.es/portalnb/jspui/html/10421/1823/page1.htm#top.

12. **Krichel, Manuel Barrueco, José y Thomas.** Acceso a prepublicaciones sobre economia: RePEc. El profesional de la información. [En línea] [Citado el: 23 de enero de 2013.] http://www.elprofesionaldelainformacion.com/contenidos/1999/diciembre/acceso_a_prepublicaciones_sobre_economia_repec.html. ISSN 1386-6710.

13. **Davis, Jim, y otros, y otros.** Dienst Protocol Specification. Dienst Protocol Specification. [En línea] 30 de mayo de 2000. [Citado el: 24 de enero de 2013.]

http://www.cs.cornell.edu/cdlrg/dienst/protocols/dienstprotocol.htm#2.%20Dienst%20architecture%20overview.

14. **Blanco, Jaime .** Notas sobre Bibliotecas digitales. Caracas : s.n., 2007. ISSN 1316-6239.

15. **Guajardo Salinas, Aldo.** Z39.50 y OAI-PMH: Protocolos de Transferencia y Recuperación de Información. Chile : s.n., 2010.

16. **Siza Ramírez, Juan Pablo.** 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 contenido. BOGOTÁ : s.n., 2010.

17. **Barrueco, José Manuel y Subirats Coll, Imma.** OAI-PMH: Protocolo para la transmisión de contenidos en Internet.

18. **Open Archives Initiative.** Open Archives Initiative Protocol for Metadata Harvesting. Open Archives Initiative Protocol for Metadata Harvesting. [En línea] [Citado el: 25 de Enero de 2013.] http://www.openarchives.org.

19. **Carpenter, Leona.** Historia y desarrollo del OAI-PMH. Historia y desarrollo del OAI-PMH. [En línea] 14 de Octubre de 2003. [Citado el: 25 de Enero de 2013.] http://travesia.mcu.es/portalnb/jspui/html/10421/1823/page2.htm.

20. **Santillán Aldana, Julio.** Las iniciativas para el acceso abierto a la información científica en el contexto de la web semántica. La Habana : s.n., 2006. ISSN- 1562-4730.

21. **Blanco Suárez, Santiago .** Metadatos. 2006.

22. **Agudelo Benjumea, Mónica María.** Los metadatos. Los metadatos. [En línea] [Citado el: 25 de Enero de 2013.] http://aprendeenlinea.udea.edu.co/lms/men/docsoac3/0301_metadatos.pdf.

23. **Peis, Eduardo y Ruiz-Rodríguez, Antonio A.** EAD (Encoded Archival Description): Desarrollo, estructura, uso y aplicaciones. EAD (Encoded Archival Description): Desarrollo, estructura, uso y aplicaciones. [En línea] [Citado el: 26 de Enero de 2013.] http://www.upf.edu/hipertextnet/numero-2/ead.html#4.

24. **Comité de Metadatos de la Biblioteca Nacional de Chile.** Guía para la creación de metadatos usando dublin core. Santiagp : s.n., 2009.

25. **Oidilotk.** Odiloa3w. Odilotid. [En línea] [Citado el: 26 de Enero de 2013.] http://www.archivo3000.com/.

26. **Portal de Archivos Españoles.** Portal de Archivos Españoles. Portal de Archivos Españoles. [En línea] [Citado el: 20 de Enero de 2013.] http://pares.mcu.es/.

27. **Surós Vicente, Annia.** Gestor de documentos históricos. Ficha Comercial. La Habana : Centro de Informatización Universitaria, 2011.

28. **Verdecia Rodríguez, Frank Ernesto.** Documento de Arquitectura. Proyecto Archivo. Ciudad de La Habana : s.n., 2012.

29. **Rational Software Corporation.** Rational Unified Proccess. 2003.

30. **Larman, Craig.** UML y Patrones. Mexico : PREN. 1999.

31. **Vaquero, Miguel.** Web Docente Departamental. Los lenguajes de marcas. [En línea] [Citado el: 25 de Enero de 2013.] http://www.deciencias.net/disenoweb/elaborardw/paginas/intro_html.html.

32. **Sun Microsystms.** Inc. Javascript . [En línea] [Citado el: 25 de Enero de 2013.] http://www.javascript.com.

33. **SpringSource.** Groovy. Groovy. [En línea] [Citado el: 25 de Enero de 2013.] http://groovy.codehaus.org.

34. —. Grails . Grails. [En línea] [Citado el: 25 de Enero de 2013.] http://www.grails.org.

35. **Visual Paradigm International.** Visual Paradigm. Visual Paradigm. [En línea] [Citado el: 22 de enero de 2013.] http://www.visual-paradigm.com.

36. **SpringSource.** SpringSource. SpringSource. [En línea] [Citado el: 25 de enero de 2013.] http://www.springsource.com/developer/sts.

37. **Tortoise Svn Team.** Tortoise SVN. About Tortoise SVN. [En línea] [Citado el: 25 de enero de 2013.] http://tortoisesvn.net/about.html.

38. **Apache Software Foundation.** Apache Tomcat. Apache Tomcat. [En línea] [Citado el: 25 de 01 de 2013.] http://tomcat.apache.org/.

39. **PostgreSQL.** PostgreSQL. [En línea] The PostgreSQL Global Development Group , 2013. [Citado el: 14 de 1 de 2013.] http://www.postgresql.org.

40. **Oracle Corporation.** Oracle. Java 2 Plataform, Enterprise Edition (J2EE) Overview. [En línea] [Citado el: 25 de Enero de 2013.] http://java.sun.com/j2ee/overview.htm.

41. **Cochran, David.** Twitter Bootstrap Web Development. 2012. ISBN 978-1849518826.

42. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software . Madrid : s.n., 2000. ISBN 84-7829-036-2..

43. **Zapata, Carlos Mario, Palacio, Carolina y Olaya, Natalí.** Unc-analista: hacia la captura de un corpus de requisitos a partir de la aplicación del experimento mago de oz. Medellín : s.n., 2007. ISSN 1794-1237.

44. **Lagoze, Carl, y otros, y otros.** The Open Archives Initiative Protocol for Metadata Harvesting. Protocol Requests and Responses. [En línea] [Citado el: 20 de 03 de 2013.] http://www.openarchives.org/OAI/openarchivesprotocol.html.

45. **Otero Reyes, Daniurkys.** Módulo para la interconexión y búsqueda en proveedores de datos según el estándar OAI-PMH para el Sistema de Gestión de Documentos Históricos . Ciudad de La Habana : s.n., 2012.

46. **Escalona, María José y Koch, Nora.** Ingeniería de Requisitos en Aplicaciones para la Web – Un estudio comparativo. Sevilla : s.n., 2002.

47. **Cáceres Tello, Jesús .** Diagramas de Casos de Uso. Alcalá : s.n.

48. Sistemas expertos que recomiendan estrategias de instrucción. un modelo para su desarrollo. **Sierra, Enrique Ariel, Hossian, Alejandro y García-Martínez, Ramón.** Buenos Aires : s.n.

49. **Departamento de Base Datos (Universidad de las Ciencias Informáticas).** Entorno Virtual de Aprendizaje. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 01 de abril de 2013.] http://eva.uci.cu/.

50. **Brito Calahorro, Nacho.** Manual de desarrollo web con grails. JavaEE, como siempre debió haber sido. España : s.n., 2009. ISBN 978-84-613-2651.

51. **Guerra Sánchez, Esther.** Patrones de Diseño. Patrón de comportamiento Strategy. 2008.

52. **Zapata, Carlos Mario y Garcés, Gilma Liliana.** Generación del diagrama de secuencias de uml 2.1.1 desde esquemas preconceptuales. Medellín : s.n., 2008. ISSN 1794-1237.

53. **Vega Angulo, Maylin y Morales Reyes, Lianet.** Desarrollo del módulo Requisas de SIGDATI. La Habana : s.n., 2011.

54. **Vega Cutiño, Ruth .** Entorno Virtual de aprendizaje. Entorno Virtual de aprendizaje. [En línea] febrero de 2012. [Citado el: 09 de 05 de 2013.] http://eva.uci.cu/file.php/92/Tema_1_Introduccion/Materiales/Apoyo/Conceptos_basicos_v2.0.pdf.

55. **Machado Peña, Yadira y Quintero Rios, Mairelis.** Gestión de indicadores influyentes en la estimación de tiempo y esfuerzo en los proyectos productivos de la Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2008.

56. **Pressman, Roger.** Ingeniería del software. Un enfoque práctico.

57. **Cibertec.** Pruebas de Software.

58. **O'Reilly Media Inc.** What is XML? O'Reilly. [En línea] [Citado el: 26 de Enero de 2013.] http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN58.

59. **MSDN.** MagazinePatterns in Practice. MagazinePatterns in Practice. [En línea] [Citado el: 25 de Enero de 2013.] http://msdn.microsoft.com/en-us/magazine/dd419655.aspx.

60. **Sarl Clevacti.** Techno Science. Don't repeat yourself. [En línea] [Citado el: 25 de enero de 2013.] http://www.techno-science.net/?onglet=glossaire&definition=11305.

61. —. Techno Science. [En línea] [Citado el: 25 de Enero de 2013.] http://www.techno-science.net/?onglet=glossaire&definition=609.

62. —. Techno-Science. Techno-Science. [En línea] [Citado el: 25 de Enero de 2013.] http://www.techno-science.net/?onglet=glossaire&definition=7737.

63. **Herrera, Antonia Heredia.** Archivística General. Teoría y Práctica 5ta Edición. Sevilla : s.n., 1991.

64. **Abraham Silberschatz, Henry F. y Korth, S. Sudarshan.** Database System Concepts.

65. **jQuery.** What is jQuery? jQuery. [En línea] [Citado el: 25 de Enero de 2013.] http://jquery.com/.

66. **Larman, Craig.** Uml y patrones. Introducción al anáalisis y diseño orientado a objetos. Mexico : s.n., 1999. ISBN: 970-17-0261-1.

Anexos

CU CRUD_Proveedor de Servicio

Objetivo	Registrar, modificar, eliminar o listar un proveedor de servicio.
Actores	Administrador del sistema.
Resumen	El Administrador del sistema selecciona realizar alguna acción sobre el Proveedor de Servicio. El sistema permite registra, modificar, eliminar o ver los detalles del Proveedor de Servicio.
Complejidad	Baja
Prioridad	Media
Precondiciones	• Para modificar, eliminar o ver detalles de Proveedor de Servicio el mismo debe estar seleccionado.
Postcondiciones	• Proveedor de Servicio registrado. • Proveedor de Servicio actualizado. • Proveedor de Servicio eliminado. • Datos de Proveedor de Servicio mostrado.

Flujo de eventos		
Flujo Básico Registrar Proveedor de Servicio		
	Actor	**Sistema**
1.	Selecciona la opción Gestionar Proveedores de Servicios.	
2.		Permite realizar las siguientes acciones: • Registrar Proveedor de Servicio. • Modificar Proveedor de Servicio. Ver Sección 1: “Modificar Proveedor de Servicio”. • Eliminar Proveedor de Servicio. Ver Sección 2: “Eliminar Proveedor de Servicio”.
3.		Muestra un formulario solicitando la información necesaria para registrar un Proveedor de Servicio: • Nombre • Teléfono • Correo electrónico • Estado • Dirección IP Las opciones Aceptar y Cancelar.
4.	Introduce los datos solicitados.	
5.	Selecciona la opción “Aceptar”.	
6.		Comprueba que se han introducido

		los datos obligatorios.
7.		Comprueba que los datos introducidos son correctos.
8.		Comprueba que no exista un Proveedor de Servicio registrado con datos especificados anteriormente.
9.		Almacena los datos introducidos y muestra un mensaje que confirma el éxito de la operación.
10.		Se muestra en una tabla en la parte inferior del formulario los detalles del Proveedor de Servicios registrado.
11.		Termina el caso de uso.
Flujos alternos		
6a. Datos obligatorios incompletos		
	Actor	**Sistema**
6a.1		Muestra un mensaje de error indicando que no se han introducido los datos obligatorios y señala los que deben incluirse.
6a.2		Ir a la acción 4.
7a. Datos Incorrectos		
	Actor	**Sistema**
7a.1		Muestra un mensaje de error indicando que se han introducido datos incorrectos y señala los que deben corregirse sin eliminar la información incorrecta.
7a.2		Ir a la acción 4.
8a. Proveedor de Servicio Registrado Anteriormente		
	Actor	**Sistema**
8a.1		Muestra un mensaje indicando que ya existe un Proveedor de Servicio registrado con esos datos.
8a.2		Ir a la acción 4.
Sección 1: “Modificar Proveedor de Servicio”		
Flujo básico Modificar Proveedor de Servicio		
	Actor	**Sistema**
1.	Selecciona la opción “Modificar Proveedor de Servicio”.	
2.		Obtiene los datos del Proveedor de Servicio seleccionado.
3.		Muestra un formulario con todos los datos del Proveedor de Servicio de forma editable. Los datos del Proveedor de Servicio son: • Nombre • Estado

		• Teléfono • Correo electrónico • Dirección IP Muestra las siguientes opciones: • Aceptar. • Cancelar.
4.	Modifica los datos del Proveedor de Servicio.	
5.	Selecciona la opción “Aceptar”.	
6.		Comprueba que se han introducido los datos obligatorios.
7.		Comprueba que los datos introducidos son correctos.
8.		Comprueba que no exista un Proveedor de Servicio registrado con datos especificados anteriormente.
9.		Actualiza los datos introducidos y muestra un mensaje que confirma el éxito de la operación.
10.		Muestra en la vista el Proveedor de Servicios modificado.
11.		Termina el caso de uso.
Flujos alternos		
a. Opción “Cancelar”		
	Actor	**Sistema**
a.1	Selecciona la opción “Cancelar”.	
a.2		Retorna a la página que le dio origen.
6a. Datos obligatorios incompletos		
	Actor	**Sistema**
6a. 1		Muestra un mensaje de error indicando que no se han introducido los datos obligatorios y señala los que deben incluirse.
6a. 2		Ir a la acción 3.
7a. Datos Incorrectos		
	Actor	**Sistema**
7a. 1		Muestra un mensaje de error indicando que se han introducido datos incorrectos y señala los que deben corregirse sin eliminar la información incorrecta.
7a. 2		Ir a la acción 3.
8a. Proveedor de Servicio Registrado Anteriormente		
	Actor	**Sistema**
8a. 1		Muestra un mensaje indicando que ya existe un Proveedor de Servicio registrado con esos datos.
8a. 2		Ir a la acción 4.
Sección 2: “Eliminar Proveedor de Servicio”		

Flujo básico Eliminar Proveedor de Servicio		
	Actor	**Sistema**
1.	Selecciona la opción “Eliminar” Proveedor de Servicio.	
2.		Muestra un mensaje de confirmación de eliminar el Proveedor de Servicio. Muestra las siguientes opciones: • Sí. • No.
3.	Selecciona la opción “Si”.	
4.		Elimina Proveedor de Servicio seleccionado y muestra un mensaje confirmando el éxito de la operación.
5.		Muestra un mensaje confirmando el éxito de la operación.
6.		Retorna a la página anterior actualizando la eliminación del usuario externo.
7.		Termina el caso de uso.
Flujos alternos		
a Opción “No”.		
	Actor	**Sistema**
a.1	Selecciona la opción “No”.	
a.2		Retorna a la página que le dio origen.

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

Ilustración 1: Especificación del CU CRUD_Proveedor de Servicio.

CU Generar Reporte

Objetivo	Consultar información de las peticiones y la cantidad de metadatos que se ha recolectado.
Actores	Administrador
Resumen	El Administrador del sistema selecciona el tipo de reporte que desea consultar. El sistema muestra los datos de acuerdo al tipo de reporte seleccionado.
Complejidad	Baja
Prioridad	Media
Precondiciones	Para mostrar un determinado reporte el mismo debe ser primero seleccionado.
Postcondiciones	
Flujo de eventos	
Flujo básico Generar Reporte	

	Actor	Sistema
1.		Permite generar diferentes reportes: • Cantidad de peticiones recibidas atendiendo al verbo. Ver Sección 1: “Cantidad de peticiones recibidas atendiendo al verbo”. • Cantidad de metadatos que ha recolectado cada proveedor de servicios. Ver Sección 2 “Cantidad de metadatos que ha recolectado cada proveedor de servicios”.
2.		Termina el caso de uso
Sección 1: “Cantidad de peticiones recibidas atendiendo al verbo”		
Cantidad de peticiones recibidas atendiendo al verbo		
	Actor	**Sistema**
1.	Selecciona la opción “Cantidad de peticiones recibidas atendiendo al verbo”.	
2.		Muestra los datos del reporte. Los datos son: • Verbos. • Cantidad de peticiones. El tipo de documento: PDF El encabezado: Título del reporte, Logo y nombre del sistema, Nombre y apellidos del funcionario autenticado, Fecha de creación y Fecha actual.
3.		Termina el caso de uso.
Sección 2: “Cantidad de metadatos que ha recolectado cada proveedor de servicios”		
Cantidad de metadatos que ha recolectado cada proveedor de servicios		
	Actor	**Sistema**
1.	Selecciona la opción “Cantidad de metadatos que ha recolectado cada proveedor de servicios”.	
2.		Muestra los datos del reporte. Los datos son: • Proveedor de Servicios. • Cantidad de metadatos recolectados. El tipo de documento: PDF El encabezado: Título del reporte, Logo y nombre del sistema, Nombre y apellidos del funcionario autenticado, Fecha de creación y Fecha actual.
3.		Termina el caso de uso.
Relaciones	**CU Incluidos**	

	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes	No aplica.	

Ilustración 2: Especificación del CU Generar Reporte.

CU CRUD_Proveedor de Datos

Objetivo	Registrar, modificar, eliminar o listar un Proveedor de Datos.
Actores	Administrador del sistema.
Resumen	El Administrador del sistema selecciona realizar alguna acción sobre el proveedor de servicio. El sistema permite registra, modificar, eliminar o ver los detalles del Proveedor de Datos.
Complejidad	Media
Prioridad	Media
Precondiciones	• Para modificar, eliminar o ver detalles del Proveedor de Datos debe estar seleccionado.
Postcondiciones	• Proveedor de Datos registrado. • Proveedor de Datos actualizado. • Proveedor de Datos eliminado. • Datos del Proveedor de Datos mostrado.
Flujo de eventos	
Flujo Básico Registrar Proveedor de Datos	

	Actor	**Sistema**
1.	Selecciona la opción Gestionar Proveedores de Datos.	
2.		Permite realizar las siguientes acciones: • Registrar Proveedor de Datos. • Modificar Proveedor de Dato. Ver Sección 1: “Modificar Proveedor de Datos”. • Eliminar Proveedor de Datos. Ver Sección 2: “Eliminar Proveedor de Datos”. • Describir Proveedor de Datos. Ver Sección 3:”Describir proveedor de datos”. • Editar descripción del Proveedor de Datos. Ver Sección 4 “Editar descripción del proveedor de datos”. • Ver detalles de la descripción. Ver Sección 5 “Detalles de la descripción del proveedor de

		datos".
3.		Muestra un formulario solicitando la información necesaria para insertar un Proveedor de Datos: La información necesaria es: • Dirección URL. La opción Obtener formatos disponibles.
4.	Introduce el dato solicitado.	
5.	Selecciona la opción "Obtener formatos disponibles".	
6.		Comprueba que se ha introducido el dato obligatorio.
7.		Comprueba que el dato introducido es correcto.
8.		Captura la URL del Proveedor de Datos y le muestra al usuario la información correspondiente con el mismo.
9.	Selecciona el formato que desea.	
10.	Selecciona la opción "Aceptar".	
11.		Comprueba que se ha seleccionado el formato que desea.
12.		Almacena el dato introducido y muestra un mensaje que confirma el éxito de la operación.
13.		Muestra en la vista el proveedor de datos registrado.
14.		Termina el caso de uso.
Flujos alternos		
6a. Datos obligatorios incompletos		
	Actor	**Sistema**
6a.1		Muestra un mensaje de error indicando que no se ha introducido el dato obligatorio y señala el que debe incluirse.
6a.2		Ir a la acción 4.
7a. Datos Incorrectos		
	Actor	**Sistema**
7a.1		Muestra un mensaje de error indicando que se ha introducido mal la URL y señala que debe corregirse sin eliminar la información incorrecta.
7a.2		Ir a la acción 4.
11a. Formato seleccionado		
	Actor	**Sistema**
11a.1		Muestra un mensaje indicando que debe seleccionar el formato del metadato.

11a.2		Ir a la acción 9.
Sección 1: "Modificar Proveedor de Datos"		
Flujo básico Modificar Proveedor de Datos		
	Actor	**Sistema**
1.	Selecciona la opción "Modificar" el Proveedor de Datos.	
2.		Obtiene el dato del Proveedor de Datos seleccionado.
3.		Muestra un formulario con el dato del Proveedor de Datos de forma editable. El dato del Proveedor de Datos es: • Dirección URL Muestra las siguientes opciones: • Aceptar. • Cancelar. La opción Obtener formatos disponibles.
4.	Modifica el dato del Proveedor de Datos.	
5.	Selecciona la opción "Aceptar".	
6.		Comprueba que se ha seleccionado el formato del metadato.
7.		Actualiza el dato introducido y muestra un mensaje que confirma el éxito de la operación.
8.		Muestra en la vista el Proveedor de Datos modificado.
9.		Termina el caso de uso.
Flujos alternos		
a. Opción "Cancelar"		
	Actor	**Sistema**
a.1	Selecciona la opción "Cancelar".	
a.2		Retorna a la página que le dio origen.
6a Formato seleccionado		
	Actor	**Sistema**
6a.1		Muestra un mensaje indicando que debe seleccionar el formato del metadato.
6a.2		
Sección 2: "Eliminar Proveedor de Datos"		
Flujo básico Eliminar Proveedor de Datos		
	Actor	**Sistema**
1.	Selecciona la opción "Eliminar" Proveedor de Datos.	
2.		Muestra un mensaje de confirmación de eliminar el Proveedor de Datos. Muestra las siguientes opciones: • Sí. • No.

3.	Selecciona la opción “Si”.	
4.		Elimina el Proveedor de Datos seleccionado.
5.		Muestra un mensaje confirmando el éxito de la operación.
6.		Retorna a la página anterior actualizando la eliminación del Proveedor de Datos.
7.		Termina el caso de uso.
Flujos alternos		
a Opción “No”.		
	Actor	**Sistema**
a.1	Selecciona la opción “No”.	
a.2		Retorna a la página que le dio origen.
Sección 3:”Describir proveedor de datos”.		
Flujo básico Describir proveedor de datos		
	Actor	**Sistema**
1		Ver **CU Describir Archivo Histórico.**
Sección 4: “Editar descripción del proveedor de datos”		
Flujo básico Editar descripción del proveedor de datos		
	Actor	**Sistema**
1		Ver **CU Describir Archivo Histórico Sección 1 “Modificar Descripción de Archivo”.**
Sección 5: “Detalles de la descripción del proveedor de datos”		
Flujo básico Detalles de la descripción del proveedor de datos		
	Actor	**Sistema**
1		Ver **CU Describir Archivo Histórico Sección 2 “Ver Descripción del Archivo”.**

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

Ilustración 3: Especificación del CU CRUD_Proveedor de Datos

CRUD-D_ Describir Archivo Histórico

Objetivo	Registrar, modificar o ver la descripción del Archivo Histórico.
Actores	Administrador del sistema: (Inicia) Registra, modifica o ve la descripción del Archivo.
Resumen	El Administrador del sistema selecciona realizar alguna acción sobre la descripción del Archivo. El sistema permite describir, modificar o ver la descripción del archivo.
Complejidad	Baja

Prioridad	Media
Precondiciones	
Postcondiciones	• Archivo registrado. • Archivo actualizado. • Datos del Archivo mostrado.

Flujo de eventos		
Flujo Básico Describir Archivo Histórico		
	Actor	**Sistema**
1.	Selecciona la opción “Describir Archivo Histórico”.	
2.		Permite realizar las siguientes acciones: • Describir Archivo Histórico. • Modificar Descripción de Archivo. Ver Sección 1: “Modificar Descripción de Archivo”. • Ver Descripción del Archivo. Ver Sección 2: “Ver Descripción del Archivo”.
3.		Muestra un formulario solicitando la información necesaria para describir el Área de Identificación. La información necesaria es: • Identificador. • Forma autorizada del nombre. • Forma paralela del nombre. • Otra forma del nombre. • Tipo de institución. Muestra las siguientes opciones: • Siguiente.
4.	Introduce los datos solicitados.	
5.	Selecciona la opción “Siguiente”.	
6.		Comprueba que se han introducido los datos obligatorios.
7.		Comprueba que los datos introducidos son correctos.
8.		Almacena los datos introducidos.
9.		Muestra un formulario solicitando la información necesaria para describir el Área de Contacto. La información necesaria es: • URL • Dirección. • Teléfono • Fax • Correo electrónico. • Nombre Completo. • Cargo

		• Correo Electrónico • Dirección Muestra las siguientes opciones: • Siguiente. • Anterior.
10.	Introduce los datos solicitados.	
11.	Selecciona la opción “Siguiente”.	
12.		Comprueba que se han introducido los datos obligatorios.
13.		Comprueba que los datos introducidos son correctos
14.		Almacena los datos introducidos.
15.		Muestra un formulario solicitando la información necesaria para describir el Área de Descripción. La información necesaria es: • Historia de la institución. • Contexto cultural y geográfico. • Fuentes legales. • Estructura administrativa. • Gestión de documentos y política de ingresos. • Edificio. • Fondos y otras colecciones custodiadas. • Instrumentos de descripción, guías y publicaciones. Muestra las siguientes opciones: • Siguiente. • Anterior.
16.	Introduce los datos solicitados.	
17.	Selecciona la opción “Siguiente”.	
18.		Comprueba que se han introducido los datos obligatorios.
19.		Comprueba que los datos introducidos son correctos
20.		Almacena los datos introducidos.
21.		Muestra un formulario solicitando la información necesaria para describir el Área de Acceso. La información necesaria es: • Horario. • Condiciones y requisitos para el uso y el acceso. • Accesibilidad. Muestra las siguientes opciones: • Siguiente.

		• Anterior.
22.	Introduce los datos solicitados.	
23.	Selecciona la opción “Siguiente”.	
24.		Comprueba que se han introducido los datos obligatorios.
25.		Comprueba que los datos introducidos son correctos
26.		Almacena los datos introducidos.
27.		Muestra un formulario solicitando la información necesaria para describir el Área de Servicios. La información necesaria es: • Servicios de ayuda a la investigación. • Servicios de reproducción. • Espacios públicos. Muestra las siguientes opciones: • Siguiente. • Anterior.
28.	Introduce los datos solicitados.	
29.	Selecciona la opción “Siguiente”.	
30.		Comprueba que se han introducido los datos obligatorios.
31.		Comprueba que los datos introducidos son correctos.
32.		Almacena los datos introducidos.
33.		Muestra un formulario solicitando la información necesaria para describir el Área de Control. La información necesaria es: • Identificador. • Identificador de la institución. • Reglas. • Estado de elaboración. • Nivel de detalle. • Fechas de creación • Última revisión • Lengua o escritura. • Fuentes. • Notas de mantenimiento. Muestra las siguientes opciones: • Terminar. • Anterior.
34.	Introduce los datos solicitados.	
35.	Selecciona la opción “Aceptar”.	
36.		Comprueba que se han introducido los datos obligatorios.
37.		Comprueba que los datos introducidos son correctos

38.		Almacena los datos introducidos.
39.		Muestra la vista “Ver Descripción del Archivo” con un mensaje confirmando el éxito de la operación.
40.		Termina el caso de uso.
Flujos alternos		
6, 12, 18, 24, 30, 36 a. Datos obligatorios incompletos		
	Actor	**Sistema**
a.1		Muestra un mensaje de error indicando que no se han introducido los datos obligatorios y señala los que deben incluirse.
a.2		Ir a la acción donde se muestran los formularios para introducir los datos.
7, 13, 19, 25, 31, 37 a. Datos Incorrectos		
	Actor	**Sistema**
a.2		Muestra un mensaje de error indicando que se han introducido datos incorrectos y señala los que deben corregirse sin eliminar la información incorrecta.
a.3		Ir a la acción donde se introducen los datos.
a. Opción “Cancelar”		
a.1	Seleccionar la opción Cancelar.	
a.2		Retorna a la página que le dio origen.
a. Opción “Anterior”		
a.1	Selecciona la opción “Anterior”	
a.2		Regresa al formulario con los datos del área anterior a la que se encuentra actualmente.
Sección 1: “Modificar Descripción de Archivo”		
Flujo básico Modificar Descripción de Archivo		
	Actor	**Sistema**
1.	Selecciona la opción “Modificar” Descripción de Archivo.	
2.		Obtiene los datos de la Descripción del Archivo Histórico.
3.		Muestra un formulario solicitando la información necesaria para describir el Área de Identificación. La información necesaria es: • Identificador. • Forma autorizada del nombre. • Forma paralela del nombre. • Otra forma del nombre. • Tipo de institución. Muestra las siguientes opciones: • Siguiente.

4.	Modifica los datos de la descripción.	
5.	Selecciona la opción “Siguiente”.	
6.		Comprueba que se han introducido los datos obligatorios.
7.		Comprueba que los datos introducidos son correctos
8.		Actualiza los datos introducidos.
9.		Muestra un formulario solicitando la información necesaria para describir el Área de Contacto. La información necesaria es: • URL • Dirección. • Teléfono • Fax • Correo electrónico. • Nombre Completo. • Cargo • Correo Electrónico • Dirección Muestra las siguientes opciones: • Siguiente. • Anterior.
10.	Modifica los datos de la descripción.	
11.	Selecciona la opción “Siguiente”.	
12.		Comprueba que se han introducido los datos obligatorios.
13.		Comprueba que los datos introducidos son correctos
14.		Actualiza los datos introducidos.
15.		Muestra un formulario solicitando la información necesaria para describir el Área de Descripción. La información necesaria es: • Historia de la institución. • Contexto cultural y geográfico. • Fuentes legales. • Estructura administrativa. • Gestión de documentos y política de ingresos. • Edificio. • Fondos y otras colecciones custodiadas. • Instrumentos de descripción, guías y publicaciones. Muestra las siguientes opciones: • Siguiente. • Anterior.

16.	Modifica los datos de la descripción.	
17.	Selecciona la opción “Siguiente”.	
18.		Comprueba que se han introducido los datos obligatorios.
19.		Comprueba que los datos introducidos son correctos
20.		Actualiza los datos introducidos.
21.		Muestra un formulario solicitando la información necesaria para describir el Área de Acceso. La información necesaria es: • Horario. • Condiciones y requisitos para el uso y el acceso. • Accesibilidad. Muestra las siguientes opciones: • Siguiente. • Anterior.
22.	Modifica los datos de la descripción.	
23.	Selecciona la opción “Siguiente”.	
24.		Comprueba que se han introducido los datos obligatorios.
25.		Comprueba que los datos introducidos son correctos
26.		Actualiza los datos introducidos.
27.		Muestra un formulario solicitando la información necesaria para describir el Área de Servicios. La información necesaria es: • Servicios de ayuda a la investigación. • Servicios de reproducción. • Espacios públicos. Muestra las siguientes opciones: • Siguiente. • Anterior.
28.	Modifica los datos de la descripción.	
29.	Selecciona la opción “Siguiente”.	
30.		Comprueba que se han introducido los datos obligatorios.
31.		Comprueba que los datos introducidos son correctos
32.		Actualiza los datos introducidos.
33.		Muestra un formulario solicitando la información necesaria para describir el Área de Control. La información necesaria es: • Identificador. • Identificador de la institución.

		• Reglas. • Estado de elaboración. • Nivel de detalle. • Fechas de creación • Última revisión • Lengua o escritura. • Fuentes. • Notas de mantenimiento. Muestra las siguientes opciones: • Terminar. • Anterior.
34.	Modifica los datos de la descripción.	
35.	Selecciona la opción “Finalizar”.	
36.		Comprueba que se han introducido los datos obligatorios.
37.		Comprueba que los datos introducidos son correctos
38.		Actualiza los datos introducidos.
39.		Muestra la vista “Ver Descripción del Archivo” con un mensaje confirmando el éxito de la operación.
40.		Termina el caso de uso.
Flujos alternos		
a. Opción “Cancelar”		
	Actor	**Sistema**
a.1	Selecciona la opción “Cancelar”.	
a.2		Retorna a la página que le dio origen.
6, 12, 18, 24, 30, 36a. Datos obligatorios incompletos		
	Actor	**Sistema**
a. 1		Muestra un mensaje de error indicando que no se han introducido los datos obligatorios y señala los que deben incluirse.
a. 2		Ir a la acción donde se modifican los datos.
7, 13, 19, 25, 31, 37a. Datos Incorrectos		
	Actor	**Sistema**
a. 1		Muestra un mensaje de error indicando que se han introducido datos incorrectos y señala los que deben corregirse sin eliminar la información incorrecta.
a. 2		Ir a la acción donde se modifican los datos.
a. Opción “Anterior”		
	Selecciona la opción “Anterior”	
		Regresa al formulario con los datos

		del área anterior a la que se encuentra actualmente.
Sección 3 “Ver Descripción del Archivo”		
Flujo básico Ver Descripción del Archivo		
	Actor	**Sistema**
1.	Selecciona la opción “Ver Descripción” del Archivo.	
2.		Obtiene los datos de la Descripción del Archivo.
3.		Muestra los datos de la Descripción del Archivo organizados por las áreas correspondientes. Los datos de la descripción son: Área de Identificación • Identificador. • Forma autorizada del nombre. • Forma paralela del nombre. • Otra forma del nombre. • Tipo de institución que conserva los fondos de archivo. Área de Contacto. • Localización y dirección. • Teléfono, fax, correo electrónico. • Personas de contacto. Área de Descripción. • Historia de la institución que custodia los fondos de archivo. • Contexto cultural y geográfico. • Atribuciones/Fuentes legales. • Estructura administrativa. • Gestión de documentos y política de ingresos. • Edificio. • Fondos y otras colecciones custodiadas. • Instrumentos de descripción, guías y publicaciones. Área de Acceso. • Horarios de apertura. • Condiciones y requisitos para el uso y el acceso. • Accesibilidad. Área de Servicios. • Servicios de ayuda a la investigación.

		• Servicios de reproducción. • Espacios públicos. Área de Control. • Identificador de la descripción. • Identificador de la institución. • Reglas y/o convenciones. • Estado de elaboración. • Nivel de detalle. • Fechas de creación, revisión o eliminación. • Lengua(s) y escritura(s). • Fuentes. • Notas de mantenimiento.
4.		Muestra la opción que permite regresar a la página anterior.
5.		Termina el caso de uso.
Flujos alternos		
4a. Opción Anterior		
	Actor	**Sistema**
4a.1	Selecciona la opción “Anterior”.	
		Cierra la página con los datos del Archivo.
4a.2		Retorna a la página que le dio origen.

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

Ilustración 4: Especificación del CU CRUD-D_ Describir Archivo Histórico.

CU Visualizar descripción de documentos de Archivo

Objetivo	Consultar la descripción completa del documento encontrado.
Actores	Usuario
Resumen	Se muestra la descripción de los documentos que fueron encontrados en la búsqueda realizada por el usuario.
Complejidad	Baja
Prioridad	Media
Precondiciones	
Postcondiciones	
Flujo de eventos	
Flujo básico Aprobar Descripción	

	Actor	**Sistema**

1.	Selecciona la opción Visualizar Descripción de Documentos de Archivo del resultado de la búsqueda obtenida.		
2.			Busca la descripción completa del documento seleccionado.
3.			Muestra la descripción obtenida.
4.	Selecciona la opción “Aceptar”.		
			Termina el caso de uso.
Relaciones		**CU Incluidos**	
		CU Extendidos	
Requisitos no funcionales			
Asuntos pendientes			

Ilustración 5: CU Visualizar descripción de documentos de Archivo

CU Codificar Respuesta

Objetivo	Codificar el resultado obtenido por el Recolectar metadatos en formato XML.		
Actores	Recolectar metadatos		
Resumen	Los objetos recibidos se codifican en formato XML de acuerdo a la norma especificada		
Complejidad	Media		
Prioridad	Alta		
Precondiciones			
Postcondiciones			
Flujo de eventos			
Flujo básico Codificar Respuesta			
	Actor		**Sistema**
1.	Envía los objetos a codificar y la norma de codificación.		
2.			Si la norma de codificación es EAD se convierten los datos a dicha norma.
3.			Si la norma de codificación es Dublin Core se convierten los datos a dicha norma.
4.			Termina el caso de uso.
Relaciones		**CU Incluidos**	
		CU Extendidos	
Requisitos no funcionales			
Asuntos pendientes	No aplica.		

Ilustración 6: CU Codificar Respuesta

CU Decodificar Respuesta

Objetivo	Decodificar la respuesta obtenida por la búsqueda de documentos de archivo.	
Actores	Buscar en proveedores de datos.	
Resumen	Los objetos recibidos se decodifican a la norma ISAD (G).	
Complejidad	Media	
Prioridad	Alta	
Precondiciones		
Postcondiciones		
Flujo de eventos		
Flujo básico Decodificar Respuesta		
	Actor	**Sistema**
1.	Envía los objetos a decodificar.	
2.		Decodifica el XML en objetos que el sistema entienda (DRecord).
3.		Termina el caso de uso
4.		
Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes	No aplica.	

Ilustración 7: CU Decodificar respuesta.

CU Mostrar descripción del Archivo

Objetivo	Mostrar información detallada por diferentes áreas del Archivo.	
Actores	Usuario	
Resumen	El usuario selecciona la opción para ver la descripción del archivo y se le muestra la información correspondiente.	
Complejidad	Baja	
Prioridad	Media	
Precondiciones		
Postcondiciones		
Flujo de eventos		
Flujo básico Mostrar descripción		
	Actor	**Sistema**
1.	Selecciona la opción “Ver Descripción” del Archivo.	
2.		Obtiene los datos de la Descripción del Archivo.
3.		Muestra los datos de la Descripción

		del Archivo organizados por las áreas correspondientes. Los datos de la descripción son: • Forma autorizada del nombre • Localización y dirección. • Teléfono, fax, correo electrónico. • Horarios de apertura • Nivel de detalle. • Servicios de ayuda a la investigación. • Condiciones y requisitos para el uso y el acceso.
4.		Muestra la opción que permite regresar a la página anterior.
5.		Termina el caso de uso.

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes	No aplica.	

Ilustración 8: CU Mostrar descripción del archivo.

CU Describir Servidor OAI-PMH

Objetivo	Registrar un servidor OAI-PMH para cumplir con la petición del verbo Identify.
Actores	Administrador del sistema.
Resumen	El Administrador del sistema selecciona describir el servidor OAI-PMH.
Complejidad	Media
Prioridad	Media
Precondiciones	
Postcondiciones	
Flujo de eventos	
Flujo Básico Describir Servidor OAI-PMH	

	Actor	**Sistema**
1.	Selecciona la opción Describir servidor OAI-PMH.	
2.		Muestra un formulario solicitando la información necesaria para describir un servidor de archivo: La información necesaria es: • Nombre del Repositorio.

		• URL base. • Versión del protocolo. • Primera fecha. • Registros de eliminación. • Granularidad. Muestra la opción: • Aceptar • Cancelar
3.	Introduce los datos solicitados.	
4.	Selecciona la opción Aceptar.	
5.		Comprueba que se han introducido los datos obligatorios.
6.		Comprueba que los datos introducidos son correctos.
7.		Almacena los datos introducidos y muestra un mensaje que confirma el éxito de la operación.
8.		Muestra una vista con los detalles de los datos registrados.
9.		Termina el caso de uso.
Flujos alternos		
5a. Datos obligatorios incompletos		
	Actor	**Sistema**
5a.1		Muestra un mensaje de error indicando que no se han introducido los datos obligatorios y señala los que deben incluirse.
5a.2		Ir a la acción 3.
6a. Datos Incorrectos		
	Actor	**Sistema**
6a.1		Muestra un mensaje de error indicando que se han introducido incorrectamente los datos de la descripción.
6a.2		Ir a la acción 3.

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

Ilustración 9: Describir Servidor OAI-PMH

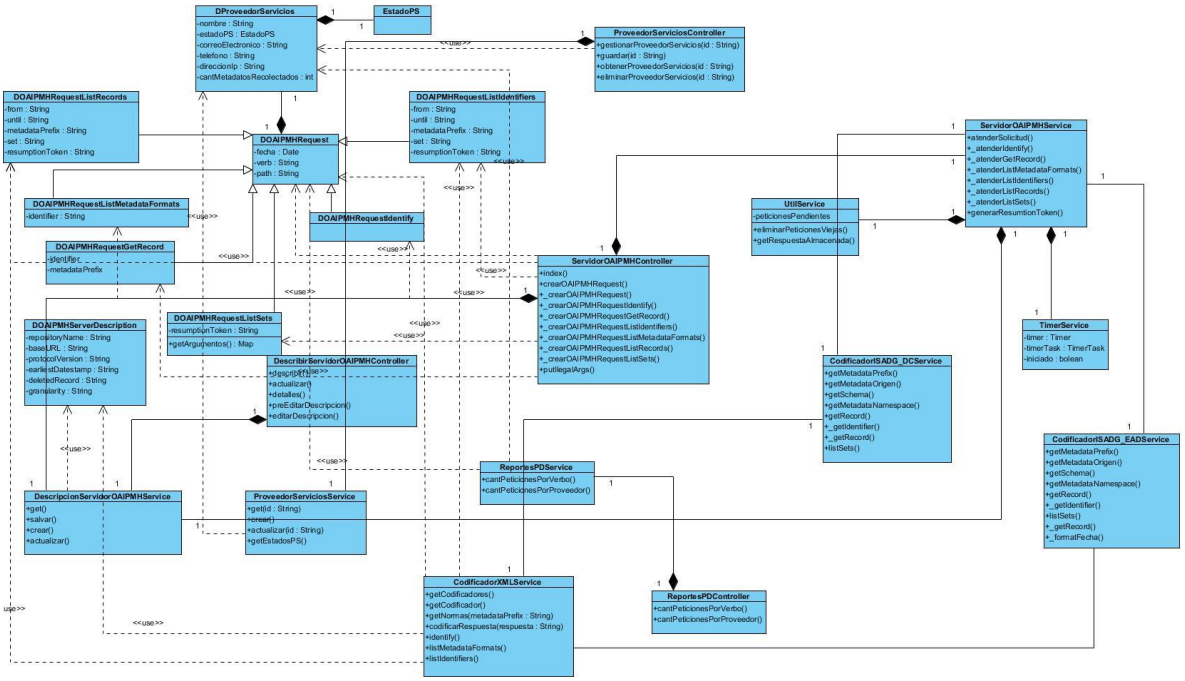

Ilustración 10: Diagrama de clases perteneciente a los PD.

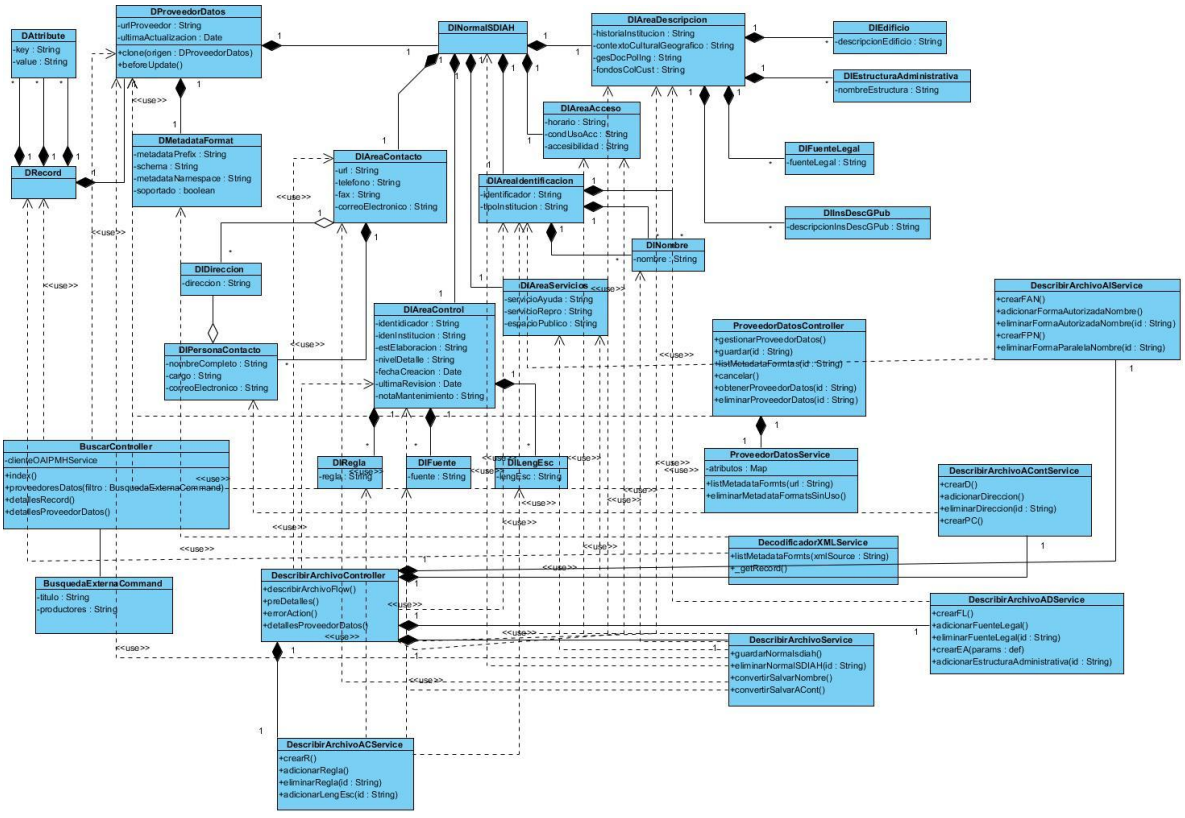

Ilustración 11: Diagrama de clases perteneciente al PS.

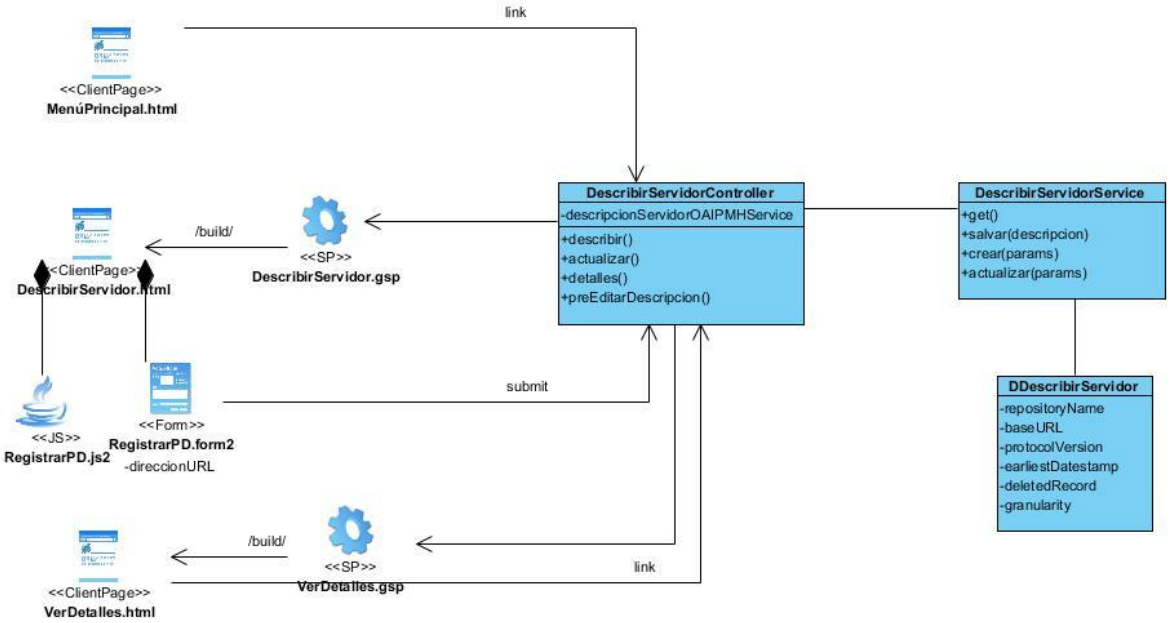

Ilustración 12: Diagrama de clases del diseño del CU Describir servidor OAI-PMH.

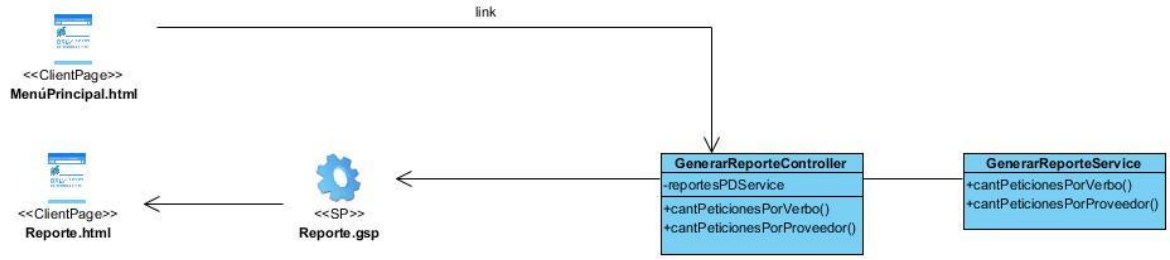

Ilustración 13: Diagrama de clases del diseño del CU Generar reporte.

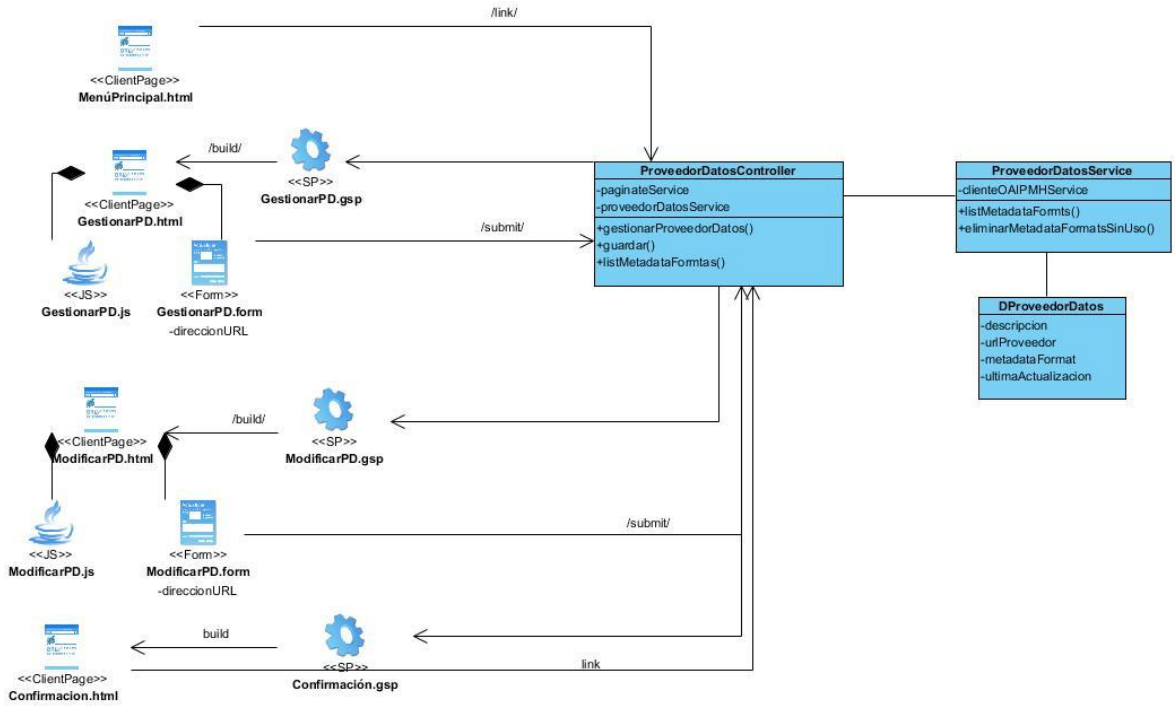

Ilustración 14: Diagrama de clases del diseño del CU Gestionar PD.

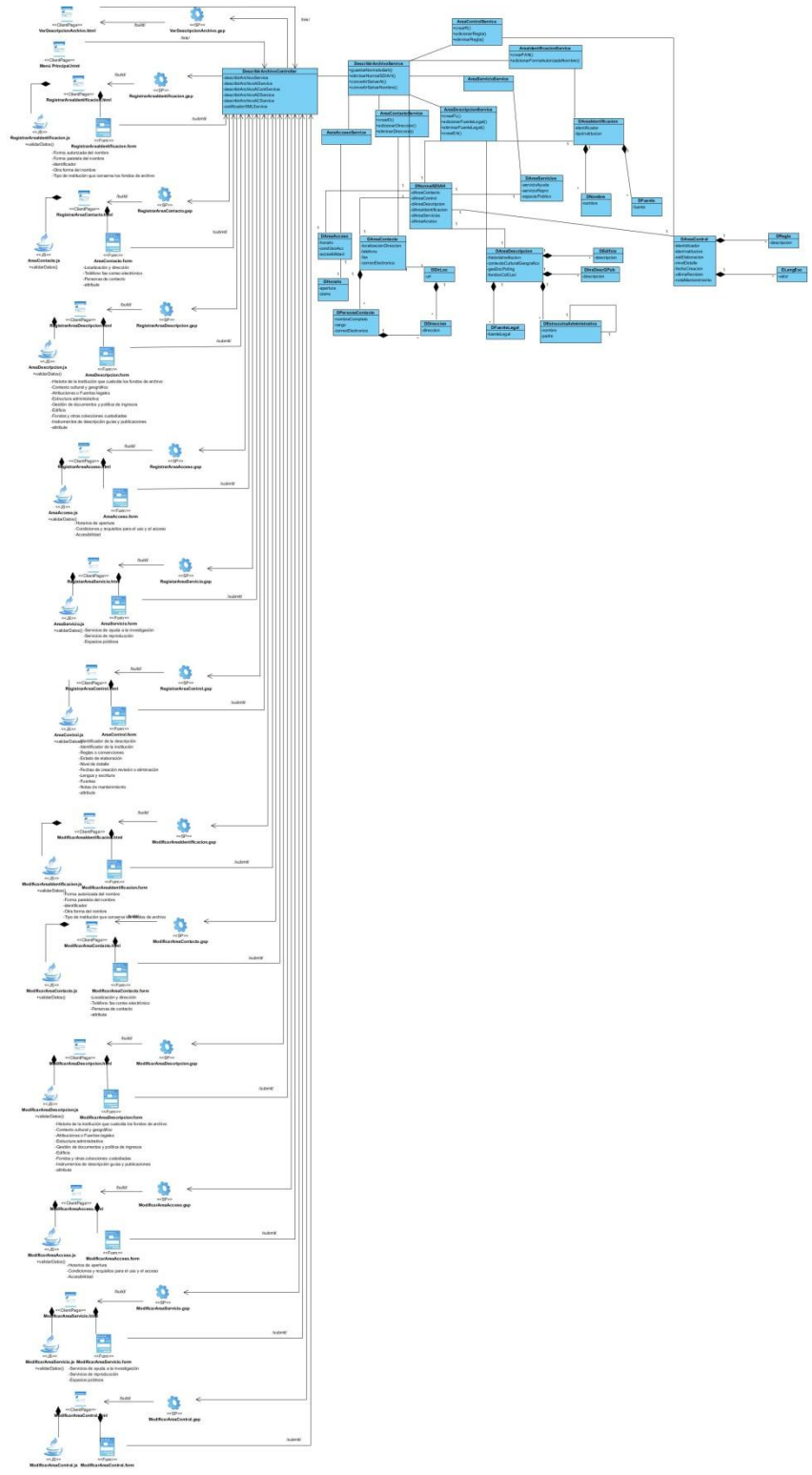

Ilustración 15: Diagrama de clases del diseño del CU Describir Archivo

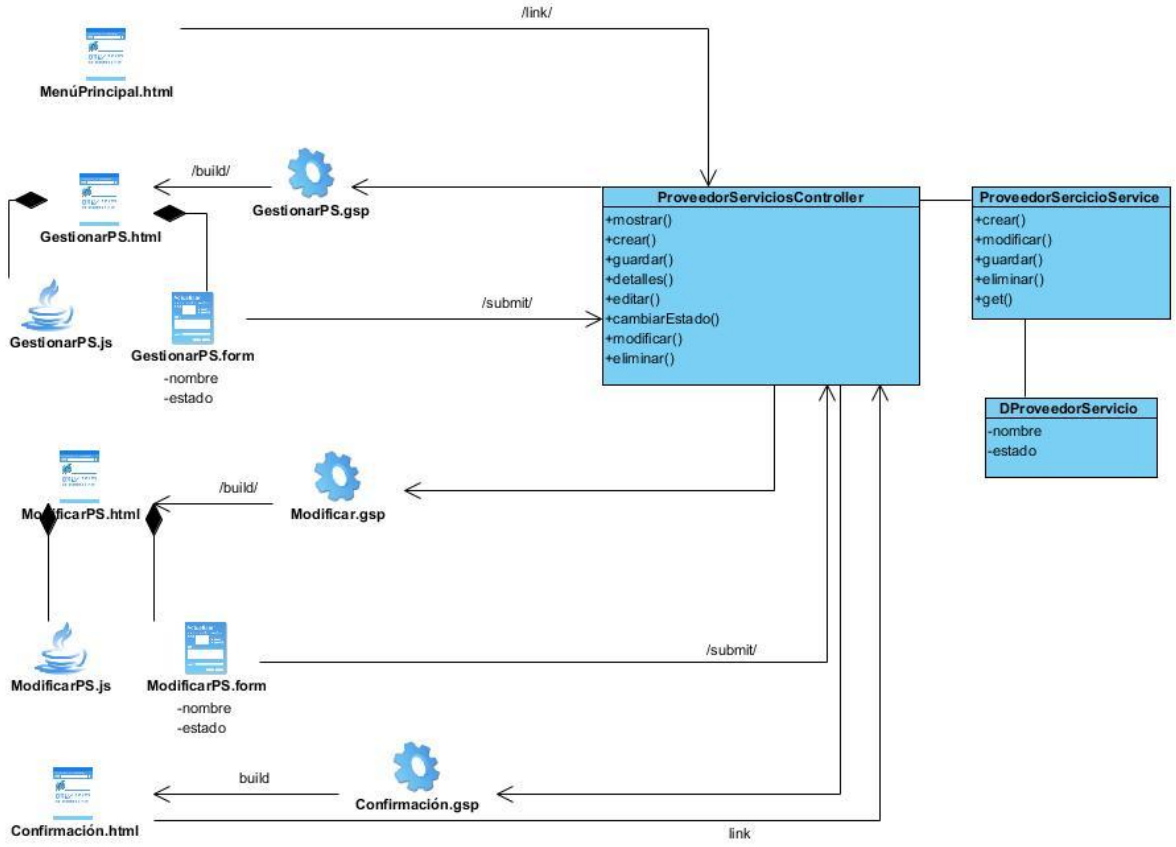

Ilustración 16: Diagrama de clases del diseño del CU Gestionar PS.

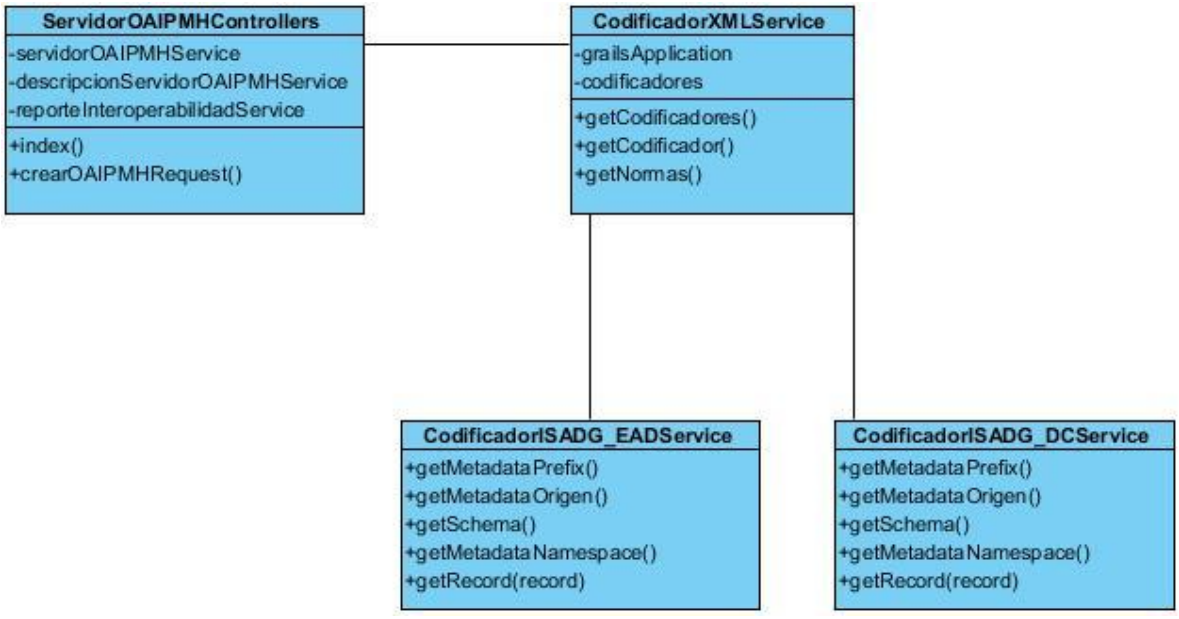

Ilustración 17: Diagrama de clases del diseño del CU Codificar Respuesta.

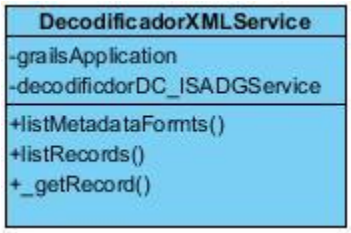

Ilustración 18: Diagrama de clases del diseño del CU Decodificar respuesta.

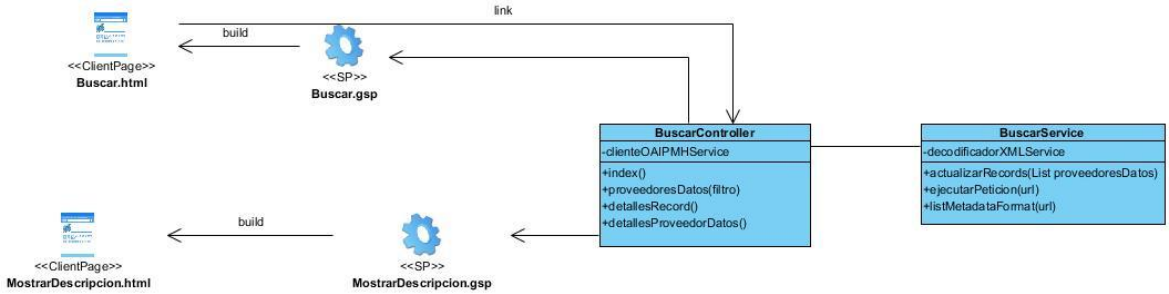

Ilustración 19: Diagrama de clases del diseño del CU Mostrar descripción del archivo.

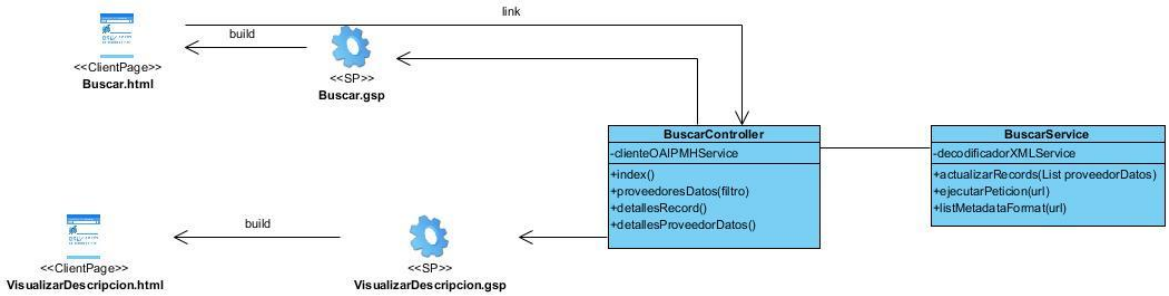

Ilustración 20: Diagrama de clases del diseño del CU Visualizar descripción del documento.

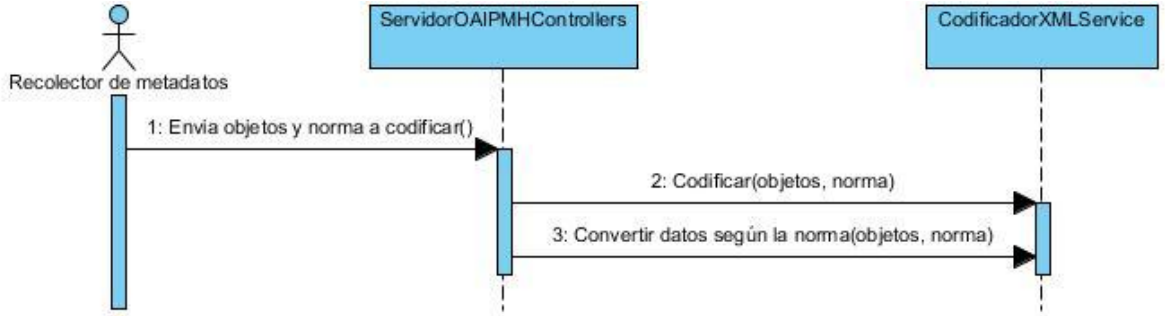

Ilustración 21: Diagrama de secuencia del CU Codificar respuesta.

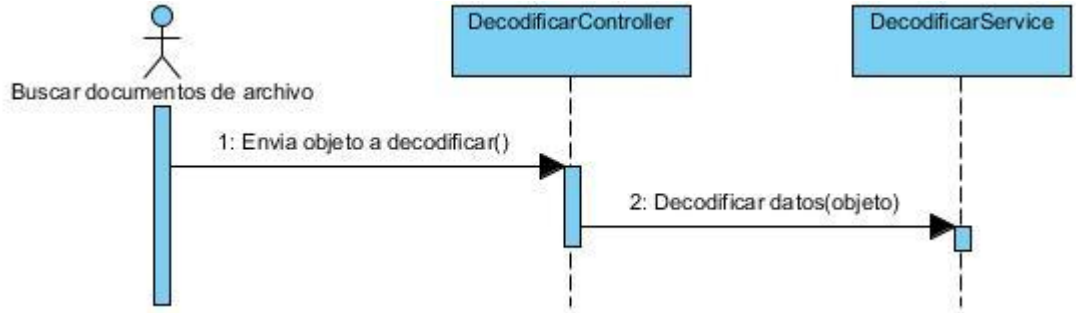

Ilustración 22: Diagrama de secuencia del CU Decodificar respuesta.

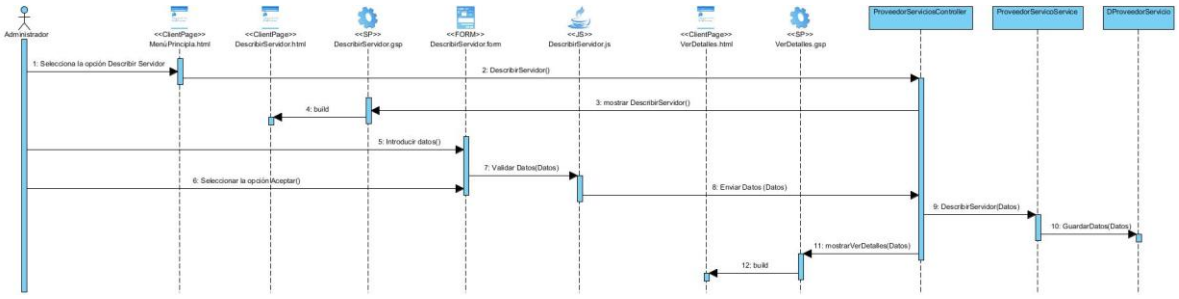

Ilustración 23: Diagrama de secuencia del CU Describir servidor OAI-PMH.

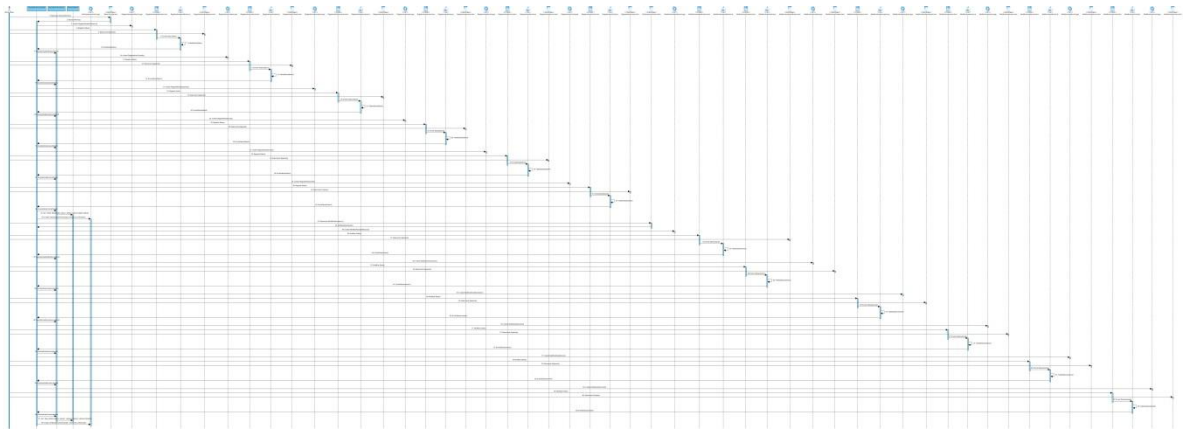

Ilustración 24: Diagrama de secuencia del CU Describir Archivo

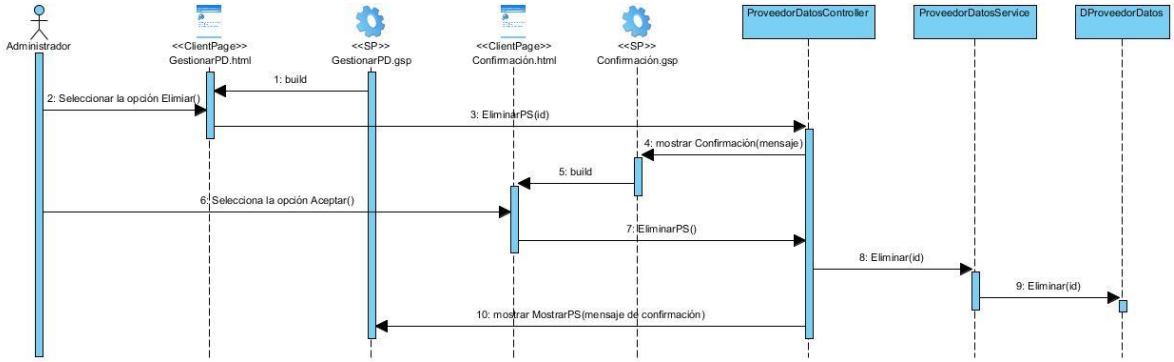

Ilustración 25: Diagrama de secuencia del CU Gestionar PD específicamente de la funcionalidad eliminar.

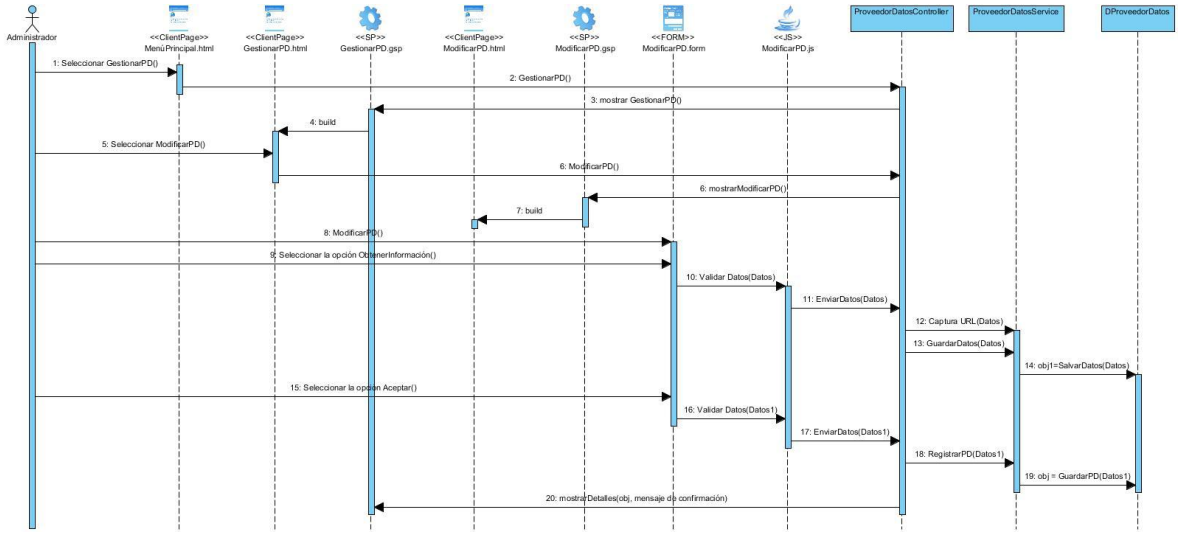

Ilustración 26: Diagrama de secuencia del CU Gestionar PD específicamente de la funcionalidad modificar.

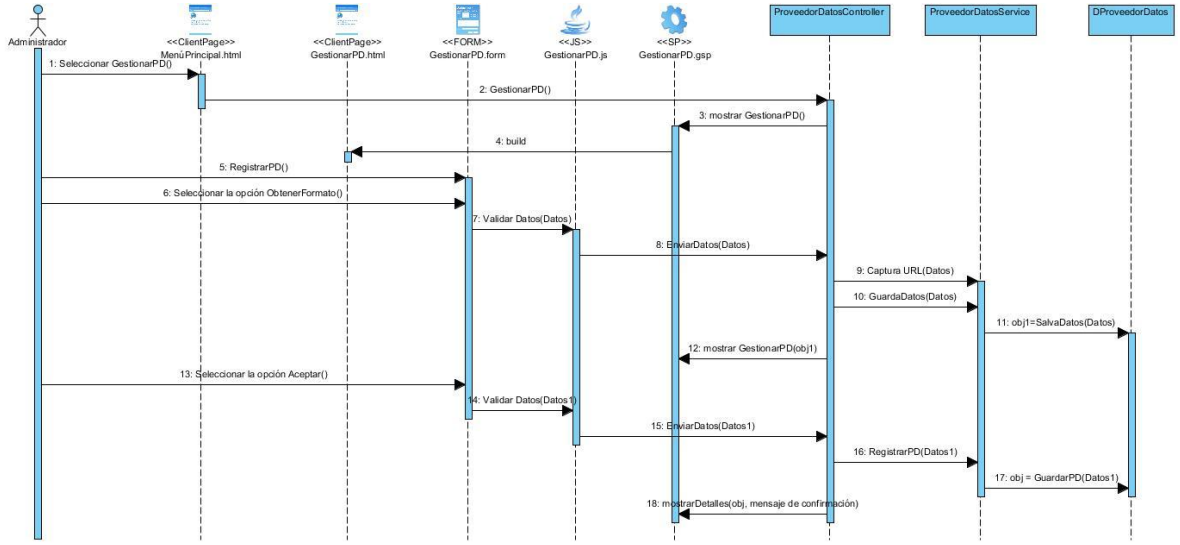

Ilustración 27: Diagrama de secuencia del CU Gestionar PD específicamente de la funcionalidad registrar.

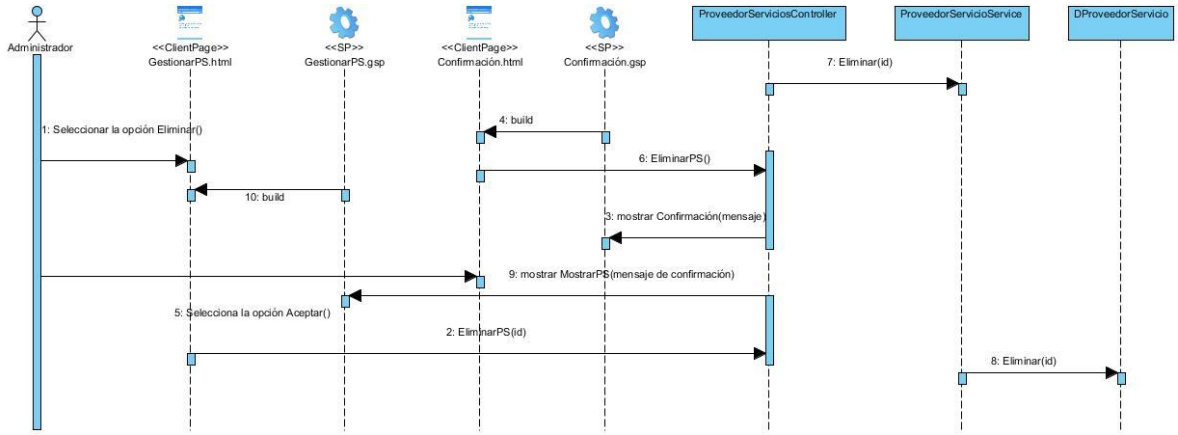

Ilustración 28: Diagrama de secuencia del CU Gestionar PS específicamente de la funcionalidad eliminar.

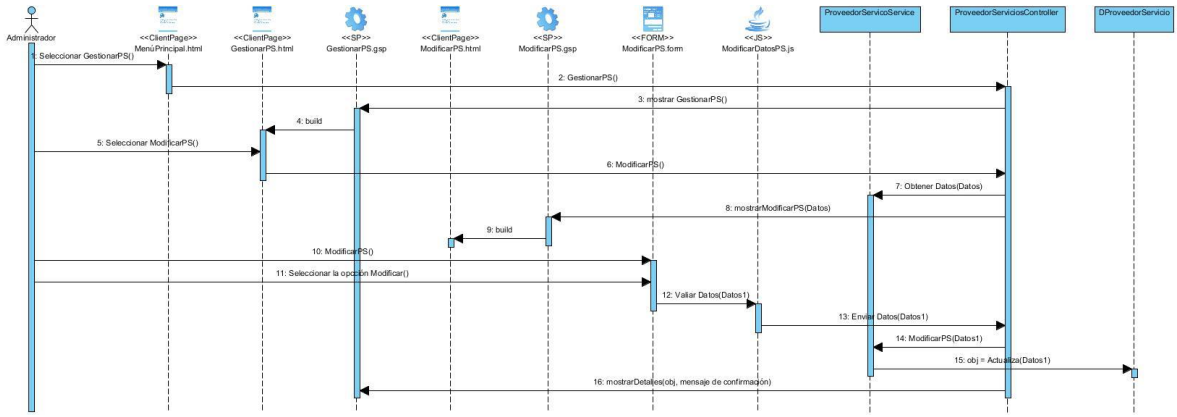

Ilustración 29: Diagrama de secuencia del CU Gestionar PS específicamente de la funcionalidad modificar.

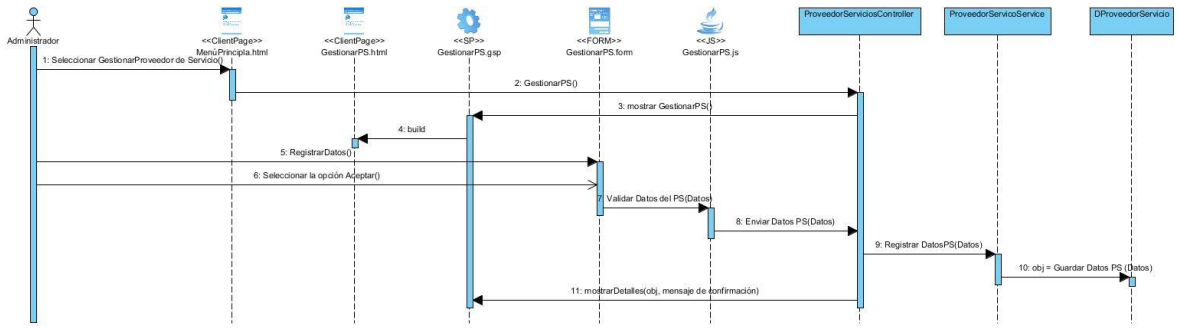

Ilustración 30: Diagrama de secuencia del CU Gestionar PS específicamente de la funcionalidad registrar.

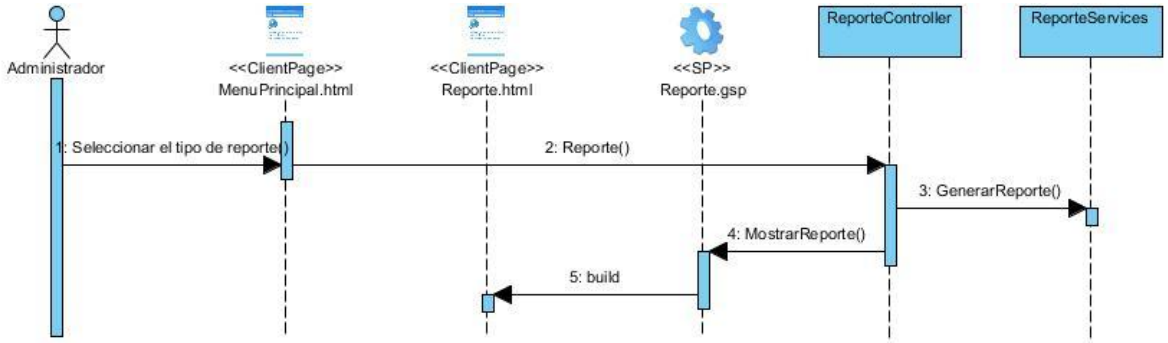

Ilustración 31: Diagrama de secuencia del CU Generar reporte.

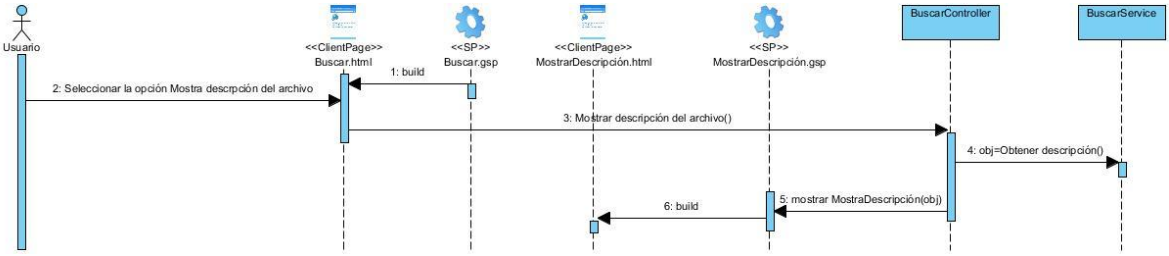

Ilustración 32: Diagrama de secuencia del CU Mostrar descripción del archivo.

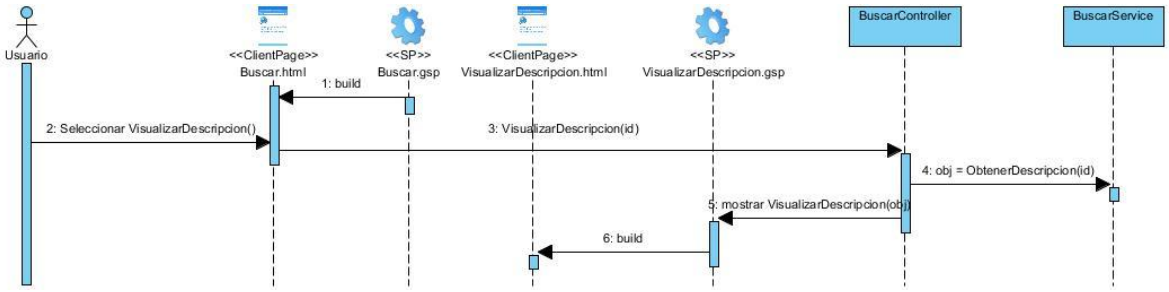

Ilustración 33: Diagrama de secuencia del CU Visualizar descripción de documentos de archivo.

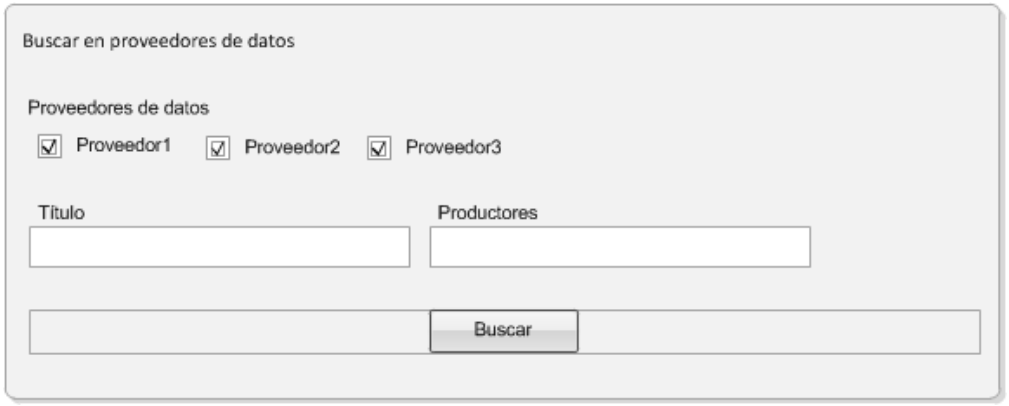

Ilustración 34: Prototipo de la interfaz Buscar.

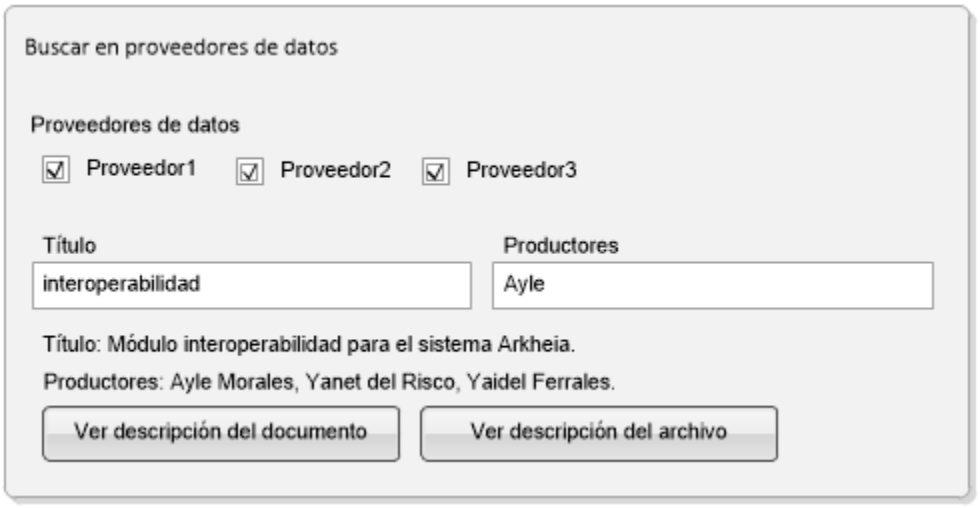

Ilustración 35: Prototipo del resultado de la búsqueda.

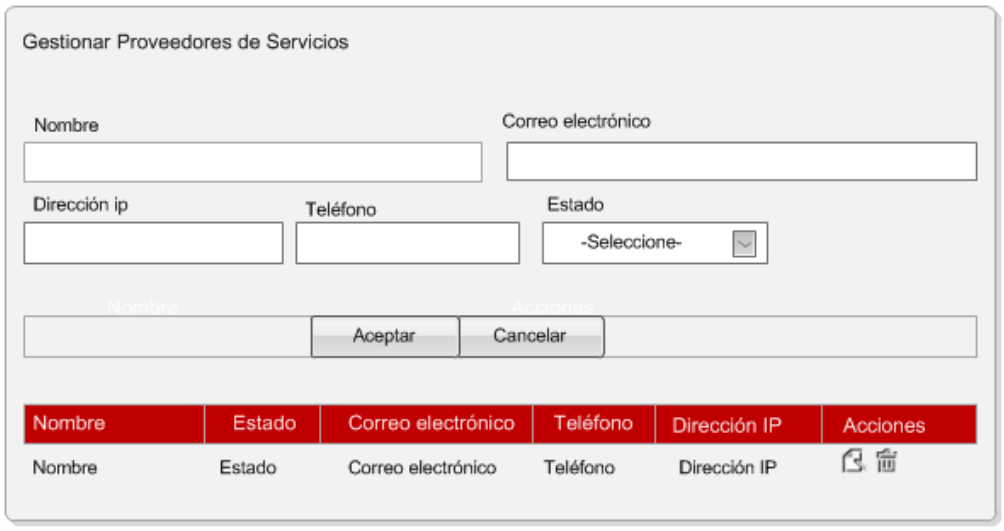

Ilustración 36: Prototipo de la interfaz Gestionar Proveedores de Servicios.

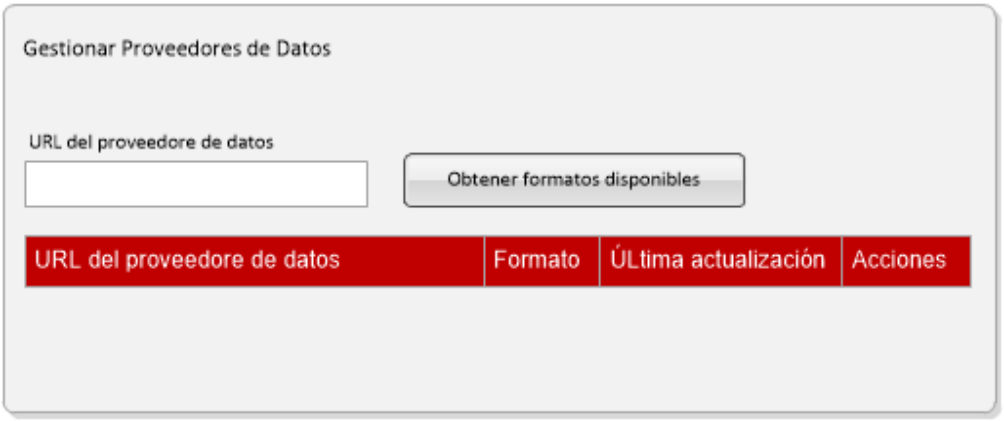

Ilustración 37: Prototipo de la interfaz Gestionar Proveedores de Datos.

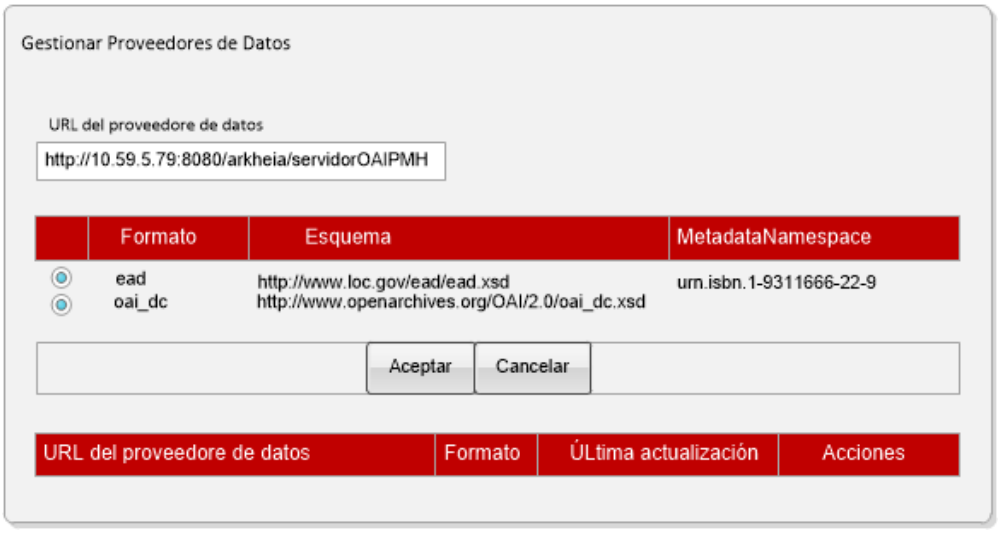

Ilustración 38: Prototipo de la interfaz Gestionar Proveedores de Datos luego de seleccionar la opción Obtener formatos disponibles.

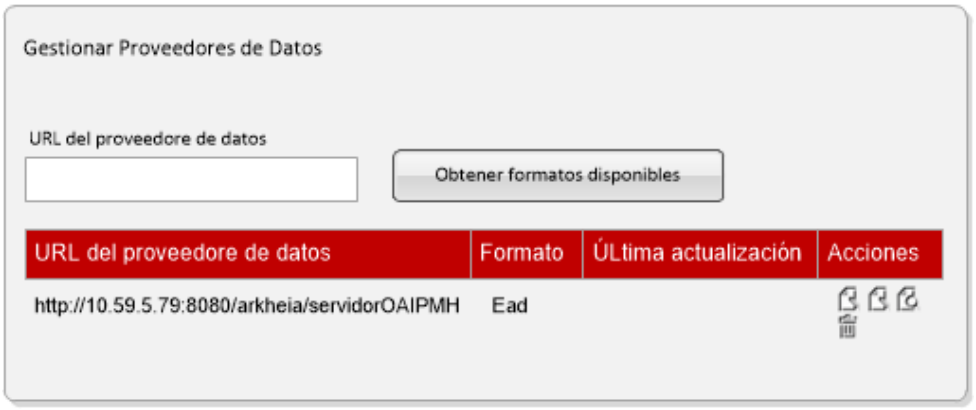

Ilustración 39: Prototipo de la interfaz Gestionar Proveedores de datos luego de seleccionar el formato.

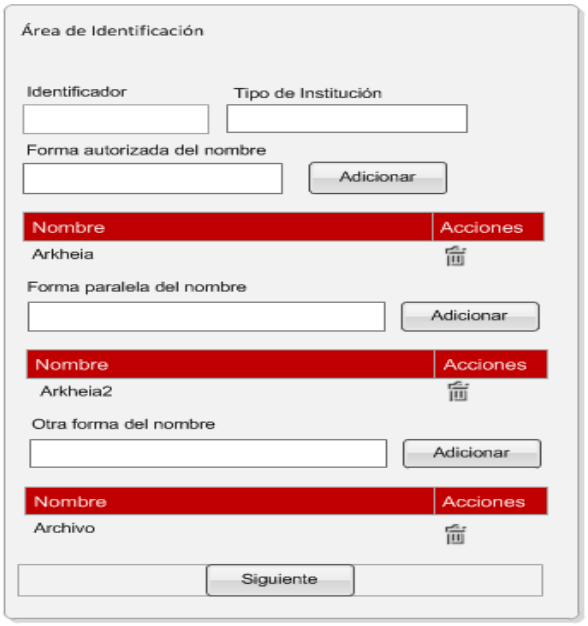

Ilustración 40: Prototipo de la interfaz Describir Archivo perteneciente al Área de Identificación.

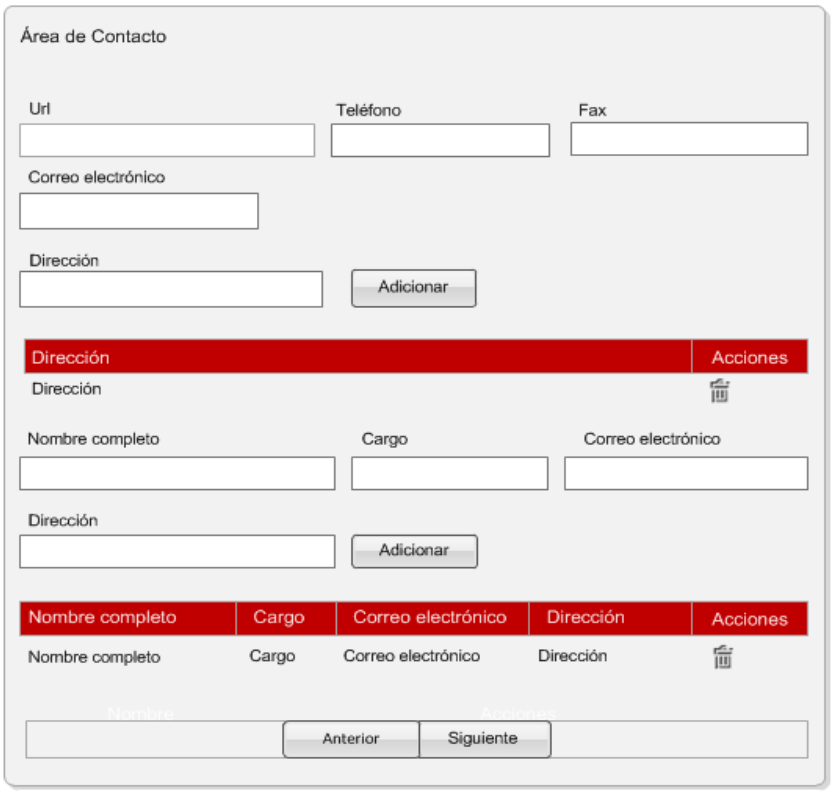

Ilustración 41: Prototipo de la interfaz Describir Archivo perteneciente al Área de Contacto.

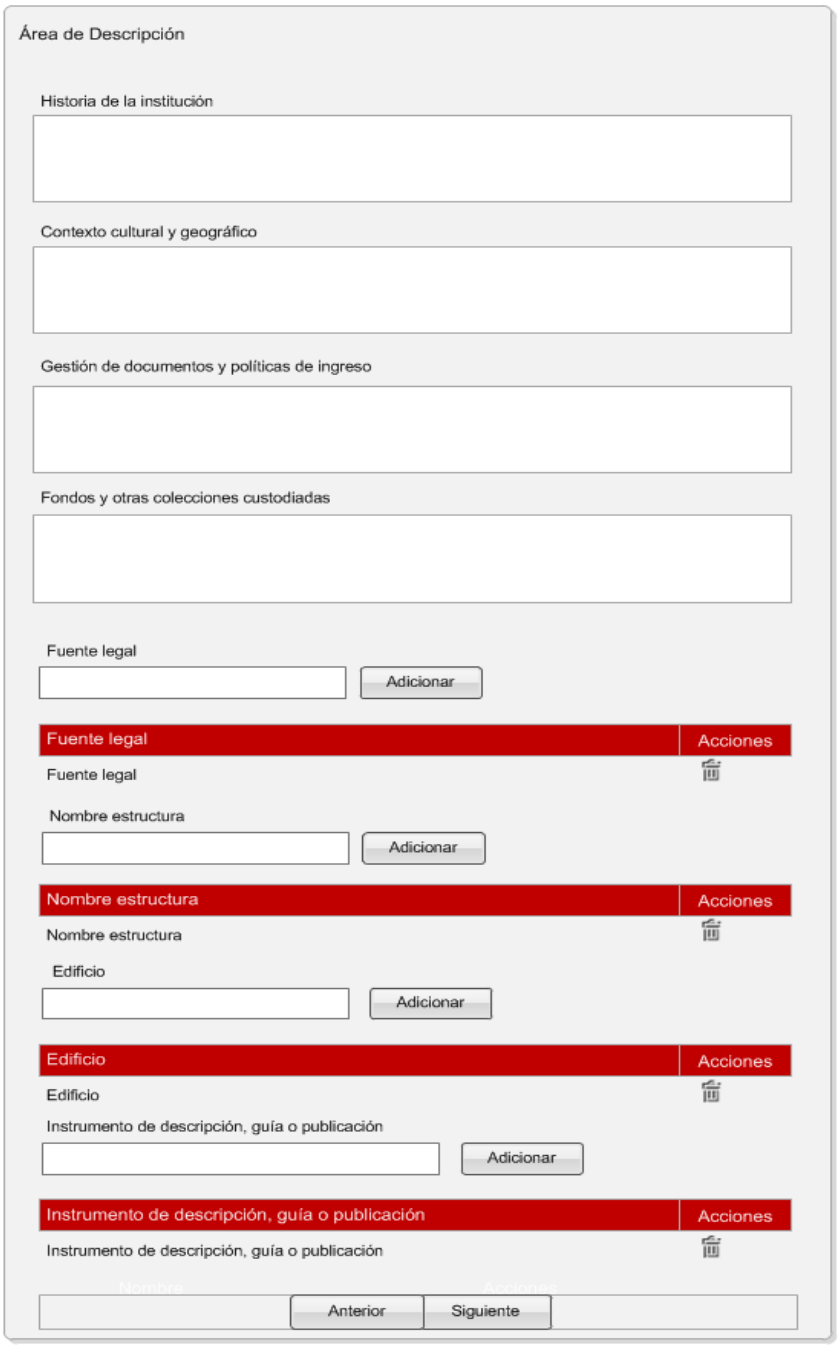

Ilustración 42: Prototipo de la interfaz Describir Archivo perteneciente al Área de Descripción.

Ilustración 43: Prototipo de la interfaz Describir Archivo perteneciente al Área de Acceso.

Ilustración 44: Prototipo de la interfaz Describir Archivo perteneciente al Área de Servicios.

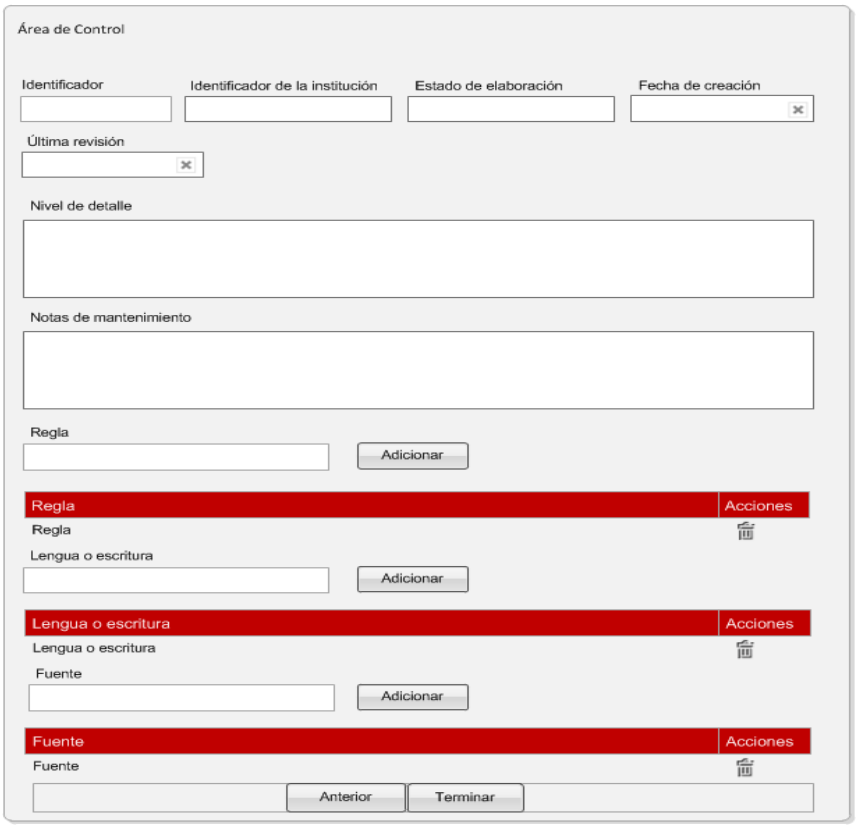

Ilustración 45: Prototipo de la interfaz Describir Archivo perteneciente al Área de Control.

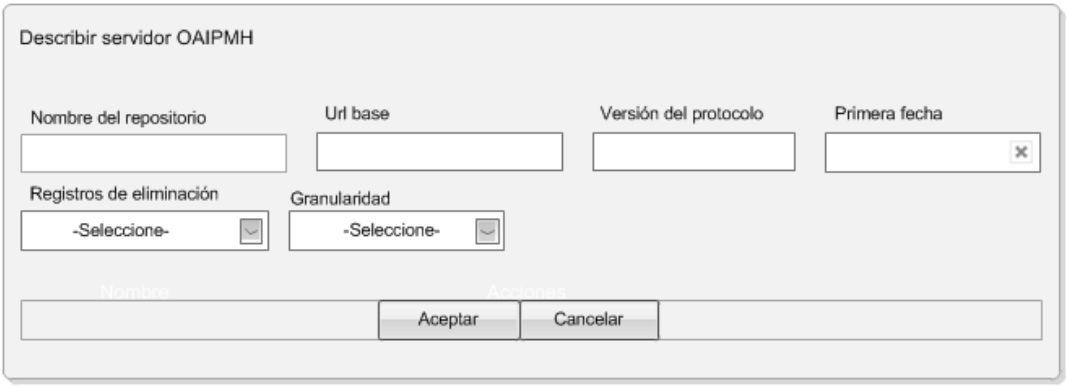

Ilustración 46: Prototipo de la interfaz Describir servidor OAIPMH.

Glosario de Términos

RePEc: Básicamente es un conjunto de herramientas conceptuales, protocolos, normas y software cuyo objeto es la distribución electrónica y descripción bibliográfica de los documentos científicos producidos por una disciplina académica, en concreto la Economía (12).

E-prints: 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.

XML: Lenguaje de marcado para documentos que contienen información estructurada. Un lenguaje de marcas es un mecanismo para identificar las estructuras en un documento (58).

OdiloTID: Compañía que se encarga del desarrollo de aplicaciones de gestión de centros de documentación.

Java 2 Enterprise Edition: Máquina virtual de java.

Convention Over Configuration) CoC: Es una filosofía de diseño y la técnica que trata de aplicar los valores predeterminados que pueden derivarse de la estructura del código en lugar de necesitar un código explícito (59).

(No te Repitas / Don't Repeat Yourself) DRY: No te Repitas (Don't Repeat Yourself, por sus siglas en inglés): Es una práctica de programación de computadoras de evitar la redundancia de código en las aplicaciones informáticas (60).

ORM: Mapeo de Objeto Relacional (Object Relational Mapping por sus siglas en inglés).

Scaffolding: Técnica de programación soportada por algunos marcos de trabajo Modelo Vista Controlador, en la cual el programador escribe una especificación que describe como la base de datos de la aplicación debe ser usada. El compilador utiliza esta especificación para generar código que la aplicación puede usar para crear, leer, actualizar y eliminar entradas de la base de datos.

Apache/BSD: Se refiere a la utilización de las licencias de Apache (The Apache license) requiere la conservación del aviso de derecho de autor, pero no es una licencia copyleft, ya que no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas (61) y la licencia BSD (Berkeley Software Distribution License) utilizada para la distribución de software. Permite reutilizar la totalidad o parte del software sin restricciones (62).

JVM: Java Virtual Machine por sus siglas en ingles es una máquina imaginaria que se implementa mediante la emulación en el software en una máquina real.

DOM: Modelo de Objeto del Documento, Document Object Model.

Spring Security: Es una API open source que provee de servicios de autenticación y autorización a las aplicaciones basadas en Spring.

URL (Uniform Resource Locator / Localizador Uniforme de Recursos): Es una referencia (una dirección) a un recurso en Internet.

Escapar: En un lenguaje de programación o marcado, sustituir caracteres que tienen funcionalidad por secuencias que representan el caracter, pero no la funcionalidad P.ej. Sustituir en HTML '>' por '>' (50).