



Universidad de las Ciencias Informáticas

Facultad 2

**Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas**

**Título: *“Herramienta para el procesamiento en lote de
documentos digitales para el sistema REPXOS”***

Autores:

Aray Villar Machado

José Javier Hernández Benítez

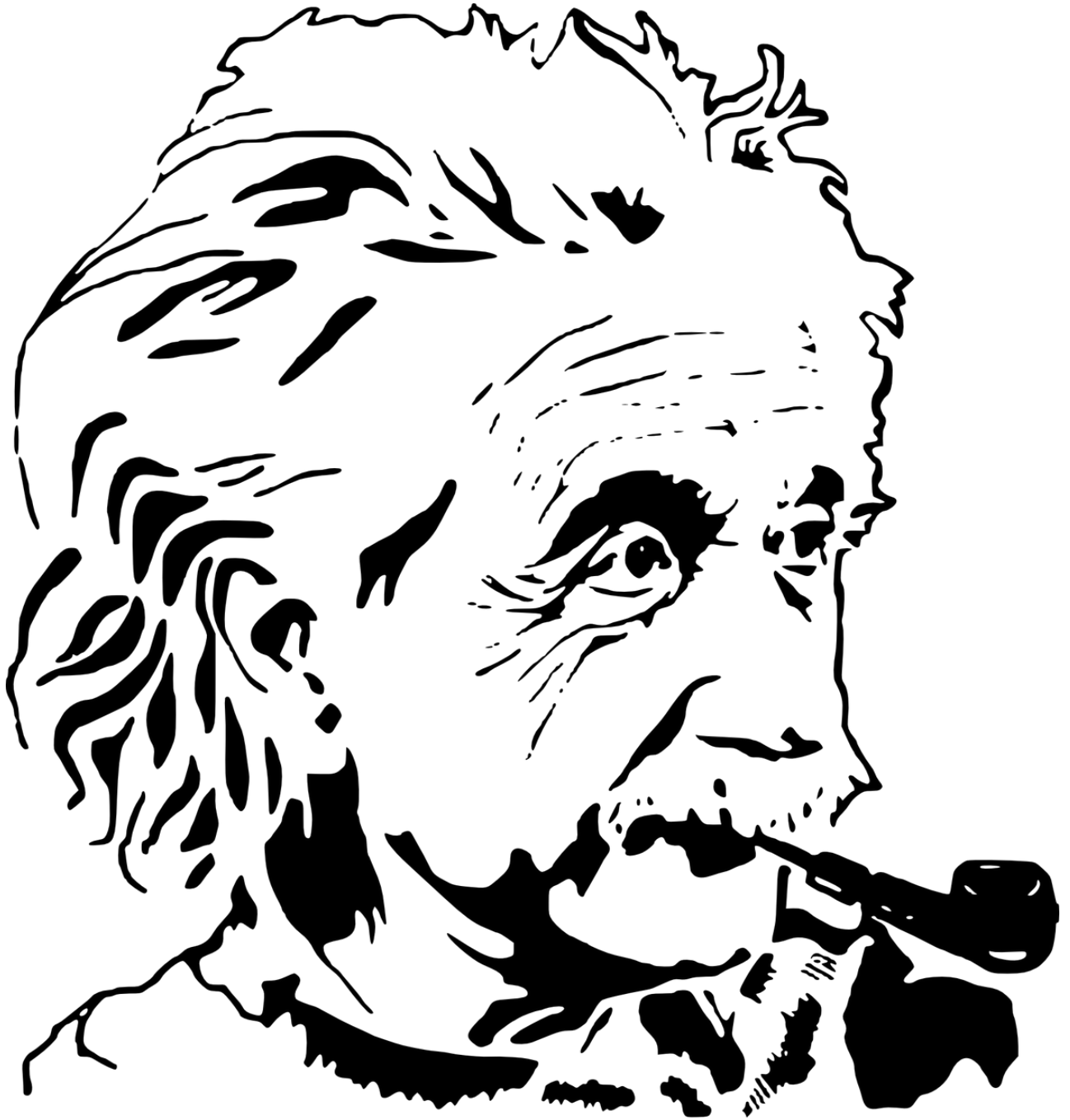
Tutores:

Ing. Luis Carlos Alvarez Fernández

Ing. Deylert Pérez Rivera

Ciudad de La Habana, 23 de junio de 2016

“Año 58 de la Revolución”



“Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber”

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los 23 días del mes de junio del año 2016.

Autores:

Aray Villar Machado

José Javier Hernández Benítez

Tutores:

Ing. Luis Carlos Álvarez Fernández

Ing. Deylert Pérez Rivera

DATOS DE CONTACTO

Tutores:

Tutor principal:

Ing. Luis Carlos Álvarez Fernández: Graduado en el año 2010 como Ingeniero en Ciencias Informáticas, en la Universidad de las Ciencias Informáticas. Se desempeña como arquitecto de Software en el Proyecto Productivo “Repositorio Institucional” en el Centro de Informatización de la Gestión Documental de la Universidad de las Ciencias Informáticas y miembro del Comité de Expertos en temas de Repositorios Digitales, del proyecto de cooperación universitaria "Red del VLIR UOS-Cuba". Correo electrónico: lcalvarez@uci.cu.

Ing. Deylert Pérez Rivera: Graduado en el año 2013 como Ingeniero en Ciencias Informáticas, en la Universidad de las Ciencias Informáticas. Se desempeña como desarrollador de Software en el Proyecto Productivo “Repositorio Institucional” en el Centro de Informatización de la Gestión Documental de la Universidad de las Ciencias Informáticas. Correo electrónico: drivera@uci.cu.

DEDICATORIA

“A mis padres y mi hermano.”

Aray

“A mis padres”

José Javier

AGRADECIMIENTOS

“A mis padres por ser mi fuente de inspiración, apoyarme siempre a pesar de estar lejos de ellos y en todo momento dejarme claro que están orgullosos de mí. Por haberme convertido en la persona que soy hoy gracias a su excelente educación.”

“A mi hermano que es mi vida y lo adoro.”

“A mi novio que me ha apoyado desde que nos conocimos, siendo siempre incondicional y muy especial en mi vida.”

“A mis familiares que siempre estuvieron presente y me brindaron su apoyo”

“A mis suegros que son para mí también como padres y los quiero mucho.”

“A mis amigos que son para siempre y contribuyeron a una mejor estancia durante estos 5 años lejos de mi familia.”

“A mis tutores y a Gleidis que siempre estuvieron dispuestos a ayudarme cuando lo necesité”

“A todo aquel que contribuyó con mi formación de una manera u otra muchas gracias.”

Aray

AGRADECIMIENTOS

“A mis padres Bárbara y Justo por todo el apoyo que me han brindado y por haber hecho de mi la persona que soy.”

“A mis abuelas Elda e Isabel que siempre han estado presente cuando las he necesitado.”

“A mi novia y compañera de tesis Aray por todo el amor que me ha dado.”

“A mi tío Juan Miguel y a mi tía Maira por toda la ayuda que me han dado.”

“A mis primos Jasiel, Javier, Abigail y Yordelis por su presencia.”

“A mis suegros por acogerme como un hijo más.”

“A mis amigos que durante 5 años han formado parte de mi vida y han sido mis hermanos.”

“A mis tutores Deylert y Luis Carlos por ayudarme y contribuir a mi formación como profesional.”

“A todo aquel que de una forma u otra me ha ayudado a lograr este sueño.”

A todos muchas gracias.

José Javier

RESUMEN

REPXOS 3.0 es un sistema para la implantación de repositorios digitales en instituciones científicas y académicas, desarrollado en el centro de Informatización de la Gestión Documental (CIGED), perteneciente a la Universidad de las Ciencias Informáticas (UCI). Para almacenar la información en REPXOS 3.0 es necesario llenar de forma manual documento a documento un formulario con los metadatos que lo describe. Una de las medidas que han tomado las instituciones es designar un grupo de especialistas durante jornadas de trabajo para incorporar dicha información a REPXOS 3.0.

Este proceso es fácil de realizar cuando es poca información pero cuando es en el orden de los cientos o miles de documentos, trae consigo pérdida de tiempo y empleo de recursos humanos, por lo que deciden en ocasiones no incorporar los documentos más antiguos, provocando que se pierda parte del patrimonio académico y científico de las instituciones.

Debido a esta situación se decidió desarrollar la aplicación web DEPXOS para facilitar el depósito de documentos digitales al sistema REPXOS 3.0. DEPXOS permite al usuario subir los ficheros que desea depositar, administrarlos, extraer sus metadatos automáticamente y editar dichos metadatos en caso que lo considere necesario para luego realizar su depósito en REPXOS 3.0.

Palabras clave: información, información científica, metadatos, procesamiento de documentos, repositorio digital.

ÍNDICE

INTRODUCCIÓN	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA	4
1.1 CONCEPTOS ASOCIADOS.....	4
1.2 DSPACE	5
1.3 DUBLIN CORE	6
1.4 HERRAMIENTAS DE EXTRACCIÓN DE METADATOS	6
1.5 MÉTODOS DE VALIDACIÓN DE METADATOS	10
1.6 SELECCIÓN DE LA VÍA DE DEPÓSITO DE DOCUMENTOS EN EL SISTEMA REPXOS 3.0.....	11
1.7 METODOLOGÍAS, TECNOLOGÍAS, HERRAMIENTAS Y LENGUAJES UTILIZADOS	13
1.8 CONCLUSIONES	18
CAPÍTULO 2: CARACTERÍSTICAS DE LA HERRAMIENTA PARA EL PROCESAMIENTO EN LOTE DE LOS DOCUMENTOS DIGITALES PARA EL SISTEMA REPXOS.	19
2.1 PROPUESTA DE SOLUCIÓN	19
2.2 MODELO DE DOMINIO	20
2.3 ESPECIFICACIÓN DE LOS REQUISITOS DE SOFTWARE	21
2.3.1 REQUISITOS FUNCIONALES	22
2.3.2 REQUISITOS NO FUNCIONALES	22
2.4 DEFINICIÓN DE LOS ACTORES	23
2.5 DEFINICIÓN DE LOS CASOS DE USO DE DEPXOS	24
2.6 CONCLUSIONES	26
CAPÍTULO 3: ARQUITECTURA Y DISEÑO DE LA HERRAMIENTA DEPXOS	27
3.1 MODELO DE DISEÑO	27
3.1.1 DIAGRAMA DE PAQUETES.....	27
3.2 PATRÓN ARQUITECTÓNICO	28
3.3 PATRONES DE DISEÑO UTILIZADO EN EL DESARROLLO DE DEPXOS	30
3.4 DIAGRAMA DE CLASES DEL DISEÑO UTILIZANDO ESTEREOTIPOS WEB	31
3.4.1 DIAGRAMAS DE INTERACCIÓN.....	32
3.5 MODELO DE DATOS	32
3.6 CONCLUSIONES	33
CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS DE LA HERRAMIENTA DEPXOS.....	34
4.1 MODELO DE IMPLEMENTACIÓN	34
4.1.1 DIAGRAMA DE COMPONENTES.....	34
4.2 ESTÁNDARES DE CODIFICACIÓN	35

4.3	MODELO DE DESPLIEGUE.....	35
4.3.1	DESCRIPCIÓN DE LOS PROTOCOLOS DE COMUNICACIÓN.....	35
4.4	PRUEBAS	36
4.5	RESULTADOS DE LAS PRUEBAS APLICADAS A DEPXOS.....	39
4.6	CONCLUSIONES	42
	CONCLUSIONES	44
	RECOMENDACIONES	45
	REFERENCIAS BIBLIOGRÁFICAS	46
	BIBLIOGRAFÍA	48
	ANEXOS.....	52
	GLOSARIO DE TÉRMINOS.....	78

INTRODUCCIÓN

Las instituciones académicas y científicas generan gran cantidad de información asociada a artículos científicos, revistas, trabajos de diplomas, libros, tesis de maestrías, tesis de doctorados, eventos, conferencias y clases. Toda esta información es de vital importancia almacenarla, con el objetivo de preservar y consultar el patrimonio científico de estas instituciones.

Las Tecnologías de la Información y las Comunicaciones (TIC) proporcionan un conjunto de aplicaciones y sistemas informáticos, que ofrecen la posibilidad de crear bibliotecas y repositorios digitales, con el fin de organizar, preservar, centralizar y difundir el patrimonio científico de las instituciones. Estas aplicaciones informáticas facilitan el almacenamiento y preservación de los documentos; así como su recuperación a través de herramientas de búsqueda por medio de los metadatos asociados a cada objeto digital.

Un ejemplo es REPXOS 3.0, un sistema de implantación de Repositorios Digitales desarrollado en el Centro de Informatización de la Gestión Documental (CIGED) perteneciente a la Universidad de las Ciencias Informáticas (UCI). Algunas instituciones que hacen uso de este sistema son el Instituto de Geografía Tropical (IGT), el Tribunal Supremo Popular (TSP), el Instituto Superior de Tecnologías y Ciencias Aplicadas (InSTEC) y la Oficina de Asuntos Históricos del Consejo de Estado (OAHCE). El Centro de Información Científico-Técnica de la UCI cuenta también con sus servicios.

Anualmente estas instituciones generan un gran cúmulo de documentos digitales, en el orden de los cientos o miles, que desean incorporar a su Repositorio Digital. Para realizar el depósito de un documento digital en REPXOS es necesario ingresar en un formulario, un conjunto de metadatos que describen el mismo (1). Este proceso se realiza de forma manual por lo que una de las medidas que se toma es designar varios especialistas durante numerosas sesiones de trabajo, para incorporar sus documentos.

Este proceso, aunque es intuitivo, cuando es necesario realizarlo para grandes cantidades de documentos, trae consigo pérdida de tiempo, y empleo de recursos humanos; por lo que en ocasiones los encargados de esta tarea deciden no incluir en el Repositorio Digital los documentos antiguos, perdiéndose así parte del patrimonio científico y académico de las instituciones.

Por lo antes planteado se puede definir como **problema a resolver**: ¿Cómo facilitar el depósito de documentos digitales al sistema REPXOS?

Del problema planteado se deriva el siguiente **objeto de estudio**: El procesamiento de documentos digitales. Enmarcándose en el **campo de acción**: Extracción automática de metadatos en documentos digitales. Se define como **objetivo general**: Desarrollar una herramienta para el procesamiento en lote de documentos digitales facilitando el depósito de los mismos en el sistema REPXOS.

Los **objetivos específicos** que se derivan del objetivo general son:

- Caracterizar las tendencias actuales a nivel mundial de la utilización de sistemas informáticos en la extracción automática de metadatos de documentos digitales.
- Diseñar una solución que contribuya al procesamiento en lote de los documentos digitales.
- Implementar la propuesta de solución haciendo uso de estándares y buenas prácticas.
- Validar la solución aplicando los métodos de pruebas.

Para lograr el cumplimiento de los objetivos expuestos anteriormente se desarrollaron las siguientes **tareas de la investigación**:

- Caracterización de los sistemas informáticos y herramientas usadas en la extracción automática de metadatos de documentos digitales, para identificar ventajas y desventajas de los mismos.
- Selección de una librería para la extracción automática de metadatos.
- Realización de un análisis de las funcionalidades de REPXOS 3.0 para definir el método de depósito de los documentos procesados y los metadatos asociados.
- Realización de un estudio de los métodos de validación de la calidad de los metadatos para la selección del que será utilizado en el desarrollo de la propuesta de solución.
- Caracterización de la metodología de desarrollo de software, herramientas, tecnologías y lenguajes de desarrollo que conforman la línea base de la arquitectura para el desarrollo de la propuesta de solución.
- Implementación de las funcionalidades de la propuesta de solución para el sistema REPXOS 3.0.
- Validación de la solución mediante los métodos de prueba para comprobar el correcto funcionamiento de la misma.

Los **métodos** utilizados en la investigación:

Métodos Teóricos:

Analítico-Sintético: Utilizado para ejecutar un análisis de la bibliografía correspondiente a la investigación y extraer los elementos significativos para el diseño eficiente de la propuesta de solución.

Modelación: Se utilizó para modelar los diagramas correspondientes a la fase de análisis y diseño, proporcionando una mejor perspectiva de la herramienta a desarrollar.

Métodos Empíricos:

Observación: Se empleó para evaluar los resultados obtenidos contra los resultados esperados con el desarrollo de la herramienta.

Estructura de la tesis:

El presente trabajo de diploma se distribuye de la siguiente manera: introducción, capítulos 1, 2, 3 y 4, conclusiones, recomendaciones, bibliografía, referencias bibliográficas, anexos y glosario de términos. Los capítulos están organizados de la siguiente manera:

Capítulo 1: Fundamentación Teórica

Se analizan las tendencias y tecnologías actuales a nivel mundial, concernientes a la extracción automática de metadatos de documentos digitales y se seleccionan las herramientas utilizadas en el desarrollo de la propuesta de solución. Se fundamenta el uso de las herramientas, tecnologías, metodología de desarrollo de software y lenguajes de programación que se utilizan, lo que constituye el basamento teórico de la investigación.

Capítulo 2: Características de la herramienta para el procesamiento en lote de los documentos digitales para el sistema REPXOS

Se elabora un modelo de dominio donde se analizan las entidades y conceptos presentes en el contexto donde se desarrolla la propuesta de solución, se definen las relaciones existentes entre cada uno de estos. Se especifican los requisitos funcionales y no funcionales con los que debe cumplir el sistema, se muestra el diagrama de casos de uso del sistema y se describen los actores y los casos de uso del sistema.

Capítulo 3: Arquitectura y diseño de DEPXOS

Se propone el diseño de la herramienta para el procesamiento en lote de documentos digitales para el sistema REPXOS (DEPXOS), a partir de los diagramas de clases del diseño empleando estereotipos web, los diagramas de colaboración, el diagrama de paquetes y el modelo de datos. Se describen los elementos de estos diagramas, así como el estilo arquitectónico y los patrones de diseño utilizados en el desarrollo de la herramienta.

Capítulo 4: Implementación y pruebas de la herramienta DEPXOS

Se pretende explicar a partir de los resultados del diseño, la implementación del sistema DEPXOS. Como parte de la implementación se muestra el diagrama de componentes, el cual contribuye a la organización del subsistema y representa la vista estática de este, y el diagrama de despliegue en el que se visualiza la situación física del sistema. Se definen las pruebas de software y algunos de los resultados obtenidos de las mismas, además de los estándares de codificación.

Capítulo 1: Fundamentación Teórica

En el presente capítulo se analizan las tendencias y tecnologías novedosas existentes en el mundo, relacionadas con la extracción automática de metadatos de documentos digitales, persiguiendo el objetivo de definir la herramienta de extracción que se va a utilizar en el desarrollo de la herramienta para el procesamiento en lote de documentos digitales para el sistema REPXOS. Se fundamenta el uso de las herramientas, tecnologías, metodología de desarrollo de software y lenguajes de programación que se utilizan, lo que constituye el basamento teórico de la investigación.

1.1 Conceptos Asociados

A continuación se relacionan algunos de los conceptos que se manejan en los temas referidos al procesamiento de documentos digitales, para permitir una mayor comprensión del funcionamiento de la propuesta de solución apoyando la investigación en curso.

Metadato

“(…) es toda aquella información descriptiva sobre el contexto, calidad, condición o características de un recurso, dato u objeto que tiene la finalidad de facilitar su recuperación, autenticación, evaluación y preservación (…)” (2)

Atendiendo a fines prácticos, los tipos y funciones de los metadatos pueden clasificarse en tres amplias categorías -descriptivos, estructurales y administrativos- con límites no siempre bien definidos y a veces superpuesto. Por ejemplo, los metadatos administrativos pueden incluir una amplia gama de información que podría considerarse como metadatos descriptivos y estructurales. (3)

Metadatos descriptivos: Son aquellos que sirven para la descripción e identificación de los recursos de información, permiten la búsqueda y recuperación de la información, como también distinguir un recurso de otro y entender el asunto o contenido del mismo. Se realizan mediante los estándares como Dublin Core; Registro Catalográfico Legible por Máquina (MARC); Lenguaje de Marcas de Hipertexto (HTML), etc. (3)

Metadatos estructurales: Estos tipos de metadatos son los que más influyen en la recuperación de la información electrónica, facilitan la navegación y presentación de los recursos electrónicos. Así, ofrecen la información sobre la estructura interna de los recursos, estableciendo las relaciones entre ellos, de manera que pueden incluso unir los archivos de imagen y textos que están relacionados. Los estándares más difundidos para ellos son Lenguaje de Marcado Generalizado Estándar (SGML) y Lenguaje de Marcas Extensible / Marco de Descripción de Recursos (XML/RDF) y Descripción de Archivística Codificada (EAD). (3)

Metadatos administrativos: Los metadatos administrativos son de carácter más técnico porque incluyen datos sobre la creación y control de calidad, datos sobre la gestión de derechos, requisitos del control de acceso y utilización, información sobre la gestión y procesamiento de las colecciones

digitales a largo y corto plazo. Ejemplo de estos metadatos son: tipo y modelo de escáner utilizado, resolución, limitaciones de reproducción, etc. (3)

Extracción automática de metadatos

La extracción automática de metadatos radica en recuperar un conjunto de atributos de un recurso digital usando para esto una herramienta informática. Una vez extraídos se pueden utilizar para describir e identificar los recursos digitales, para luego ser depositados en una base de datos, repositorio o lugar donde se almacene información digital y preservarla.

Ítem

En DSpace un ítem se refiere a un fichero en sí, conocido como “bitstreams” y el conjunto de metadatos que describe ese fichero. (1)

REPXOS

Es un sistema para la implantación de repositorios digitales. Permite adaptarse a múltiples usos para la gestión y recuperación de documentos digitales en cualquier formato. Permite incorporar y buscar documentos además de navegar dentro de ellos, administrar el sistema y realizar suscripciones a las diferentes colecciones. Está basado en el software DSpace. (4)

1.2 DSpace

DSpace está desarrollado en Java bajo la licencia Distribución de Software Berkeley (BSD), además de las características mencionadas previamente, posee la capacidad para ser personalizado, permitiendo satisfacer las necesidades de cualquier institución. Algunas de las principales formas en las que se puede personalizar la aplicación son: (3)

Interfaz de usuario: Permite adaptar la apariencia del sitio web a la imagen corporativa de su institución. DSpace ofrece dos opciones de interfaz de usuario: la interfaz tradicional, JSPUI (basado en Páginas de Servidor Java (JSP)), y XMLUI (basado en XML), que ofrece varios diseños de interfaz de usuario.

Personalizar los metadatos: Dublin Core es el formato de metadatos nativo dentro de DSpace, sin embargo, se puede agregar o cambiar cualquier campo.

Estándares compatibles: DSpace cumple con muchos protocolos estándar para el acceso, importación y exportación de datos. Los estándares que soporta DSpace son: Iniciativa de Archivos Abiertos - Protocolo para la Recolección de Metadatos (OAI-PMH), Servicio Web simple que ofrece servicios de depósito en un repositorio (SWORD), Creación y control de versiones distribuidos en web (WebDAV), OpenSearch, OpenURL, Distribución Realmente Sencilla (RSS), etc.

Configuración de la navegación y el motor de búsqueda: Se puede decidir qué campos se desean mostrar para la navegación, tales como autor, título, fecha, idioma, etc. De igual forma se pueden configurar los campos de metadatos que se requieran en la interfaz de búsqueda. DSpace permite la

indexación a texto completo de cualquier ítem para permitir las búsquedas por contenido si se considera oportuno.

Capacidad de utilización de los mecanismos de autenticación local: DSpace incorpora plugins para la mayoría de los métodos de autenticación, cuenta con su propio método de autenticación interna, o también se puede configurar para utilizar varios métodos de autenticación a la vez.

Base de datos configurable: Posee soporte para dos sistemas gestores de bases de datos: PostgreSQL y Oracle.

Idioma: DSpace está disponible en más de veinte idiomas por lo que se puede configurar para que sea compatible con varios idiomas a la vez.

1.3 Dublin Core

Es el formato de metadatos nativo dentro de DSpace. Es un estándar de metadatos simple, eficaz para describir una amplia gama de recursos de red. Cada elemento es opcional y puede ser repetido, soporta dos niveles: Simple y Cualificado. (5) En el Anexo 1 se puede encontrar los elementos y cualificadores del estándar Dublin Core.

1.4 Herramientas de extracción de metadatos

Existen herramientas y librerías que permiten extraer de forma automática los metadatos de los documentos digitales. A continuación se describen algunas de ellas.

Apache Tika

Es un conjunto de herramientas para detectar y extraer metadatos y texto estructurado del contenido de varios tipos de documentos usando librerías para el análisis sintáctico. Tika es un proyecto de la Fundación de Software de Apache (ASF) escrito en Java. Proporciona un único conjunto de funciones y una única interfaz Java para acceder a las librerías de análisis sintáctico, consume pocos recursos y posee un rápido procesamiento. Se distribuye bajo la Licencia Apache 2.0. (6)

Los formatos que puede extraer la herramienta son los siguientes (versión 1.12) (6)

- HTML
- XML y derivados
- Formatos de documentos de Microsoft Office
- Formato OpenDocument
- Formato de documentos iWorks
- Formato de documento portátil (PDF)
- Formato de publicaciones electrónicas (EPUB)
- Formato de texto enriquecido (RTF)
- Formatos de compresión y empaquetado (Tar, AR, CPIO, Zip, 7Zip, Gzip, BZip2, XZ, Pack200 y RAR)

- Formato de texto
- Formatos de fuentes y sindicación
- Formatos de ayuda (CHM)
- Formatos de audio
- Formatos de imagen
- Formatos de videos
- Archivos de clases de Java y compilados (JAR)
- Código fuente (Java, C, C++, Groovy y otros)
- Formatos de correo electrónico
- Formatos CAD
- Formatos de fuentes tipográficas
- Formatos científicos
- Programas ejecutables y librerías
- Formatos criptográficos
- Formatos de bases de datos (SQLite y Microsoft Access)

FOCA

Es una herramienta utilizada principalmente para encontrar metadatos e información oculta en los documentos que examina, enfocado en procesos de auditoría de red. Estos documentos pueden estar en páginas web, con FOCA se pueden descargar y analizar. Los documentos son buscados usando tres posibles buscadores: Google, Bing y Exalead, aunque también existe la posibilidad de añadir ficheros locales para extraer sus metadatos (7).

Una vez que se tienen todos los datos extraídos, FOCA une la información buscando qué documentos han sido creados en el mismo equipo y qué servidores y equipos pueden inferirse de dichos datos. FOCA es distribuido bajo la licencia Acuerdo de Licencia con el Usuario Final (EULA). (7)

Soporta los formatos (7)

- Formatos de documentos (Microsoft Office y Open Office)
- Formato PDF
- Formatos de Adobe InDesing
- Formatos SVG

Metadata Extraction Tool

Es una herramienta de código abierto escrita en Java para la extracción de metadatos de distintas fuentes con el objetivo de usarlos para tareas de preservación de recursos digitales. Permite la extracción automática de metadatos enfocados a la preservación de recursos digitales y exporta dichos

metadatos en formato XML listos para ser usados en actividades de preservación. Puede ser usada desde una interfaz de usuario en Windows o desde la línea de comandos en sistemas UNIX para procesar ficheros por lotes o de forma individual. Se distribuye bajo los términos de la Licencia Apache 2.0. (8)

Soporta los formatos (8)

- Formatos de imagen: BMP, GIF, JPEG y TIFF
- Formatos de documentos (MS Word, Word Perfect, Open Office, MS Works, MS Excel, MS Power Point y PDF)
- Formatos de audio y video (WAV y MP3)
- Formatos de lenguajes de marcado (HTML y XML)

GROBID

Es una librería de aprendizaje automático de código abierto. Distribuida bajo los términos de la licencia Apache 2.0, para la extracción, análisis sintáctico y la reestructuración de documentos sin estructura, como el formato PDF, en documentos codificados con el estándar TEI con un enfoque particular en publicaciones técnicas y científicas. Posee las siguientes funcionalidades: (9)

- Extracción de cabecera y análisis sintáctico de artículos en formato PDF. La extracción cubre la información bibliográfica (por ejemplo, título, resumen, autores, afiliaciones, palabras clave, etc.).
- Extracción y análisis sintáctico de referencias de artículos en formato PDF. (Funciona con notas al pie de página).
- Análisis sintáctico de nombres, particularmente de nombres de autor en las cabeceras y nombres de autor en las referencias (dos modelos distintos).
- Análisis sintáctico de afiliaciones y bloques de direcciones.
- Análisis sintáctico de fechas.
- Extracción completa de texto de artículos en formato PDF.

La tabla 1 muestra las principales características de las herramientas mencionadas anteriormente.

Herramientas / Características	Licencia	Características	Plataformas soportadas	Formatos soportados	Propósito
Apache Tika	Apache 2.0	Conjunto de herramientas	Multiplataforma	HTML, XML y derivados, formatos de	General

		escritas en Java. Puede ser usada desde una interfaz gráfica, desde la línea de comandos o como librería de terceros en aplicaciones escritas en Java.		documentos de Microsoft Office, formato OpenDocument, PDF, formatos de audio, video, etc. (alrededor de 24 tipos de formatos)	
FOCA	EULA	Aplicación desktop con interfaz de usuario. Escrito en .NET	Windows	Formatos de documentos (Microsoft Office y Open Office), PDF, formatos de Adobe InDesing y formatos SVG	Auditoría de redes
Metadata Extraction Tool	Apache 2.0	Herramienta escrita en Java. Puede ser usada desde una interfaz gráfica, desde la línea de comandos o como librería de terceros en aplicaciones escritas en Java.	Multiplataforma	HTML, XML, formatos de documentos de Microsoft Office, Open Office, PDF y formatos de audio y video (MP3 y WAV)	Preservación de recursos digitales
GROBID	Apache 2.0	Biblioteca de aprendizaje automático escrita en Java. Puede ser usada en proyectos escritos en Java,	Multiplataforma	PDF	Publicaciones técnicas y

		mediante un servicio REST o mediante la línea de comandos			científicas
--	--	---	--	--	-------------

Tabla 1: Herramientas y sus características.

Fueron seleccionadas para el desarrollo de la propuesta de solución la librería de aprendizaje automático GROBID y el conjunto de herramientas Apache Tika. Con el uso de la funcionalidad para la extracción de cabeceras de GROBID es posible el acceso a información invariable y correcta, algo que se puede ver afectado en los metadatos de los recursos, debido a que estos pueden ser alterados por las aplicaciones donde son creados, editados, etc. Además, GROBID posee un enfoque particular en documentos técnicos y científicos. A pesar de todo lo descrito anteriormente esta librería presenta una desventaja, solo permite el procesamiento de recursos en formato PDF. Debido a que la propuesta de solución se enfrentará a recursos de disímiles formatos se seleccionó también Apache Tika, pues respecto al resto de las herramientas estudiadas es la que mayor cantidad de formatos soporta y posee mecanismos específicos para realizar la extracción de metadatos utilizando el estándar Dublin Core.

1.5 Métodos de validación de metadatos

Se recomienda el uso del marco de trabajo Bruce & Hillman para la evaluación automática de metadatos en repositorios digitales por el hecho de que los siete parámetros utilizados en dicho marco de trabajo son fáciles de entender por humanos y además capturan todas las dimensiones de calidad propuestas en otros marcos de trabajo. Su nivel de compactación también ayuda a realizar la medición de la calidad de los metadatos automáticamente. El marco de trabajo Bruce & Hillman define los siguientes siete parámetros para medir la calidad de los metadatos: (10)

Compleitud: Una instancia de metadato debe describir el recurso lo más completo posible. Mientras más campos de metadatos contenga el recurso, más diversos serán los usos que se le pueda dar al mismo. (10)

Exactitud: La información proporcionada por la instancia de metadato debe ser la más correcta posible. Los errores tipográficos y los errores de hecho afectan esta dimensión de la calidad. La exactitud puede ser medida con valores booleanos en casos como el tamaño del fichero o el formato, sin embargo, campos como el título o la descripción contienen un rango más amplio para evaluar. (10)

Procedencia: La fuente de los metadatos puede ser otro factor para determinar su calidad. Conocimiento sobre quién creó la instancia, el nivel de experiencia del indexador, las metodologías que fueron seguidas durante la indexación y las transformaciones que han sufrido los metadatos pueden dar una idea de la calidad de la instancia. (10)

Conformidad con lo esperado: El grado con que los metadatos cumplen los requisitos de una

determinada comunidad de usuarios para una tarea dada se puede considerar como una dimensión importante de calidad. Si la información almacenada en los metadatos ayuda a la comunidad a encontrar, identificar, seleccionar y obtener recursos sin la necesidad de realizar un gran cambio en su flujo de trabajo esta se puede considerar conforme a las expectativas de la comunidad. (10)

Consistencia lógica y coherencia: Los metadatos deben ser consistentes con los estándares y conceptos del dominio. La información contenida en los metadatos también debe tener coherencia interna, es decir, todos los campos de la instancia de metadato deben describir al mismo recurso de forma similar. (10)

Accesibilidad: Los metadatos que no pueden ser leídos o entendidos no poseen ningún valor. Si los metadatos van a ser utilizados por un GPS, el principal problema al que se puede enfrentar es a la accesibilidad física, por ejemplo, formatos incompatibles o enlaces rotos. Por otro lado, si los metadatos van a ser utilizados por los humanos, por ejemplo, la descripción, el mayor problema es la accesibilidad cognitiva (el metadato es muy difícil de entender). Estas dos dimensiones deben ser combinadas para estimar que tan difícil es acceder y comprender la información presente en los metadatos. (10)

Puntualidad: Los metadatos deben cambiar cada vez que el recurso es modificado. La puntualidad en un repositorio digital se relaciona principalmente con el grado en que una instancia de metadato sigue siendo actual. La actualidad de una instancia de metadatos podría medirse como lo útil que los metadatos permanecen con el paso del tiempo. (10)

Para la propuesta de solución fue seleccionado el parámetro de validación de metadatos completitud, debido a su facilidad de implementación y a que es ideal para recursos que aún no se encuentran dentro del repositorio. Los demás parámetros requieren el acceso a todos los recursos e instancias de metadatos de un repositorio, leer todo el contenido de un recurso o procedimientos de cálculo más complicados que pueden afectar la velocidad de respuesta de la propuesta de solución.

1.6 Selección de la vía de depósito de documentos en el Sistema REPXOS 3.0.

En REPXOS es posible el depósito de ítems mediante el uso del protocolo SWORD o usando REST.

SWORD

Es un protocolo ligero para el depósito de contenido de una ubicación a otra y está basado en el estándar AtomPub. SWORD interactúa con el repositorio en dos vías: (12)

- El cliente SWORD puede solicitar un documento de servicio (describe el contrato entre un usuario y el repositorio: lo que el usuario debe proporcionar y la respuesta que devolverá el repositorio) al servidor SWORD (alojado en el repositorio). El documento de servicio lista las colecciones en las cuales el usuario puede hacer el depósito y el tipo de objetos que acepta. (12)

- El cliente SWORD puede enviar un fichero al servidor SWORD. El servidor procesaría el depósito y respondería devolviendo una entrada Atom que contendría la URL del nuevo ítem creado. (12)

Normalmente los usuarios deben proporcionar sus credenciales de inicio de sesión para solicitar un documento de servicio o hacer un depósito. El servicio de envío se puede configurar para hacer más simple u omitir pasos en el proceso de envío. También es posible que un usuario haga un depósito en nombre de otro, por ejemplo, un administrador puede hacer depósitos en nombre de un usuario (depósito mediado) o un sistema de publicación central puede depositar en nombre de un autor. En este escenario no solo se validan las credenciales del usuario, sino también que el mediador tiene la autorización para depositar en nombre del autor. (12)

El depósito en SWORD se realiza mediante un paquete que contiene el fichero que será depositado y los metadatos asociados a él (usando un formato basado en XML usando el estándar de metadatos Dublin Core). (12)

REST

Es un estilo de arquitectura para el diseño de aplicaciones en red. La idea es usar el protocolo HTTP para la comunicación entre las máquinas en vez de mecanismos más complicados como Común de Intermediarios en Peticiones a Objetos (CORBA), Llamada a Procedimiento Remoto (RPC) o Protocolo Simple de Acceso a Objetos (SOAP). REST ofrece ventajas como: (13)

- Independiente de la plataforma (no importa si el servidor es UNIX o si el cliente es Windows, etc.). (13)
- Independiente del lenguaje (aplicaciones en Java se pueden comunicar con otras en PHP, etc.). (13)
- Basado en estándares (se ejecuta sobre HTTP). (13)

REPXOS 3.0 contiene un módulo REST API que proporciona una interfaz de programación para comunidades, colecciones, ítems y bitstreams, lo que significa que es posible administrar no solo los ítems, sino también las comunidades y colecciones. El REST API de REPXOS 3.0 incluye autenticación, permite la creación, actualización y eliminación de objetos, acceso a materiales restringidos si estás autorizado y utiliza SSL. (13)

Fue seleccionado como método para el depósito el estilo de arquitectura REST debido a su independencia tanto del lenguaje como de la plataforma, además REST no mantiene estados, permitiendo un consumo menor de los recursos del servidor al realizar el depósito en lote de documentos. El uso de REST también permite el diseño de una aplicación más escalable, pensando en un futuro añadir más funcionalidades para administrar REPXOS.

1.7 Metodologías, tecnologías, herramientas y lenguajes utilizados

La selección de las metodologías, tecnologías, herramientas y lenguajes fue basada en las definidas para el desarrollo de los módulos y herramientas externas al sistema REPXOS, teniendo en cuenta que no afecta el ambiente de despliegue, ni cuestiones de mantenimiento.

Proceso Unificado Ágil (AUP)

Fases Variación AUP-UCI (14)

Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto. (14)

Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la transición se capacita a los usuarios finales sobre la utilización del software. (14)

Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto. (14)

Descripción de las disciplinas

AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno), se decide para el ciclo de vida de los proyectos de la UCI tener 8 disciplinas, pero a un nivel más atómico que el definido en AUP. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas. Se mantiene la disciplina Implementación, en el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación y la disciplina Despliegue se considera opcional. Las restantes 3 disciplinas de AUP asociadas a la parte de gestión para la variación UCI se cubren con las áreas de procesos que define CMMI-DEV v1.3 para el nivel 2, serían Gestión de la configuración (CM), Planeación de proyecto (PP) y Monitoreo y control de proyecto (PMC). (14)

Modelado de negocio (opcional): El Modelado del Negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito. (14)

Requisitos: El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del

Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el Modelado de negocio. (14)

Análisis y diseño: En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis. (14)

Implementación: En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema. (14)

Pruebas internas: En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible, componentes de prueba ejecutables para automatizar las pruebas. (14)

Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación. (14)

Pruebas de Aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido. (14)

Despliegue (Opcional): Constituye la instalación, configuración, adecuación, puesta en marcha de soluciones informáticas y entrenamiento al personal del cliente. (14)

Todas las disciplinas antes definidas (desde Modelado de negocio hasta Despliegue) se desarrollan en la Fase de Ejecución, de ahí que en la misma se realicen iteraciones y se obtengan resultados incrementales (14), ver Figura 1.

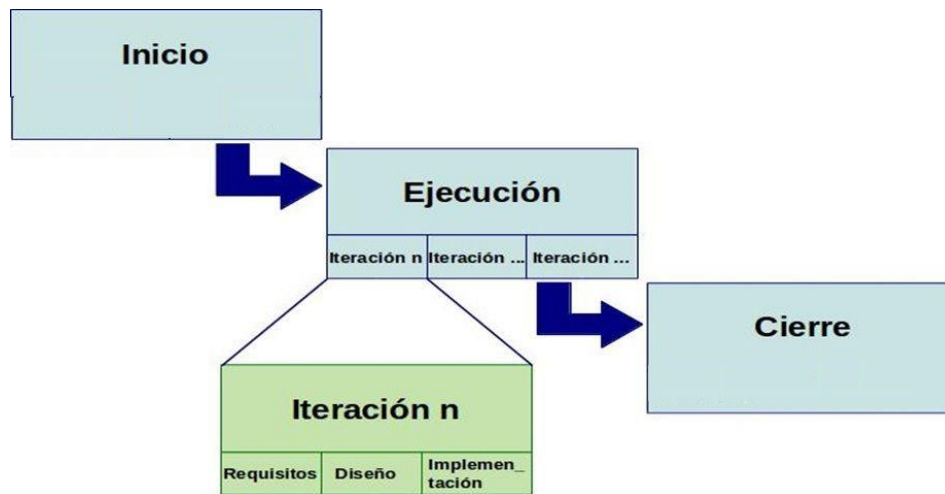


Figura 1: Fases e Iteraciones

El escenario de la disciplina Requisitos que aplica a DEPXOS es el Escenario No2, el mismo plantea:

Escenario No2: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información. (14)

Los proyectos que modelen el negocio con Modelo Conceptual solo pueden modelar el sistema con Casos de Uso del Sistema. (14)

Sistema Gestor de Base de Datos: PostgreSQL 9.4.6

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional y de código abierto, distribuido bajo la licencia BSD y con su código fuente disponible libremente. Se ejecuta en los sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc. Presenta documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios. Soporte nativo para los lenguajes: Java, PHP, C, C++, Perl, Python, etc. Soporte de todas las características de una base de datos profesional (triggers, procedimientos almacenados, funciones, secuencias, relaciones, reglas, tipos de datos definidos por usuarios, vistas, vistas materializadas, etc.). (15)

Entorno de Desarrollo Integrado NetBeans 8.0

Es un entorno de desarrollo gratuito y de código abierto para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para una gran variedad de lenguajes existentes como: Groovy, Java, HTML, PHP, C/C++, entre otros. Existe además un número importante de módulos para extender el NetBeans IDE. (16)

Algunas de sus características principales son:

Buen editor de código: Destaca código fuente sintáctica y semánticamente, permite refactorizar el código fácilmente, proporciona plantillas de código, consejos de codificación y generadores de código. El editor soporta varios lenguajes como Groovy, Java, C/C++, entre otros y puede ser extendido para ampliar el soporte a otros lenguajes.

Fácil y eficiente gestión de proyectos: Proporciona diferentes vistas para los datos, asistentes de ayuda para crear y gestionar proyectos de manera sencilla y soporte para herramientas de control de versiones como Subversion, Mercurial y Git.

Desarrollo rápido de interfaces de usuario: Permite el diseño de interfaces gráficas para Java, HTML5, C/C++ y PHP mediante el uso de editores y herramientas de arrastrar y soltar.

Lenguaje de programación Groovy v2.1.7

Groovy es un lenguaje de programación dinámico con tipado opcional y orientado a objetos para la plataforma Java inspirado en lenguajes como Python y Ruby. Posee características como: (17)

- Curva de aprendizaje corta.
- Es un lenguaje conciso, con una sintaxis expresiva y fácil de leer.
- Perfecta integración con Java y librerías de terceros.
- Posee características muy potentes como closures, meta-programación, programación funcional, inferencia de tipos y compilación estática.
- El bytecode generado por Groovy es totalmente compatible con el generado por Java.

Lenguaje de programación JavaScript:

Es un lenguaje de programación sencillo e interpretado por lo que no es necesario compilar un programa para ejecutarlo, utilizado para crear páginas web dinámicas, dependiendo totalmente del código HTML de la página. Fue desarrollado por la empresa Netscape a partir del lenguaje Java. (18)

Posee características como: (19)

- Permite a una página web reaccionar o responder directamente a la interacción del usuario con elementos de formulario.
- Procesamiento de los datos en el cliente antes de ser enviados al servidor.
- Cambio de contenido y estilo en los navegadores de forma dinámica y respuesta instantánea a la interacción del usuario.

Framework Grails v2.4.3

Es un framework de desarrollo web para la plataforma Java, libre y altamente productivo desarrollado sobre el lenguaje Groovy.

Principales características de Grails (20)

- Usa la filosofía convención sobre configuración de forma tal que las configuraciones durante el desarrollo son mínimas.
- Aplica el principio no te repitas (DRY) de forma tal que facilita al máximo la reutilización de código (en forma de servicios).
- Sigue una filosofía ágil, basado en un lenguaje dinámico como Groovy que permite ahorrar tiempo a la hora de programar.
- Tiene una base muy sólida, pues está desarrollado sobre tecnologías como Spring e Hibernate que son muy confiables y están probadas.

JavaScript Asíncrono y XML (AJAX)

Es una técnica de desarrollo web que genera aplicaciones web interactivas. Está formado por varias tecnologías como XHTML y CSS, para crear una presentación basada en estándares, DOM, para la interacción y manipulación dinámica de la presentación, XML, XSLT, y JSON, para el intercambio y la manipulación de información. Además XMLHttpRequest, para el intercambio asíncrono de información y JavaScript, para unir todas las demás tecnologías, por lo que es necesario para desarrollar aplicaciones AJAX dominar cada una de las tecnologías anteriores. (21)

JQuery

Es una librería JavaScript rápida y pequeña. Permite interactuar con documentos HTML de una forma muy simple, manipular el DOM, manejar eventos, realizar múltiples animaciones y una Interfaz de Programación de Aplicaciones (API) que simplifica el uso de AJAX) en múltiples navegadores. (22)

JQuery es liberado bajo la licencia Instituto Tecnológico de Massachusetts (MIT), lo cual permite usar el proyecto JQuery en cualquier otro proyecto, incluso de carácter comercial, siempre y cuando la cabecera de derechos de autor se deje intacta. (22)

jQuery Form Plugin

Permite actualizar fácil y discretamente formularios HTML para usar AJAX. Sus principales métodos son ajaxForm y ajaxSubmit, ellos recopilan información desde el elemento de formulario para determinar cómo manejar el proceso de envío. Ambos métodos soportan numerosas opciones, que permiten tener un control total sobre cómo son enviados los datos. (23)

Permite además mejorar la interacción del usuario con la aplicación, evitando las recargas constantes de la página, teniendo en cuenta que el intercambio de información con el servidor se ejecuta en segundo plano.

Lenguaje Unificado de Modelado (UML 2.1)

UML es un lenguaje estándar para especificar, visualizar, construir y documentar los artefactos de sistemas informáticos, así como para el modelado de negocios y otros sistemas. (24)

UML brinda variados elementos de esquematización para representar las diferentes partes de un sistema, de ellos se utilizan en el desarrollo de la aplicación los diagramas de casos de uso, de clases del diseño, de despliegue, de componentes, de colaboración y el resto de los diagramas.

Visual Paradigm 8.0

Es una herramienta profesional que se utiliza para el modelado, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones. Permite dibujar todos los tipos de diagramas de clases, ingeniería inversa, generar código desde diagramas y generar documentación. (25)

1.8 Conclusiones

Con la realización de este capítulo se han llegado a las siguientes conclusiones:

- Luego del estudio realizado se caracterizaron los sistemas informáticos y herramientas usadas en la extracción automática de metadatos de documentos digitales y se identificaron ventajas y desventajas.
- Se seleccionó la librería GROBID para la extracción automática de metadatos teniendo en cuenta que tiene como propósito el trabajo con publicaciones técnicas y científicas apoyado por la herramienta Apache Tika para la extracción de metadatos en formatos diferentes a PDF.
- Se realizó un análisis de las funcionalidades de REPXOS 3.0 para definir el método de depósito de los documentos procesados y los metadatos asociados seleccionándose REST puesto que consume pocos recursos y es independiente de la plataforma y del lenguaje de programación.
- El estudio de los métodos de validación de la calidad de los metadatos arrojó como el más idóneo el de completitud debido a su facilidad de implementación y a que es ideal para recursos que aún no se encuentran dentro del repositorio.
- Se caracterizó la metodología de desarrollo de software AUP, herramientas, tecnologías y lenguajes de desarrollo que conforman la línea base de la arquitectura para el desarrollo de la propuesta de solución.

Capítulo 2: Características de la herramienta para el procesamiento en lote de los documentos digitales para el sistema REPXOS.

El presente capítulo tiene como objetivo describir la solución propuesta para el sistema REPXOS. Se presenta el modelo de dominio donde se analizan las entidades y conceptos presentes en el contexto donde se desarrolla la propuesta de solución y se definen las relaciones existentes entre cada uno de estos. Se especifican los requisitos funcionales y no funcionales con los que debe cumplir el sistema, además de mostrar el diagrama de casos de uso del sistema, se describen los actores y los casos de uso del sistema.

2.1 Propuesta de Solución

La propuesta de solución nombrada DEPXOS es una aplicación web que se encarga del procesamiento en lote de documentos digitales para el sistema REPXOS 3.0. La aplicación permitirá a los administradores de REPXOS 3.0 la gestión de recursos digitales y la extracción de sus metadatos, que incluye su validación utilizando para esto el parámetro completitud, basado en pesos definidos previamente por el administrador.

Una vez extraídos los metadatos de los recursos, dichos recursos comienzan a ser tratados en el sistema como ítems. Los ítems pueden ser eliminados completamente del sistema o editar sus metadatos de manera opcional. Una vez listos, los administradores pueden realizar el depósito de los mismos de manera individual o todos a la vez, siempre seleccionando previamente a qué colección realizar el depósito.

La comunicación con REPXOS 3.0 se realiza a través del estilo de arquitectura REST, de esta forma la autenticación en DEPXOS se realiza usando las credenciales propias de los administradores de REPXOS y se garantiza que solo ellos puedan realizar el depósito.

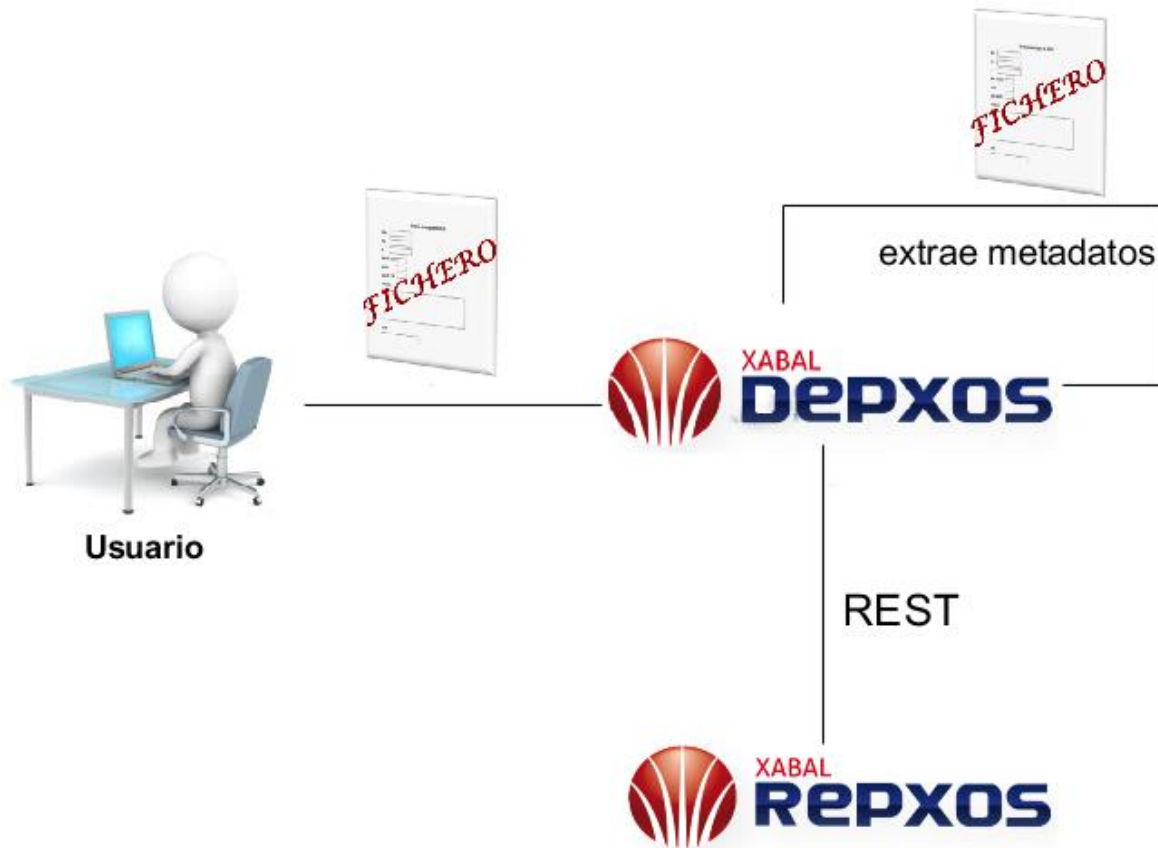


Figura 2: Propuesta de solución DEPXOS

2.2 Modelo de Dominio

El modelo de dominio es una representación de los conceptos reales que se relacionan con el proyecto y las relaciones existentes entre ellos. Se utiliza el término "de Dominio" para diferenciarlo del Modelo de Negocio debido a que es un concepto mucho más amplio. Se centra en una parte del negocio específicamente, la relacionada con el ámbito del proyecto, por lo que el término "dominio" representa una parte del "negocio". El Modelo de Dominio permite aumentar la comprensión del problema y contribuir a esclarecer la terminología o nomenclatura del dominio.

Al no poder identificarse claramente los procesos del negocio, solo los elementos conceptuales, se decide realizar la representación de los elementos a través de un modelo de dominio. El propósito es lograr que se comprenda mejor cómo ocurren los procesos, apoyados en la identificación de los requisitos funcionales. A continuación se muestra el modelo donde se describen los principales conceptos relacionados con el sistema, así como las relaciones entre ellos:

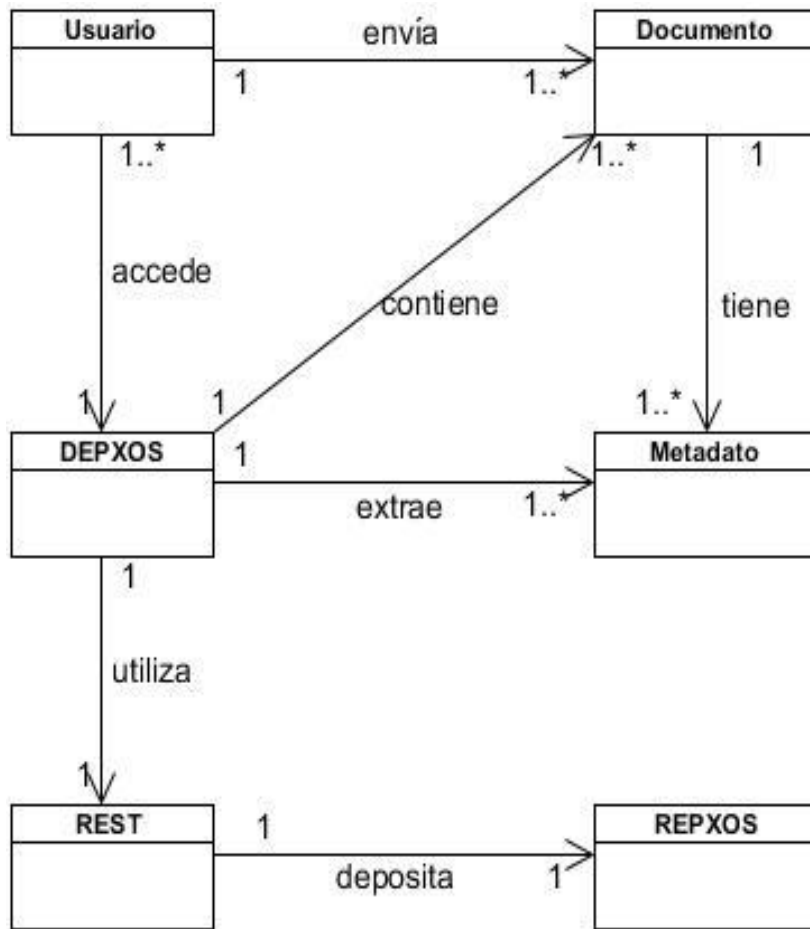


Figura 3: Modelo de dominio de DEPXOS

A continuación se muestra la definición de los conceptos del modelo del dominio

DEPXOS: es el sistema que permite gestionar los metadatos y depositar los recursos en REPXOS 3.0.

Documento: es el tipo de archivo que será depositado al sistema REPXOS 3.0.

Metadatos: datos que describen un documento.

REPXOS: sistema al que serán depositados los documentos digitales.

REST: servicio mediante el cual se conectará DEPXOS con el sistema REPXOS 3.0 para realizar el depósito.

Usuario: es la persona que interactúa con DEPXOS y sus funcionalidades.

2.3 Especificación de los requisitos de software

Durante el proceso de construcción de un software una de las actividades fundamentales es la captura de los requisitos del sistema. Estos son condiciones o capacidades que debe cumplir el software, por lo que es necesario que se verifiquen y validen evitando así errores en próximas etapas que afectarán el desarrollo y correcto funcionamiento del producto final.

2.3.1 Requisitos Funcionales

Los requisitos funcionales son declaraciones de los servicios que el sistema debe proporcionar o son descripciones de cómo se deben llevar a cabo algunos cálculos. (26)

La siguiente tabla muestra los requisitos funcionales identificados para el desarrollo de DEPXOS y la descripción de los mismos:

Requisito	Descripción
RF1 Autenticar Usuario	El usuario introduce su correo electrónico, contraseña y la URL de REPXOS 3.0 para acceder al sistema
RF2 Subir Fichero	Sube uno o múltiples ficheros a DEPXOS
RF3 Listar Fichero	Lista los ficheros existentes en DEPXOS
RF4 Eliminar Fichero	Elimina los ficheros en DEPXOS
RF5 Extraer Metadato	Extrae los metadatos de los ficheros y valida la calidad de los mismos
RF6 Listar Ítem	Lista los ítems existentes en DEPXOS
RF7 Eliminar Ítem	Elimina los ítems en DEPXOS
RF8 Editar Metadato	Edita los metadatos de un ítem en DEPXOS
RF9 Editar Opciones de Validación	Edita las opciones de validación de metadatos
RF10 Depositar Ítem en REPXOS	Realizar el depósito del ítem a REPXOS

Tabla 2: Descripción de los requisitos funcionales de DEPXOS

2.3.2 Requisitos no Funcionales

Los requisitos no funcionales restringen el sistema en desarrollo y el proceso de desarrollo que se debe utilizar. Pueden ser requisitos del producto, organizacionales o externos. A menudo están relacionados con las propiedades emergentes del sistema y, por lo tanto, se aplican al sistema completo. (26)

Eficiencia

RnF 1 Capacidad: El sistema soporta 100 usuarios conectados concurrentemente.

RnF 2 Recursos: Para el correcto funcionamiento del sistema se debe tener:

Servidor de Base de Datos

Procesador: Intel Core i3 a 3.20 Ghz

Memoria RAM: 4 Gb mínimo

Disco Duro: 80 Gb mínimo

Servidor de Aplicaciones

Procesador: Intel Core i3 a 3.20 Ghz

Memoria RAM: 4 Gb mínimo

Disco Duro: 80 Gb mínimo en dependencia de la cantidad de documentos que se desee procesar

El sistema donde se despliega la herramienta debe cumplir como mínimo con los recursos antes planteados. De lo contrario no se garantiza el correcto funcionamiento de la misma.

Restricciones del diseño

RnF 3 Lenguaje de programación: Se utiliza para la construcción del sistema los lenguajes de programación JavaScript y Groovy v2.1.7 y las herramientas que se utilizan son de distribución bajo licencias libres, para favorecer el desarrollo de nuevos módulos o subsistemas.

Interfaz

RnF 4 Interfaces de Hardware: El hardware donde se instaló el sistema posee una interfaz de red cuya velocidad de transferencia es de 100 Mbps. Para garantizar el correcto funcionamiento de la aplicación, se sugiere que se utilice una interfaz de red con la velocidad de transferencia antes mencionada.

RnF 5 Interfaces de Software: El sistema debe integrarse con el gestor de base de datos PostgreSQL 9.4.6 y el contenedor de servlets Apache Tomcat 7.0.56.

Seguridad

RnF 6 Seguridad del Sistema: El software está protegido contra accesos no autorizados. Se utilizan mecanismos de validación que ofrecen el cumplimiento de la seguridad del sistema: cuenta, contraseña y nivel de acceso, de manera que cada usuario pueda tener disponible solamente las opciones relacionadas con su actividad y tenga datos de acceso propios, garantizando así la confidencialidad.

2.4 Definición de los actores

Un actor presenta comportamientos, incluyendo el propio sistema que se está estudiando cuando solicita los servicios de otros sistemas, (...). Los actores no son solamente roles que juegan personas, sino también organizaciones, software y máquinas. (27)

En la herramienta solo interactúa un actor, el cual se define a continuación en la Tabla 3:

Actor	Objetivos
Usuario	<ul style="list-style-type: none">• Autenticar Usuario• Subir Fichero• Gestionar Fichero• Extraer Metadato

	<ul style="list-style-type: none">• Gestionar Ítem• Editar Opciones de Validación• Depositar Ítem en REPXOS
--	---

Tabla 3: Actores que intervienen

2.5 Definición de los casos de uso de DEPXOS

A partir del análisis de los requisitos funcionales del sistema se definieron los siguientes casos de uso:

- CU1: Autenticar Usuario
- CU2: Subir Fichero
- CU3: Gestionar Fichero
- CU4: Extraer Metadato
- CU5: Gestionar Ítem
- CU6: Editar Opciones de Validación
- CU7. Depositar Ítem en REPXOS

2.5.1 Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. (28)

El Diagrama de casos de uso del sistema se muestra en la siguiente figura:

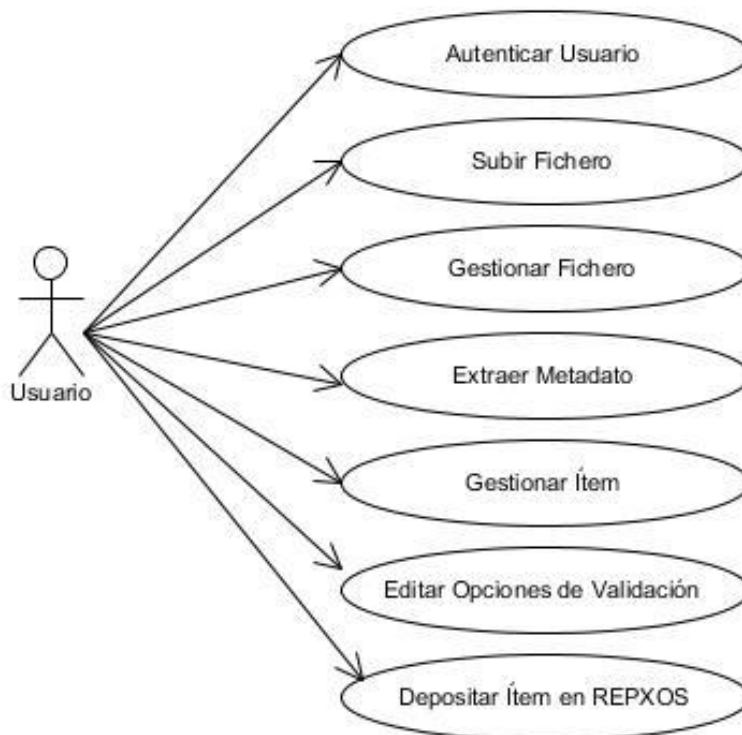


Figura 4: Diagrama de caso de usos de DEPXOS

2.5.2 Descripción de los casos de uso del sistema

A continuación se muestran la descripción del CUS Editar Opciones de Validación. Las demás descripciones se encuentran en el Anexo 2.

Objetivo	Editar las opciones de validación de los metadatos	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario escoge la opción de editar la validación de los metadatos, modifica algún campo si desea y termina cuando se muestra el mensaje de información “Opciones editadas correctamente”	
Complejidad	Baja	
Prioridad	Baja	
Precondiciones	El usuario está autenticado	
Postcondiciones	El usuario editó la validación de los metadatos correctamente	
Flujo de eventos		
Flujo básico Editar Opciones de Validación		
Actor		Sistema
1	Selecciona la opción Editar opciones de validación	
2		Muestra una interfaz que permite editar las opciones de validación de los metadatos
3	Edita los valores deseados y selecciona la opción Aceptar	
4		Valida los datos y muestra el mensaje de información “Opciones editadas correctamente”
Flujos alternos		
4a Valores inválidos		
Actor		Sistema
1		Muestra el mensaje de información "Los valores tienen que encontrarse entre 0 y 100"
4b Valores vacíos		
Actor		Sistema

1		Muestra el mensaje de información “Existen datos inválidos”
Relaciones	CU incluidos	-
Prototipo elemental de interfaz gráfica de usuario		

Tabla 4: DCU Editar Opciones de Validación

2.6 Conclusiones

Con la realización de este capítulo se han llegado a las siguientes conclusiones:

- Se analizaron y definieron las características a incorporarle a la solución.
- Se especificó el modelo de dominio que permitió definir los conceptos fundamentales del negocio, de esta forma se obtuvo una visión más clara del entorno sobre el cual se desarrolla el problema a resolver.
- Al identificar las características con las que debe contar la aplicación se obtuvieron los requisitos funcionales y los requisitos no funcionales.
- Se modeló el diagrama de casos de uso del sistema posibilitando un mejor entendimiento del proceso.
- Con las especificaciones de los casos de uso del sistema se establecieron los flujos básicos de estos y se logró una descripción detallada para una mejor comprensión del sistema que se implementó.

Capítulo 3: Arquitectura y diseño de la herramienta DEPXOS

En este capítulo se propone el diseño del sistema DEPXOS, a partir de los diagramas de clases del diseño empleando estereotipos web, los diagramas de colaboración, el diagrama de paquetes y el modelo de datos. Se describen los elementos de estos diagramas, así como el estilo arquitectónico y los patrones de diseño utilizados en la implementación de DEPXOS.

3.1 Modelo de diseño

Modelo de diseño es el conjunto de diagramas que describen el diseño lógico. Comprende los diagramas de clases software, diagramas de interacción, diagramas de paquetes, etcétera. (27)

3.1.1 Diagrama de Paquetes

Un diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones. Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema. Además, son buenos elementos de gestión, se usan en un modelo de desarrollo para agrupar elementos relacionados y cada uno de los paquetes se puede asignar a un individuo o a un equipo de desarrollo. (29)

A continuación se muestra el diagrama de paquetes de DEPXOS y su descripción:

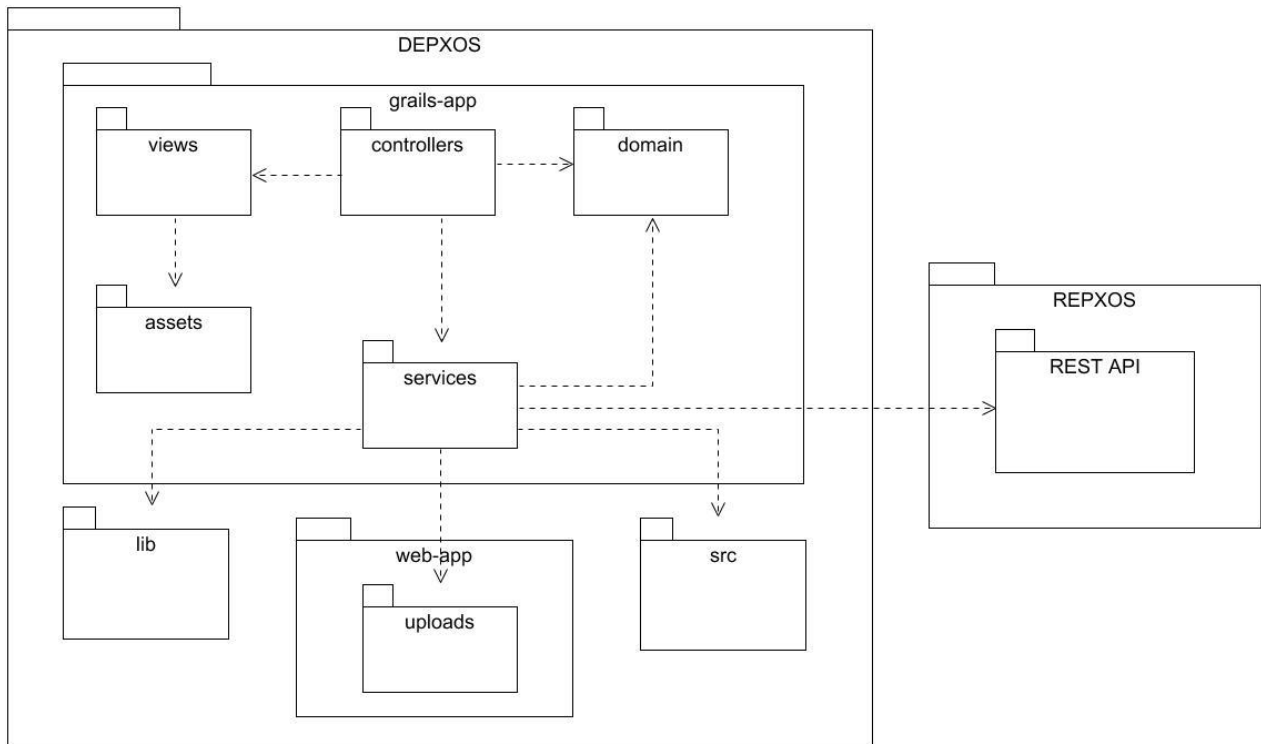


Figura 5: Diagrama de Paquetes

A continuación se muestra la descripción de los paquetes

assets: almacena los archivos css, js y las imágenes usadas en la aplicación.

controllers: se ubican los controladores de la aplicación.

domain: almacena las clases de domino que serán persistidas en la base de datos.

grails-app: contiene la mayor parte del código de la aplicación.

lib: contiene las librerías usadas en la aplicación.

REST API: contiene todas las clases relacionadas con el REST API de REPXOS.

services: contiene los servicios encargados de implementar la lógica de negocio de la aplicación.

src: contiene clases que se usan en la aplicación, pero no es necesario que sean persistidas.

uploads: almacena los ficheros subidos por los usuarios a la aplicación.

views: contiene las vistas de usuario de la aplicación.

web-app: almacena otros archivos css, js e imágenes.

3.2 Patrón Arquitectónico

Los patrones arquitectónicos son los relacionados con el diseño a gran escala, que se aplican típicamente durante las primeras iteraciones (la fase de elaboración) cuando se establecen las estructuras y conexiones más importantes. (27)

La arquitectura utilizada para la implementación de DEPXOS es la propuesta por el framework Grails. Dicho framework implementa el patrón Modelo-Vista-Controlador el cual establece que los componentes de un sistema de software se deben estructurar en tres capas distintas: la capa de acceso a datos, la capa de presentación y la capa de control. Grails engloba los controladores y las vistas en una misma capa, llamada capa web. Además introduce una nueva capa, llamada capa de servicios la cual se encarga de implementar la lógica del negocio y de esta forma liberar al controlador de tareas repetitivas, garantizando de esta forma una mayor capacidad de reutilización de código durante el desarrollo.

En el caso más simple, desde una vista el usuario envía los datos que desea procesar en un formulario, estos datos son capturados desde un controlador y enviados a un servicio el cual se encarga de procesarlos y persistirlos en la base de datos, ya sea añadir nuevos datos, consultarlos, editarlos o eliminarlos. Una vez procesados los datos el servicio envía la respuesta al controlador, el cual se encarga de enviarlos a la vista correcta. Por último, la vista es la encargada de dar formato a los datos y mostrarlos al usuario.

Esta arquitectura posee la capacidad de realizar modificaciones en una capa específica sin afectar a las restantes. A continuación se brinda una descripción detallada de las capas utilizadas para el desarrollo de la aplicación:

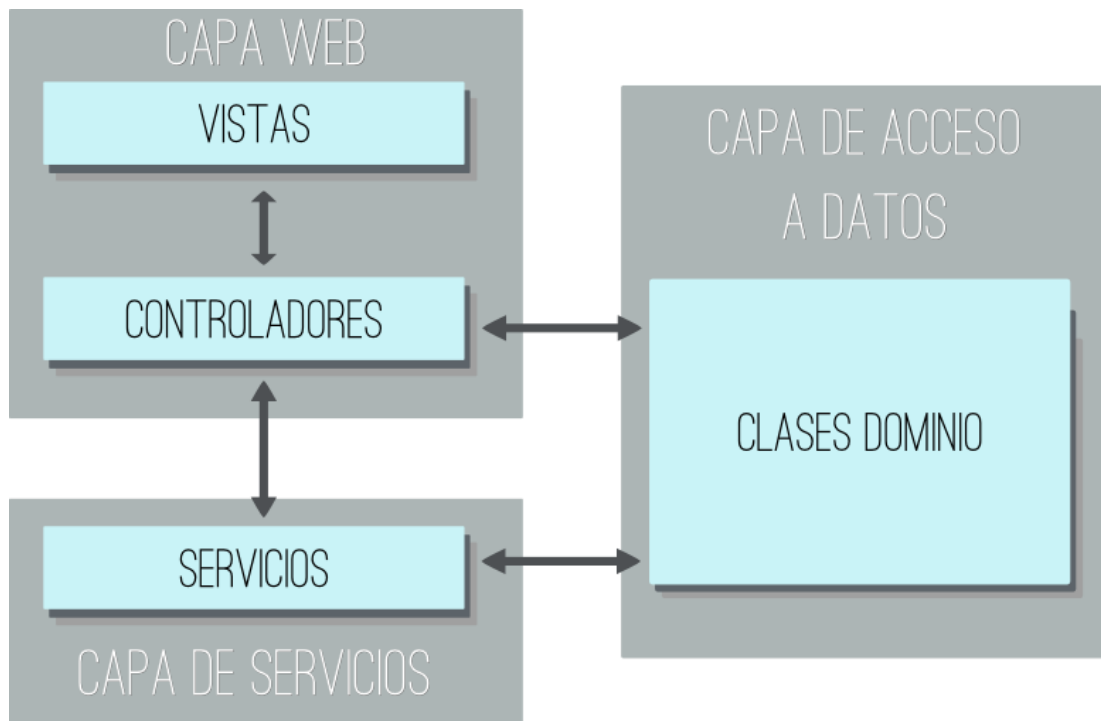


Figura 6: Arquitectura de una aplicación Grails

La capa web está formada por dos partes principales:

Las vistas: Son las responsables de presentar la información a los usuarios y están implementadas usando la tecnología Groovy Server Pages (GSP).

Los controladores: Son los encargados de establecer la comunicación entre las demás capas. Manejan las solicitudes provenientes desde la vista, realizan cambios en el modelo directamente o invocando un servicio y crean la respuesta, la cual puede ser generada en el mismo controlador o la puede delegar a una vista.

La capa de acceso a datos contiene los objetos que representan los datos en la aplicación. Estos pueden ser creados, leídos, actualizados y eliminados y son persistidos en una base de datos para garantizar que perduren en el tiempo. Para evitar el trabajo de forma directa con una base de datos y el uso de consultas SQL, Grails utiliza GORM, un motor de persistencia construido sobre Hibernate, de esta forma solo es necesario definir clases y relaciones entre ellas y GORM se encarga de realizar todo el trabajo de persistencia, mapeando estas clases en una base de datos.

La capa de servicios es la encargada de implementar la lógica de negocio de la aplicación, dejando a los controladores solo la responsabilidad de manejar el flujo de solicitudes. Por tanto, la función de esta capa es relacionarse directamente con el modelo de datos, realizar operaciones de lógica de negocio y devolver los datos al controlador.

3.3 Patrones de diseño utilizado en el desarrollo de DEPXOS

Un patrón de diseño es una buena práctica documentada de la solución de un problema que ha sido aplicado satisfactoriamente en múltiples entornos. Es una solución recurrente a un problema común observado o descubierto durante el estudio o construcción de numerosas aplicaciones. Su principal objetivo es incrementar la calidad del software en términos de reusabilidad, mantenimiento y extensibilidad. (27)

Inversión de Control (IoC)

La inversión de Control es un patrón utilizado en Grails, según el cual las dependencias de un componente no deben gestionarse desde el propio componente para que este sólo contenga la lógica necesaria para hacer su trabajo. (20)

Cuando se crea un componente en la aplicación, Grails configura Spring para que gestione su ciclo de vida (cuándo se crea, cuántas instancias se mantienen vivas a la vez, cómo se destruyen, etc.) y sus dependencias (qué otros componentes se necesitan para realizar su trabajo y cómo conseguirlos). El objetivo de esta técnica es mantener los componentes lo más sencillos que sea posible, incluyendo únicamente código que tenga relación con la lógica de negocio, así la aplicación será más fácil de comprender y mantener. (20)

El uso de este patrón se evidencia específicamente al usar un servicio dentro de un controlador. En este caso solo es necesario declarar el servicio, el programador no tiene que preocuparse por gestionar dicho servicio ni por las dependencias que este tenga, lo que permite un código más legible, fácil de mantener y menos propenso a errores.

GOF (Gang of Four)

Se divide en tres tipos de patrones de relevancia especial para el diseño orientado a objetos: patrones creacionales, estructurales y conductuales. (30)

Singleton: El patrón Singleton asegura que una clase tenga una sola instancia y proporciona un punto de acceso global a ella. Los servicios en Grails por defecto hacen uso de este patrón, de esta forma solo existe una instancia de los servicios durante la ejecución, logrando una reducción de uso de memoria y de tiempo de procesamiento (se elimina la sobrecarga de crear nuevos objetos).

GRASP

Describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades, expresados como patrones. (27)

Experto: La responsabilidad de realizar una labor específica es otorgada a la clase que contiene la información necesaria para realizar dicha labor. En la propuesta de solución un objeto de tipo File es el responsable de validar sus metadatos y exponer dicha evaluación a los demás objetos, pues la clase File es el experto en esa información.

Bajo Acoplamiento: El diseño se debe realizar de forma tal que la dependencia entre clases sea baja. De esta forma se tiene clases menos dependientes, logrando una reducción del impacto de los cambios y garantizando un mayor grado de reutilización.

Este patrón se pone en práctica al asociar a cada controlador un solo servicio, disminuyendo de esta forma el acoplamiento.

Alta Cohesión: Los elementos deben tener responsabilidades altamente relacionadas y efectuar la menor cantidad de operaciones posibles. Al existir una alta cohesión se hace más fácil el mantenimiento, entendimiento y la reutilización.

La utilización de este patrón se evidencia en los controladores, estos mantienen su código lo más simple posible manejando solo los eventos del sistema y delegando las demás tareas, como la implementación de la lógica de negocio a los servicios.

Controlador: Un Controlador es un objeto que no pertenece a la interfaz de usuario, responsable de recibir o manejar un evento del sistema. Un Controlador define el método para la operación del sistema. El controlador recibe la solicitud del servicio desde la capa de presentación y coordina su realización, normalmente delegando a otros objetos.

3.4 Diagrama de clases del diseño utilizando estereotipos web

Los diagramas de clases del diseño utilizando estereotipos web especifican la estructura de clases de un sistema, así como sus relaciones. Definen de forma correcta las relaciones de dependencia, generalización y asociación de clases que constituyen el sistema.

A continuación se muestra el diagrama de clases del diseño utilizando estereotipos web del CUS Editar Opciones de Validación. Los diagramas de los demás CUS se encuentran en el Anexo 3.

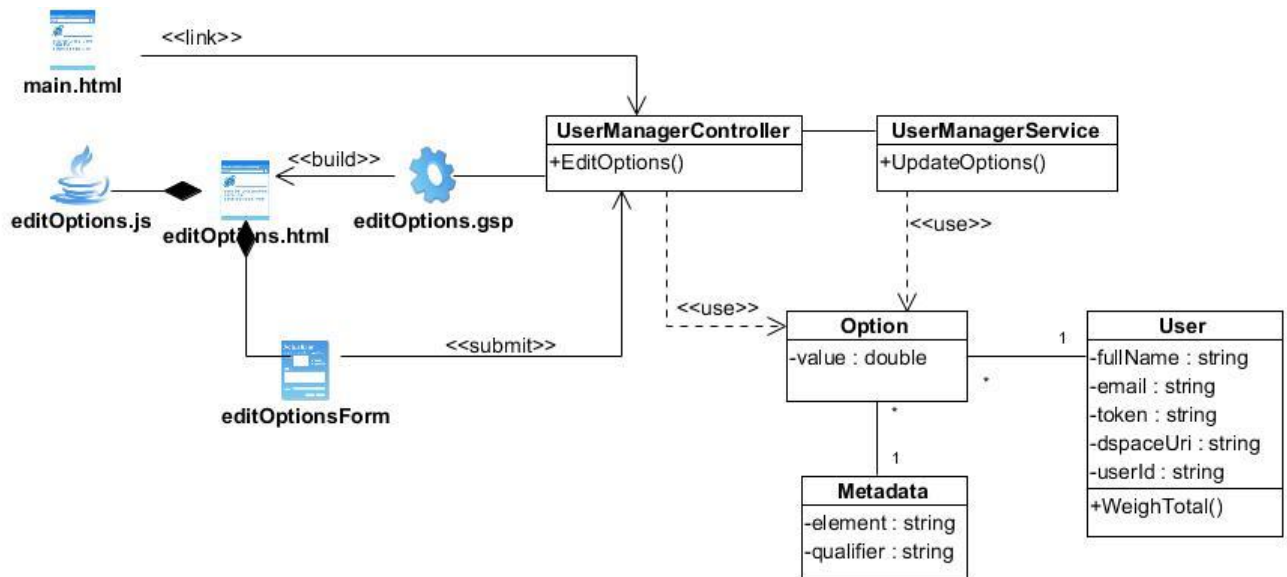


Figura 7: Diagrama de Clases del diseño Editar Opciones de Validación

3.4.1 Diagramas de Interacción

El diagrama de interacción proporciona la información para entender la dinámica del modo en el que se conectan y comunican los objetos entre las capas. Ilustran los escenarios más significativos desde el punto de vista de la arquitectura (en el sentido de que ilustran muchos aspectos de gran escala o grandes ideas del diseño). (27)

El UML define dos tipos de estos diagramas, ambos sirven para expresar interacciones semejantes o idénticas de mensaje: diagrama de colaboración y de secuencia.

Diagramas de Colaboración: Un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos (en cuanto a la interacción se refiere). A diferencia de los Diagramas de Secuencia, los Diagramas de Colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes se determinan explícitamente mediante números de secuencia. (31)

A continuación se muestra el diagrama de colaboración del CUS Editar Opciones de Validación, los diagramas de los demás CUS se encuentran en el Anexo 4.

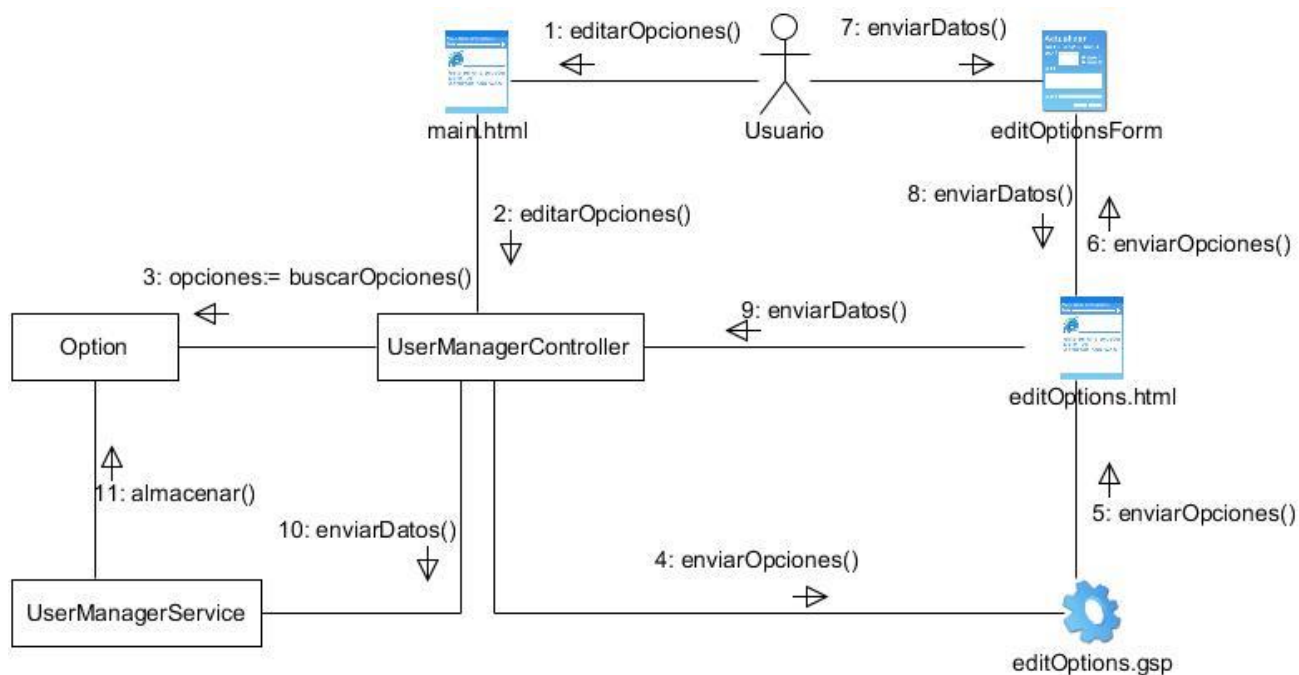


Figura 8: Diagrama de Colaboración Editar Opciones de Validación

3.5 Modelo de datos

El modelo de datos de DEPXOS ofrece una descripción abstracta sobre la representación de los datos en un Sistema Gestor de Bases de Datos. Este modelo contiene características propias de estos objetos y la forma en que están relacionados dichos objetos.

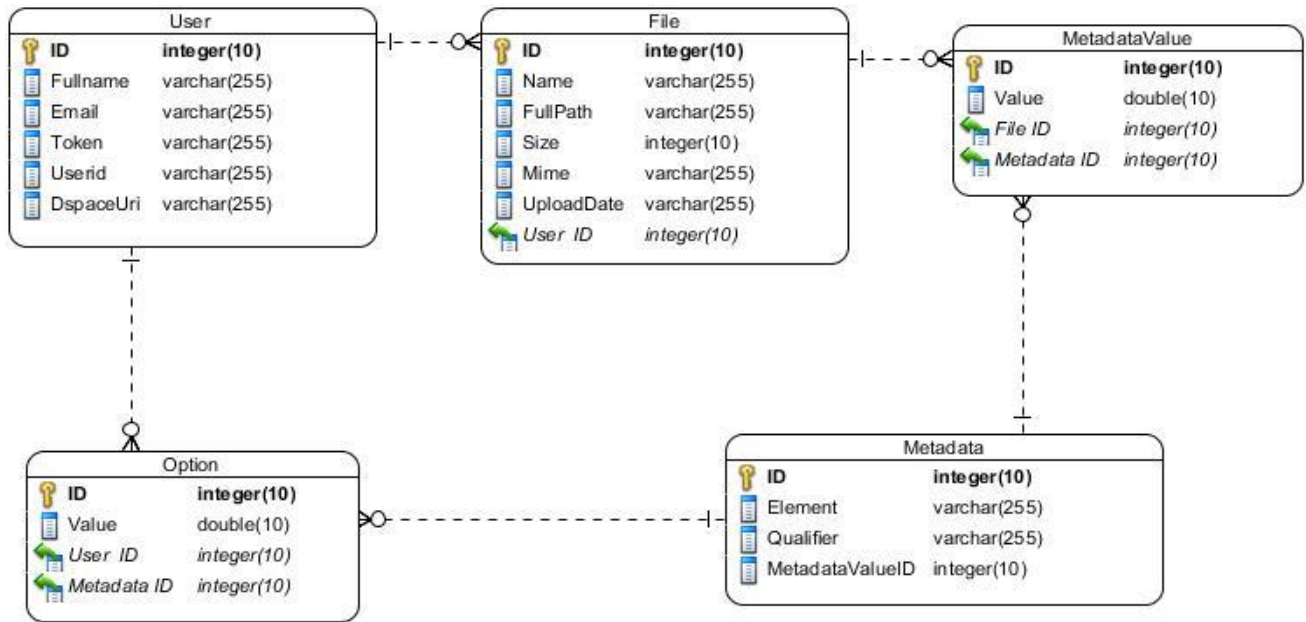


Figura 9: Modelo de datos

3.6 Conclusiones

Con la realización de este capítulo se han llegado a las siguientes conclusiones:

- Se mostraron un grupo de diagramas y una breve descripción de los patrones utilizados; Inversión de Control (IoC), GoF (Gang of Four) específicamente Singleton y GRASP como patrones de diseño y MVC (Modelo-Vista-Controlador) como patrón de arquitectura.
- Se realizaron los diagramas de colaboración, de paquete, de clase del diseño utilizando estereotipos web y el diagrama de modelo de datos, todo esto para posibilitar un mejor entendimiento del funcionamiento de DEPXOS.

Capítulo 4: Implementación y pruebas de la herramienta DEPXOS.

En el presente capítulo se propone explicar a partir de los resultados del diseño, la implementación del sistema DEPXOS. Como parte de la implementación se muestra el diagrama de componentes, el cual contribuye a la organización del subsistema y representa la vista estática de este, y el diagrama de despliegue en el que se visualiza la situación física del sistema. Se definen las pruebas de software y algunos de los resultados obtenidos de las mismas además de los estándares de codificación.

4.1 Modelo de Implementación

El modelo de implementación representa cómo los elementos del modelo de diseño se implementan en términos de componentes. De igual forma describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y la relación existente entre ellos. (32)

4.1.1 Diagrama de Componentes

En el diagrama de componentes, un componente representa un módulo físico de código o paquete, puede ser relacionado en un diagrama de componentes, el cual muestra varios componentes de un sistema, describiendo sus elementos físicos y sus dependencias. (32)

A continuación se muestra el diagrama de componentes de DEPXOS:

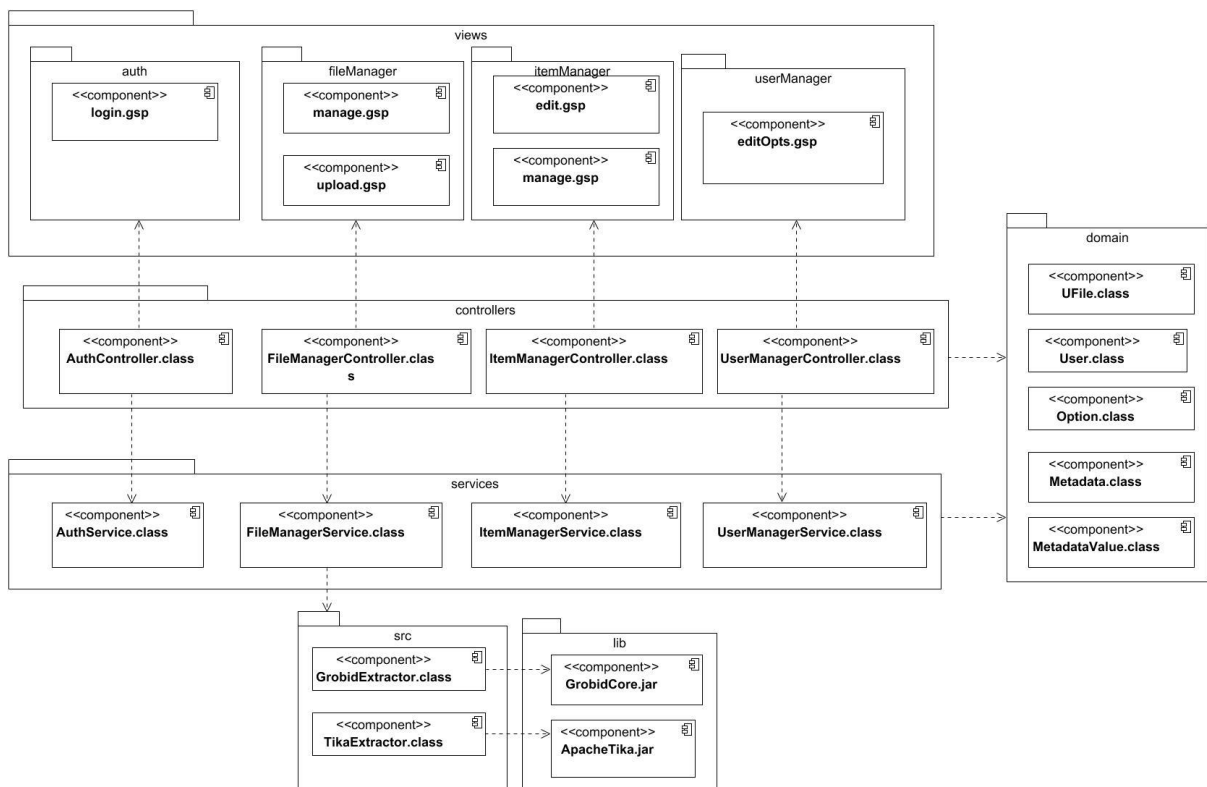


Figura 10: Diagrama de componentes

4.2 Estándares de codificación

Los estándares de codificación se definen con vistas a mejorar el entendimiento del código perteneciente a DEPXOS por otros desarrolladores y alcanzar una uniformidad en el mismo. A continuación se listan los elementos pertenecientes al estándar de codificación definido para el desarrollo de DEPXOS:

- Utilizar nombres en inglés para las clases y métodos.
- Dejar espacio de separación dentro del código, para que sea entendible.
- Utilizar el estilo de escritura CamelCase.
- Escribir en inglés los nombres de las variables.
- Los nombres de las variables deben ser cortos y significativos.

4.3 Modelo de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos. (33)

A continuación se muestra el diagrama de despliegue de DEPXOS para el sistema REPXOS 3.0:

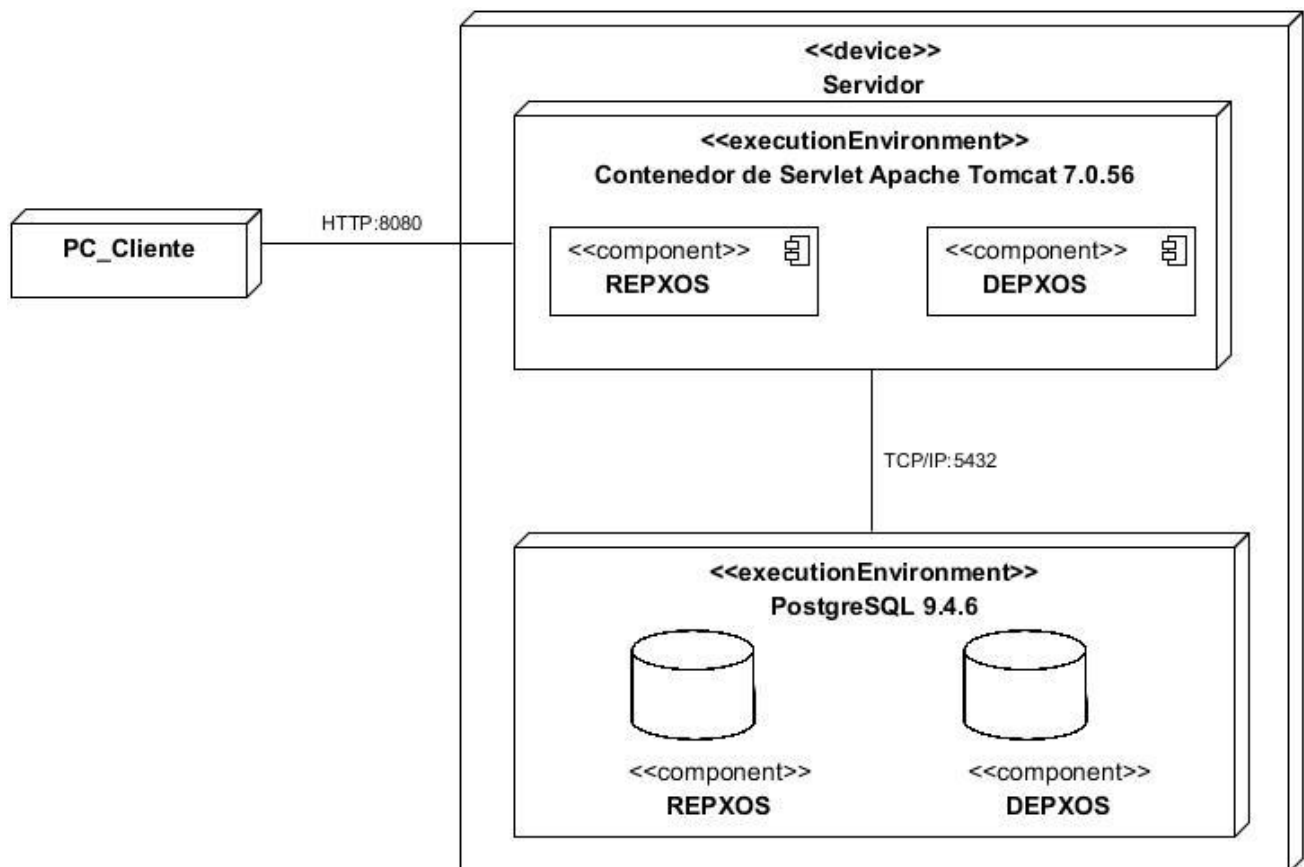


Figura 11: Diagrama de despliegue

4.3.1 Descripción de los protocolos de comunicación

En el diagrama de despliegue antes visto se utilizaron los protocolos de comunicación TCP/IP y HTTP debido a que DEPXOS se desarrolla sobre tecnología web. Se describe a continuación cada uno de los protocolos utilizados:

Familia de Protocolos TCP/IP: se utiliza para enlazar computadoras que usan sistemas operativos similares o diferentes, incluyendo PC, minicomputadoras y computadoras centrales sobre redes de área local (LAN). En el caso de los sistemas DEPXOS y REPXOS se utiliza para interconectarlos con el servidor de base de datos.

HTTP: protocolo de transferencia de hipertexto, es usado en cada transacción de la web. Define la sintaxis y semántica que utilizan los elementos software de la arquitectura web (cliente, servidor, proxy para comunicarse). En el sistema DEPXOS se utiliza este protocolo para el acceso de los clientes web a los servicios que brinda la aplicación.

4.4 Pruebas

Un instrumento adecuado para determinar el status de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante Técnicas de prueba. (34)

La prueba de software es imprescindible para garantizar la calidad de la aplicación representando una revisión final de las especificaciones del diseño y del código.

Elementos a tener en cuenta para que una prueba tenga éxito:

- Estrategia de prueba
- Niveles de prueba
- Tipo de prueba
- Método de prueba
- Caso de prueba

Estrategia de prueba

- Describe el enfoque y los objetivos generales de las actividades de prueba.
- Incluye los niveles de prueba a ser diseccionados, el tipo de prueba a ser ejecutada y los casos de prueba diseñados para lograr los objetivos.
- Define:

- Técnicas de pruebas (manual o automática) y herramientas a ser usadas.
- Criterios de éxitos y culminación de las pruebas.
- Consideraciones especiales relacionadas con los recursos necesarios para realizar las pruebas.

Una estrategia de prueba del software integra los métodos de diseño de caso de pruebas del software en una serie bien planeada de pasos que desembocará en la eficaz construcción del software. La estrategia proporciona un mapa que describe los pasos que se darán como parte de la prueba, indica cuando se planean y cuando se dan estos pasos, además de cuanto esfuerzo, tiempo y recursos consumirán. Por tanto, cualquier estrategia de prueba debe incorporar la planeación de pruebas, el diseño de casos de pruebas, la ejecución de pruebas y la recolección y evaluación de los datos resultantes. (30)

Una estrategia de prueba del software debe ser lo suficientemente flexible como para promover un enfoque personalizado. Al mismo tiempo, debe ser lo adecuadamente rígido como para promover una planeación razonable y un seguimiento administrativo del avance del proyecto. (30)

Niveles de prueba

La prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes niveles de pruebas: (35)

- Pruebas unitarias
- Prueba de integración
- Prueba de sistema
- Prueba de implantación
- Prueba de aceptación
- Prueba de regresión

Se estará trabajando en el nivel de sistema, con el objetivo de comprobar la funcionalidad de todo el sistema (35) y en el nivel de pruebas unitarias con el objetivo de probar el comportamiento de cada uno de los componentes de forma independiente. (35)

Tipos de prueba (35)

- Funcionalidad (función, seguridad, volumen, usabilidad)
- Fiabilidad (integridad, estructura, estrés)
- Rendimiento (benchmark, contención, carga, performance profile)
- Soportabilidad (configuración, instalación)

Se aplicaron pruebas de funcionalidad, y dentro de ellas de función. Estas pruebas fijan su atención en la validación de las funciones, métodos, servicios y casos de uso, (35) permitiendo comprobar el correcto funcionamiento de los requisitos funcionales de DEPXOS. Se realizaron además pruebas de rendimiento específicamente de carga, usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante, (35) esta última prueba fue realizada empleando la herramienta JMeter 2.11.

Casos de prueba

Los casos de prueba deben diseñarse para descubrir errores debidos a cálculos erróneos, comparaciones incorrectas o flujo de control inadecuado. Se diseñan para garantizar que: se satisfagan todos los requisitos de funcionamiento, se logran todas las características de comportamiento, todo el contenido es preciso y se presenta de manera adecuada, se logran todos los requisitos de rendimiento, la documentación es correcta y se satisfacen la facilidad de uso y otros requisitos. (30)

Métodos de Prueba

Caja Negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento, se enfocan en los requisitos funcionales del software; es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requisitos funcionales para un programa. (30)

Se basa en técnicas como:

Técnica de partición de equivalencia: es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos de los que pueden derivarse casos de prueba. Un caso de prueba ideal descubre de primera mano una clase de errores que de otro modo podrían requerir la ejecución de muchos casos de prueba antes de observar el error general. (30)

Técnica de análisis de valores límites: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables. (35)

Técnica de Grafos de causa-efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones. (35)

La técnica a emplear para desarrollar las pruebas de caja negra es la de partición de equivalencia teniendo en cuenta que es la técnica que utilizan los proyectos de la UCI y por las características antes mencionadas.

A continuación se muestra el caso de prueba de caja negra aplicadas al CUS Editar Opciones de Validación. Los casos de prueba de los demás CUS se encuentran en el Anexo 5.

Escenario	Descripción	type	language	contributor	identifier	rights	publisher	subject	coverage	creator	format	date	source	description	relation	title	Respuesta del sistema	Flujo central
EC 1.1 Editar Opciones de Validación	Permite al usuario editar opcionalmente las opciones de validación del recurso basado en sus metadatos	V 6	V 7	V 8	V 9	V 10	V 11	V 12	V 13	V 14	V 15	V 16	V 17	V 18	V 19	V 20	1- Brinda la posibilidad de editar las opciones de validación del recurso basado en sus metadatos, para ello puede modificar los valores de los siguientes campos: <ul style="list-style-type: none"> • type • language • contributor • identifier • rights • publisher • subject • coverage • creator • format • date • source • description • relation • title 2- Muestra el mensaje de información: "Opciones editadas correctamente"	1- Seleccionar del menú superior la opción "Opciones de validación" 2- Modificar opcionalmente los datos 3- Seleccionar la opción "Aceptar"
EC 1.2 Valores vacíos	Valida que no existan campos vacíos	I vacío	V 2	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	El sistema le asigna inicialmente valor 0 a cada campo. Muestra el mensaje de información: "Existen datos inválidos"	1- Modificar los datos y dejar algún campo vacío 2- Seleccionar la opción "Aceptar"
EC 1.3 Valores inválidos	Valida que no existan datos inválidos	I -5	V 2	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	V 3	Muestra el mensaje de información: "Los valores tienen que encontrarse entre 0 y 100"	1- Modificar los datos e incluir como mínimo un dato inválido 2- Seleccionar la opción "Aceptar"

Tabla 5: Editar opciones de validación

4.5 Resultados de las pruebas aplicadas a DEPXOS

En la siguiente tabla se exponen los resultados de las pruebas funcionales realizadas a DEPXOS, se desarrollaron 3 iteraciones. Se encontraron 7 no conformidades (NC), en la primera iteración, de ellas

2 fueron errores ortográficos, 4 errores de concordancia entre la aplicación y el diseño de los casos de prueba y 1 error de validación en los datos de entrada. Se corrigieron las NC detectadas.

En la segunda iteración fueron detectados 2 errores, 1 de ortografía y otro de correspondencia entre el sistema y la descripción de un caso de prueba. Se corrigieron las NC detectadas. Al realizar la tercera iteración, no se detectaron NC, quedando demostrado que los errores encontrados fueron resueltos en su totalidad.

No. de Iteraciones	Total de no conformidades	No Procede	Resueltas
1ra Iteración	7	1	6
2da Iteración	2	0	2
3ra Iteración	0	0	0

Tabla 6: Resultados de las pruebas de funcionalidad

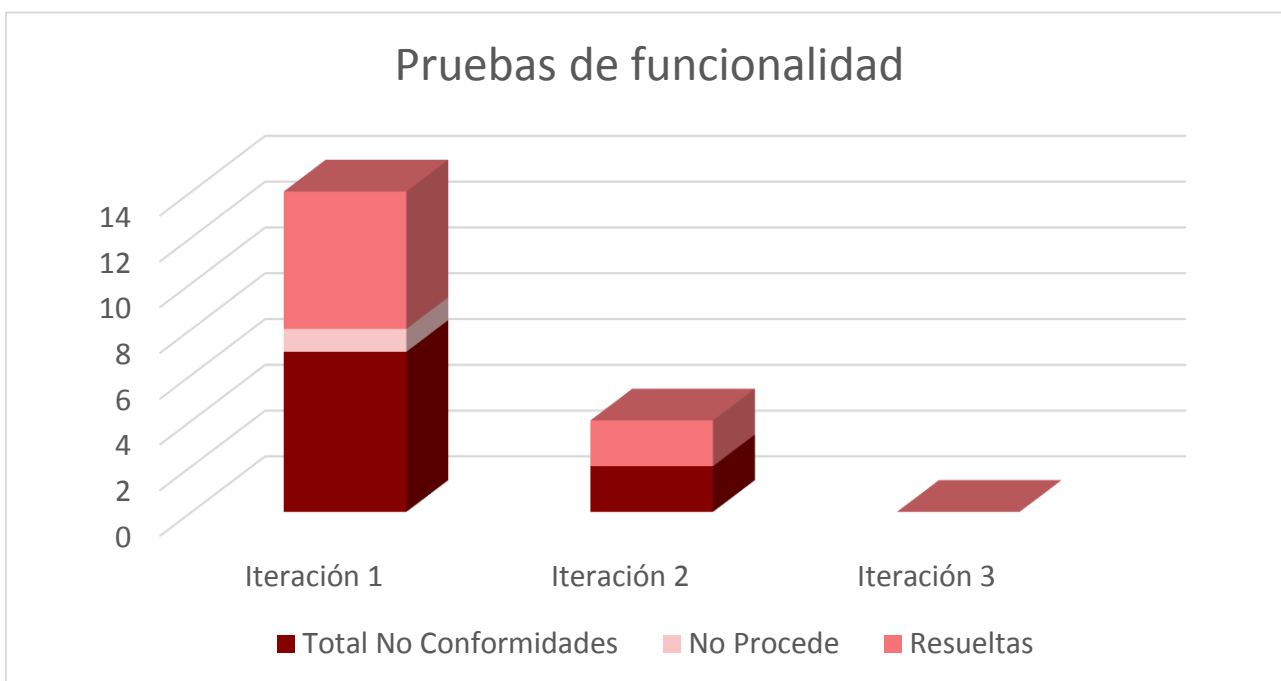


Figura 12: Resultados de las pruebas de funcionalidad

Pruebas de carga

Se realizaron pruebas de carga para 100 usuarios conectados concurrentemente manipulando un total de 50 documentos por cada uno de ellos. Aunque DEPXOS nunca va a tener concurrentemente esta cantidad de administradores trabajando queda demostrado que la aplicación es capaz de soportarlos. En estos casos se observa que las pruebas se han realizado con un 0.00% de errores.

Las pruebas fueron realizadas en una PC con requerimientos de 4GB de RAM y un procesador Intel CORE i3 de segunda generación. A continuación se muestra el reporte de las pruebas aplicadas con la herramienta JMeter:

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Media de By...
/depXos-1.0/	100	3126	138	4644	914,68	0,00%	19,0/sec	176,91	9545,2
/depXos-1.0/fileManager/upload	100	16703	904	21094	4617,32	0,00%	3,9/sec	1,29	335,6
/depXos-1.0/fileManager/extractAll	100	339715	202	356709	48814,25	0,00%	16,0/min	0,09	361,1
/depXos-1.0/itemManager/selectCollecti...	100	1794	12	5069	1744,31	0,00%	16,0/min	0,13	499,3
/depXos-1.0/itemManager/sendAll	100	30045	17	73637	26153,66	0,00%	13,7/min	0,08	361,1
Total	500	78277	12	356709	133459,27	0,00%	1,1/sec	2,45	2220,5

Figura 13: Reporte de las pruebas de carga

En cuanto a la funcionalidad Subir Ficheros la media de tiempo fue de 16,703 segundos y un rendimiento de 3,9 peticiones por segundo. Extraer Metadatos presentó una media de 5,66 minutos con un rendimiento de 16 peticiones por minuto. El depósito en REPXOS tardó una media de 30.045 segundos con un rendimiento de 13,7 peticiones por minuto. En general el flujo completo (Autenticar Usuario, Subir Ficheros, Extraer Metadatos y Depositar en REPXOS) demoró una media de 78,277 segundos con un rendimiento de 1,1 peticiones por segundo, demostrando que la aplicación es capaz de soportar 100 usuarios procesando 50 documentos concurrentemente.

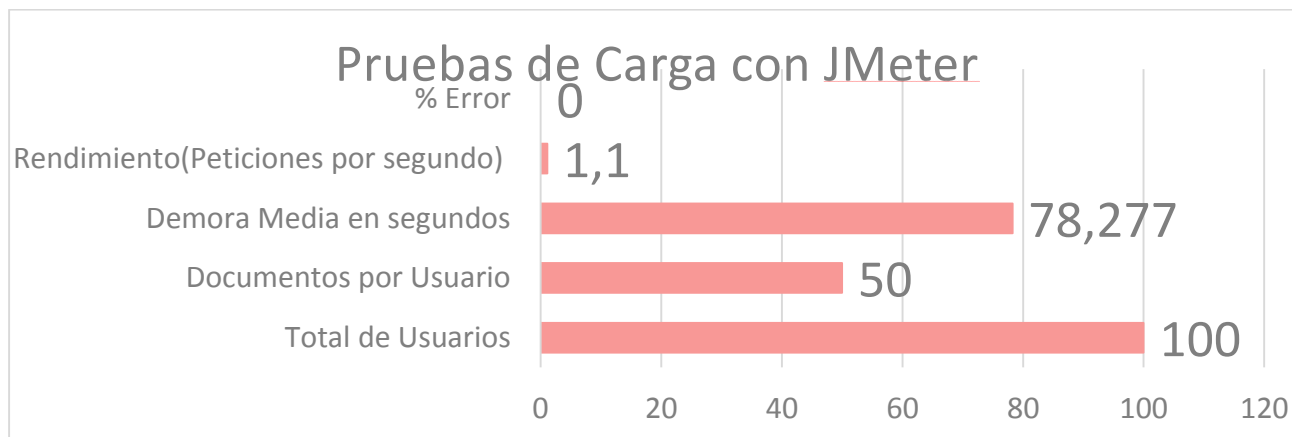


Figura 14: Resultados de las pruebas de carga

Pruebas unitarias

Se desarrollaron además pruebas unitarias, probándose el comportamiento de cada uno de los componentes de forma independiente. A continuación se muestran los resultados de las pruebas:



Unit Test Results - Summary

Executed 44 tests without a single error or failure!

Tests with failure and errors
Package summary
Show all tests

depxos
Executed 44 tests without a single error or failure!

- ✓ AuthControllerSpec
- ✓ AuthServiceSpec
- ✓ ErrorsControllerSpec
- ✓ FileManagerControllerSpec
- ✓ FileManagerServiceSpec
- ✓ ItemManagerControllerSpec
- ✓ ItemManagerServiceSpec
- ✓ UserManagerControllerSpec
- ✓ UserManagerServiceSpec

Figura 15: Pruebas unitarias realizadas a la herramienta DEPXOS



AuthControllerSpec

Package: depxos

AuthControllerSpec

Executed 7 tests without a single error or failure!

✓ La petición por GET a login() redirecciona a la vista login.gsp

Executed in 6.986 seconds.

✓ Un error en el commandObject de login() redirecciona a la vista login.gsp

Executed in 0.077 seconds.

✓ La petición de un usuario no autorizado es bloqueada

Executed in 0.701 seconds.

✓ El controlador actúa ante una página no encontrada

Executed in 0.039 seconds.

✓ El controlador actúa ante un error interno

Executed in 0.041 seconds.

✓ login() redirecciona a los errores en caso de perder conexión

Executed in 0.062 seconds.

✓ Al loguearse un usuario, redirecciona a upload.gsp

Executed in 0.096 seconds.

Figura 16: Pruebas unitarias realizadas al controlador AuthController.groovy

4.6 Conclusiones

Con la realización de este capítulo se han llegado a las siguientes conclusiones:

- En este capítulo se explicó a partir de los resultados del diseño, la implementación de la herramienta DEPXOS para el sistema REPXOS 3.0.
- Mediante el diagrama de componentes y el diagrama de despliegue quedó definida la distribución física y lógica de la arquitectura del sistema y sus conexiones.
- Se definió además los estándares de codificación y las pruebas de software.
- En las pruebas funcionales se encontraron en la primera iteración 7 NC, 1 no procedió y se resolvieron 6, en la segunda iteración se encontraron 2 y se resolvieron ambas y en la tercera no se encontraron NC.
- En las pruebas de carga en general el flujo completo (Autenticar Usuario, Subir Ficheros, Extraer Metadatos y Depositar en REPXOS) demoró una media de 78,277 segundos con un rendimiento de 1,1 peticiones por segundo, demostrando que la aplicación es capaz de soportar 100 usuarios procesando 50 documentos concurrentemente.
- Se realizaron además 44 pruebas unitarias exitosas.

CONCLUSIONES

El presente trabajo de diploma concluye con el cumplimiento del objetivo general propuesto desarrollándose una herramienta que permite realizar el procesamiento en lote de documentos digitales para su posterior depósito en el sistema REPXOS 3.0. De manera general se obtuvieron los siguientes resultados:

- Se caracterizaron las tendencias actuales a nivel mundial de la utilización de sistemas informáticos en la extracción automática de metadatos de documentos digitales identificando ventajas y desventajas de las mismas, lo cual permitió seleccionar dos herramientas para usar en la propuesta de solución: GROBID y Apache Tika.
- Se diseñó una solución que contribuyó al procesamiento en lote de los documentos digitales, que permite administrar dichos documentos, extraer sus metadatos y depositarlos en REPXOS 3.0.
- Se implementó la propuesta de solución haciendo uso de estándares y buenas prácticas.
- Se validó la solución aplicando los métodos de pruebas, lo que permitió la identificación de no conformidades existentes en la aplicación y su posterior corrección.

RECOMENDACIONES

Añadir a DEPXOS las funcionalidades correspondientes para la administración de comunidades, colecciones, ítems y bitstreams que proporciona el REST API de REPXOS 3.0

REFERENCIAS BIBLIOGRÁFICAS

1. DSpace, Equipo de desarrollo de. DSpace 5.x Documentation. [En línea] 2015. <https://wiki.duraspace.org/display/DSDOC5x>.
2. José A. Senso, Antonio de la Rosa Piñero. Scielo (Scientific Electronic Library Online). [En línea] <http://www.scielo.br/pdf/ci/v32n2/17038.pdf/>.
3. Carmine, Fernando Daniel. Ingesta asistida de contenidos en repositorios digitales - Un framework para DSpace. Diciembre 2014.
4. Informáticas (UCI), Universidad de las Ciencias. Catálogo productos y servicios. La Habana : s.n., 2015.
5. scholarlycommunication. [En línea] <http://scholarlycommunication.library.tamu.edu/repository-getting-started/policies/metadata-policy.html>.
6. CHRIS A. MATTMANN, JUKKA L. ZITTING. Tika in Action. NY 11964 : Manning Publications Co., 2012. ISBN 9781935182856.
7. Eleven Paths. [En línea] <https://www.elevenpaths.com/es/labstools/foca-2/>.
8. Zealand, Library National of. Metadata Extraction Tool. [En línea] 2003. [Citado el: 11 de febrero de 2016.] <http://www.natlib.govt.nz/services/get-advice/digital-libraries/metadata-extraction-tool>.
9. GROBID Documentation. [En línea] 2016. <http://grobid.readthedocs.org/en/latest/>.
10. Automatic evaluation of metadata quality in digital repositories. International Journal on Digital. Ochoa, Xavier y Duval, Erik.
11. The SWORD course - How SWORD Works.
12. Stuart Lewis, Leonie Hayes, Vanessa Newton-Wade, Antony Corfield, Richard Davis, Tim Donohue and Scott Wilson. If SWORD is the answer, what is the question?
13. Learn REST: A Tutorial. [En línea] miércoles, 13 de febrero de 2008.
14. PROGRAMA DE MEJORA. Metodología de desarrollo para la Actividad productiva de la UCI.
15. Informáticas, Comunidad de PostgreSQL Universidad de las Ciencias. PostgreSQL Cominidad Técnica de Desarrollo. [En línea] 2016. https://postgresql.uci.cu/?page_id=30.
16. NetBeans. [En línea] © 2016. [Citado el: 15 de febrero de 2016.] https://netbeans.org/index_es.html.
17. [En línea] © 2003-2016 the Groovy project. <http://www.groovy-lang.org/>.
18. Pérez, Javier Eguíluz. Introducción a JavaScript. 2009.
19. Danny Goodman, Michael Morrison. JavaScript Bible. Indianapolis, Indiana: Wiley Publishing, Inc, 2004. 0-7645-5743-2.
20. Brito, Nacho. Manual de desarrollo web con GRAILS. [En línea] 9 de junio de 2009. [Citado el: 21 de febrero de 2016.] <http://www.manual-de-grails.es>.
21. Pérez, Javier Eguíluz. Introducción a AJAX.

REFERENCIAS BIBLIOGRÁFICAS

22. Alvarez, Miguel Angel. Manual de jQuery.
23. malsup.com/jquery/form/. [En línea] <http://malsup.com/jquery/form/>.
24. Lenguaje Unificado de Modelado (UML). [En línea] <http://www.uml.org>.
25. Visual Paradigm. [En línea] <http://www.visual-paradigm.com>.
26. Sommerville, Ian. Ingeniería de Software. Séptima edición. México: Pearson Educación, 2005. ISBN: 84-7829-074-5.
27. Larman, Craig. UML y patrones. Introducción al análisis y diseño orientado a objetos. Mexico: Prentice Hall, 1999. ISBN 970-17-0261-1.
28. wikilearning. [En línea] 2007. http://www.wikilearning.com/tutorial/desarrollo_orientado_a_objetos_con_uml-%20diagrama_de_casos_de_uso/6321-5.
29. ooCities. [En línea] [Citado el: 23 de abril de 2016.] <http://www.oocities.org/es/monsalvelaura/fase2/analisis.html#5>.
30. Pressman, Roger S. INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO. Séptima edición. Impreso en México: MCGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V, © 2010. ISBN: 978-607-15-0314-5.
31. Xavier Ferré Grau, María Isabel Sánchez Segura. Desarrollo Orientado a Objetos con UML. Facultad de Informática–UPM, Artículo. 2001, Vol. 3.
32. Ivar Jacobson, Grady Booch y James Rumbaugh. El Proceso Unificado De Desarrollo de Software. Madrid: Addison Wesley, 2000. ISBN 84-7829-036-2.
33. Sparx Systems. [En línea] 2000-2016. [Citado el: 9 de abril de 2016.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
34. Pruebas de Software. [En línea] [Citado el: 9 de abril de 2016.] <http://www.pruebasdesoftware.com/laspruebasdesoftware.htm>.
35. Jorin, Ing. Michael González. Tesis de maestría. Proceso de pruebas para la liberación de productos software. Ciudad de La Habana: s.n., 2007.

BIBLIOGRAFÍA

1. DSpace, Equipo de desarrollo de. DSpace 5.x Documentation. [En línea] 2015. <https://wiki.duraspace.org/display/DSDOC5x>.
2. José A. Senso, Antonio de la Rosa Piñero. Scielo (Scientific Electronic Library Online). [En línea] [http://www.scielo.br/pdf/ci/v32n2/17038.pdf/..](http://www.scielo.br/pdf/ci/v32n2/17038.pdf/)
3. Carmine, Fernando Daniel. *Ingesta asistida de contenidos en repositorios digitales - Un framework para DSpace*. Diciembre 2014.
4. Informáticas(UCI), Universidad de las Ciencias. *Catálogo productos y servicios*. La Habana : s.n., 2015.
5. scholarlycommunication. [En línea] <http://scholarlycommunication.library.tamu.edu/repository-getting-started/policies/metadata-policy.html>.
6. CHRIS A. MATTMANN, JUKKA L. ZITTING. *Tika in Action*. NY 11964 : Manning Publications Co., 2012. ISBN 9781935182856.
7. Eleven Paths. [En línea] <https://www.elevenpaths.com/es/labstools/foca-2/>.
8. Zealand, Library National of. Metadata Extraction Tool. [En línea] 2003. [Citado el: 11 de febrero de 2016.] <http://www.natlib.govt.nz/services/get-advice/digital-libraries/metadata-extraction-tool>.
9. GROBID Documentation. [En línea] 2016 . <http://grobid.readthedocs.org/en/latest/>.
10. *Automatic evaluation of metadata quality in digital repositories. International Journal on Digital* . Ochoa, Xavier y Duval, Erik.
11. *The SWORD course - How SWORD Works*.
12. Stuart Lewis, Leonie Hayes, Vanessa Newton-Wade, Antony Corfield, Richard Davis, Tim Donohue and Scott Wilson. *If SWORD is the answer, what is the question?*
13. Learn REST: A Tutorial. [En línea] miércoles, 13 de febrero de 2008 .
14. *PROGRAMA DE MEJORA. Metodología de desarrollo para la Actividad productiva de la UCI*.
15. Informáticas, Comunidad de PostgreSQL Universidad de las Ciencias. *PostgreSQL Comunidad Técnica de Desarrollo*. [En línea] 2016. https://postgresql.uci.cu/?page_id=30.
16. NetBeans. [En línea] © 2016. [Citado el: 15 de febrero de 2016.] https://netbeans.org/index_es.html.
17. [En línea] © 2003-2016 the Groovy project. <http://www.groovy-lang.org/>.
18. Pérez, Javier Eguíluz. *Introducción a JavaScript* . 2009.

BIBLIOGRAFÍA

19. Danny Goodman, Michael Morrison. *JavaScript Bible*. Indianapolis, Indiana : Wiley Publishing, Inc, 2004. 0-7645-5743-2.
20. Brito, Nacho. Manual de desarrollo web con GRAILS. [En línea] 9 de junio de 2009. [Citado el: 21 de febrero de 2016.] <http://www.manual-de-grails.es>.
21. Pérez, Javier Eguíluz. *Introducción a AJAX*.
22. Alvarez, Miguel Angel. *Manual de jQuery*.
23. malsup.com/jquery/form/. [En línea] <http://malsup.com/jquery/form/>.
24. Lenguaje Unificado de Modelado (UML). [En línea] <http://www.uml.org..>
25. Visual Paradigm. [En línea] <http://www.visual-paradigm.com..>
26. Sommerville, Ian. *Ingeniería de Software. Séptima edición*. México : Pearson Educación, 2005. ISBN: 84-7829-074-5.
27. Larman, Craig. *UML y patrones. Introducción al análisis y diseño orientado a objetos*. Mexico : Prentice Hall, 1999 . ISBN 970-17-0261-1.
28. [wikilearning](http://www.wikilearning.com/tutorial/desarrollo_orientado_a_objetos_con_uml-%20diagrama_de_casos_de_uso/6321-5..). [En línea] 2007. http://www.wikilearning.com/tutorial/desarrollo_orientado_a_objetos_con_uml-%20diagrama_de_casos_de_uso/6321-5..
29. [ooCities](http://www.oocities.org/es/monsalvelaura/fase2/analisis.html#5). [En línea] [Citado el: 23 de abril de 2016.] <http://www.oocities.org/es/monsalvelaura/fase2/analisis.html#5>.
30. Pressman, Roger S. *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO. Séptima edición*. Impreso en México : McGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V, © 2010. ISBN: 978-607-15-0314-5.
31. Xavier Ferré Grau, María Isabel Sánchez Segura. Desarrollo Orientado a Objetos con UML. *Facultad de Informática–UPM, Artículo*. 2001, Vol. 3.
32. Ivar Jacobson, Grady Booch y James Rumbaugh. *El Proceso Unificado De Desarrollo de Software*. Madrid : Addison Wesley, 2000. ISBN 84-7829-036-2.
33. Sparx Systems. [En línea] 2000-2016. [Citado el: 9 de abril de 2016.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
34. Pruebas de Software. [En línea] [Citado el: 9 de abril de 2016.] <http://www.pruebasdesoftware.com/laspruebasdesoftware.htm>.

BIBLIOGRAFÍA

35. Jorin, Ing. Michael González. Tesis de maestría. Proceso de pruebas para la liberación de productos software. Ciudad de La Habana : s.n., 2007.
36. Rumbaugh, James , Jacobson, Ivar y Booch, Grady. *El Lenguaje Unificado de Modelado*.
37. CodeJobs. [En línea] 30 de marzo de 2013. <https://www.codejobs.biz/es/blog/2013/03/30/que-es-groovy>.
38. Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x. [En línea] [Citado el: 7 de Febrero de 2016.] <http://book.cakephp.org/2.0/es/cakephp-overview/understanding-model-view-controller.html>.
39. Microsoft Word - Dinámicas.doc - get.php. [En línea] [Citado el: 15 de abril de 2016.] <http://www.lsi.us.es/docencia/get.php?id=361>.
40. Significados. [En línea] 2013-2016. <http://www.significados.com/item/>.
41. TuxNots. [En línea] [Citado el: 18 de abril de 2016.] <https://sites.google.com/site/tuxnots/materias-de-la-facu/metodologia-de-sistemas/patronesgrasppatronesgofdiferenciaentregraspygof>.
42. Blanco Bueno, Carlos. Tema01. Construcción y Pruebas de Software. [En línea] [Citado el: 15 de abril de 2016.] <http://ocw.unican.es/enseñanzas-tecnicas/ingenieria-del-software-ii/materiales/tema1-pruebasSistemasSoftware.pdf>.
43. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Desing Patters. Elements of Reusable Object-Oriented Software*. s.l. : KevinZhang, Agosto de 1997.
44. Gataminza, Félix del Valle. *Documentos digitales: hipertexto y edición digital*. Madrid : Universidad de Complutense, 2006.
45. Lapuente, María Jesús Lamarca. hipertexto. [En línea] 2013 de 12 de 8. http://www.hipertexto.info/documentos/meta_html.htm.
46. Luis Carlos Alvarez Fernández, Javier Ruiz García. *Repositorio Institucional de la Universidad de las Ciencias Informáticas: Actualidad y Proyecciones*. Habana : s.n., 2012.
47. Madrid, Universidad Carlos III de. Metadatos Recuperación y Acceso a la Información. [En línea] 2010. [Citado el: 11 de febrero de 2016.] <http://www.metadatos-xmlrdf.com/metadatos/dublin-core..>
48. Marcello Visconti, Hernan Astudillo. Patrones.pdf. [En línea] 25 de Marzo de 2015. <http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf>.
49. Maven, Apache. The Apache Software Foundation. [En línea] 21 de enero de 2012. <http://tika.apache.org/1.2/formats.html..>

BIBLIOGRAFÍA

50. Patricia Testa, Paula Ceriotto. *Descripción de objetos digitales: metadatos*. Buenos Aires, Argentina : s.n.
51. Pavón Mestras, Juan. *Patrones de diseño orientado a objetos*.
52. REBIUM. CRUE REBIUM Red de Bibliotecas Universitarias Españolas. [En línea] 1994. http://www.rebiun.org/opencms/opencms/handle404?exporturi=/export/docReb/informe_jordi_prats.doc..
53. Sebastian, Juan. Actualidad Geek(Framework). [En línea] 27 de septiembre de 2015. http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.
54. Software, Facultad de Informática - Unidad Docente de Ingeniería de. *Patrones del Gang of Four*. España : Unidad Politécnica de Madrid. .
55. Visconti, Marcello y Astudillo, Hernan . Patrones.pdf. [En línea] 25 de Marzo de 2015. <http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf>.

ANEXOS

Anexo 1: Registro de metadatos Dublin Core por defecto

Elementos	Cualificadores	Descripción
Contributor		Una persona, organización o servicio responsable del contenido del recurso. En general todos los contribuidores no especificados
Contributor	advisor	Usado principalmente para el asesor de las tesis
Contributor	author	Autor del trabajo (Usado por defecto)
Contributor	editor	Persona u organización que contribuye a un recurso mediante la revisión
Contributor	illustrator	Persona u organización que contribuye a un recurso complementando el contenido primario con dibujos, diagramas, fotografías, etc.
Contributor	other	Otros contribuidores no especificados
Coverage	spatial	Las características espaciales del contenido
Coverage	temporal	Las características temporales del contenido
Creator		Puede ser utilizado como una alternativa a "contributor.author"
Date		Una fecha asociada a un evento en el ciclo de vida de un ítem Utilizar la forma calificada si es posible
Date	accessioned	Fecha en que DSpace toma posesión del ítem
Date	available	Fecha o rango de fechas en que el ítem se puso a disposición del público
Date	copyright	Fecha de copyright
Date	created	Fecha de creación o fabricación de contenido intelectual
Date	issued	Fecha de publicación o distribución
Date	submitted	Fecha recomendada para tesis y disertaciones
Identifier		Alcance general para los identificadores inequívocos que no se definen por la forma calificada
Identifier	citation	Estándar de cita bibliográfica para el ítem
Identifier	govdoc	Un número de un documento del gobierno

Identifier	isbn	Estándar internacional para número de libro
Identifier	issn	Estándar Internacional para número de serie.
identifier	sici	Artículo sucesivo del ítem e identificador de la contribución
identifier	ismn	Estándar Internacional de número de música
identifier	other	Es un tipo de identificador conocido común a una colección local
identifier	uri	Identificador de Recurso Uniforme
description		Campo para cualquier descripción no definida por los cualificadores
description	abstract	Una descripción textual del recurso, tal como un resumen en el caso de un documento o una descripción del contenido en el caso de un documento visual
description	provenance	Registra un suceso de administración del ciclo de vida del elemento en el repositorio. Es generado automáticamente por DSpace
description	sponsorship	Información sobre los organismos patrocinadores, individuos o acuerdos contractuales para el ítem
description	statementofresponsibility	Declaración de responsabilidad sobre la preservación de los registros MARC
description	tableofcontents	Tabla de contenido para un determinado ítem
description	uri	Identificador de Recurso Uniforme indicando la descripción del artículo
format		Campo para cualquier información de formato no definido por los cualificadores
format	extent	Tamaño o duración
format	medium	Medio físico
format	mimetype	Tipo MIME asociado con un archivo contenido en el artículo
language		El idioma del artículo. Se recomienda el uso de dc.language.iso
language	iso	Norma ISO actual de idioma del contenido intelectual, incluidos los códigos de país (por ejemplo, "en_US")


publisher		Entidad responsable de la publicación, distribución o impresión; o editor de la instancia dictada con anterioridad a la obra
relation		Hace referencia a un elemento relacionado. Se recomiendan los campos cualificados
relation	isformatof	Hace referencia a una forma física adicional
relation	ispartof	Hace referencia a un elemento que contiene física o lógicamente
relation	ispartofseries	Nombre de serie y número de serie del artículo
relation	haspart	Hace referencia a un elemento contenido física o lógicamente
relation	isversionof	Hace referencia a una versión anterior
relation	hasversion	Hace referencia a una versión posterior
relation	isbasedon	Hacer referencia a la fuente original del artículo
relation	isreferencedby	Identifica un recurso que hace referencia el artículo
relation	requires	El recurso referido es requerido para apoyar una función, la coherencia o alguna actividad en particular
relation	replaces	Un recurso relacionado que es sustituido, desplazado o sustituido por el recurso descrito
relation	isreplacedby	Un recurso relacionado que suplanta, desplaza o sustituye el recurso descrito
relation	uri	Referencia el Identificador de Recurso Uniforme para el artículo relacionado
rights		Condiciones que rigen el uso y la reproducción
rights	uri	Referencia términos que rigen el uso y la reproducción
source		Un recurso relacionado a partir del cual se deriva el recurso descrito. No usar; sólo con los metadatos recolectados
source	uri	No usar; sólo con los metadatos recolectados
subject		Una palabra clave o frase que describe el contenido del artículo
subject	classification	Número de clasificación local o número de clasificación no definido por otro campo
subject	ddc	Número de clasificación decimal Dewey
subject	lcc	Número de Clasificación de la biblioteca del congreso

subject	lcsh	Encabezados de la biblioteca del congreso
subject	mesh	Encabezados de Temas Médicos
subject	other	Vocabulario controlado local; los vocabularios globales recibirán un cualificador específico
title		Título principal del artículo
title	alternative	Título alternativo para el elemento, como una traducción o forma abreviada del título
type		Naturaleza o género del contenido. Ejemplo: Presentación, Grabación, Software, Reporte técnico, Tesis, Vídeo; Hoja de trabajo, etc.

Anexo 2: Descripción de los CUS

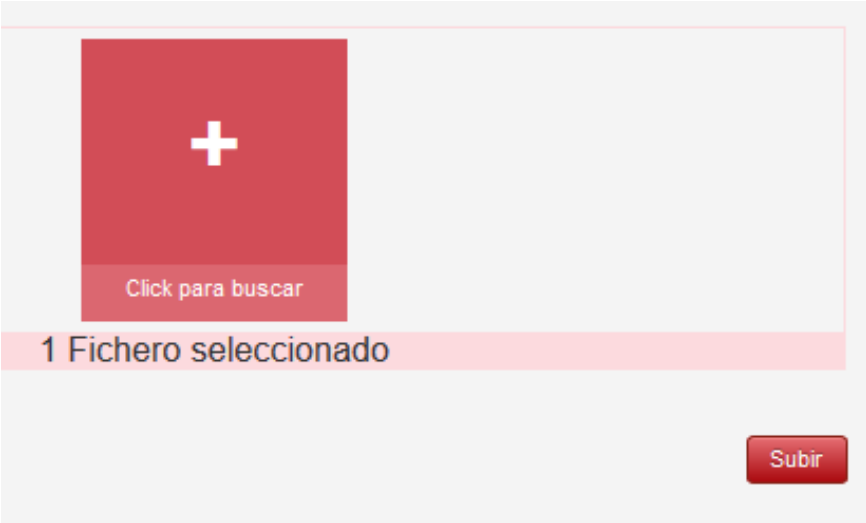
Anexo 2.1: Autenticar Usuario

Objetivo	Autenticar el usuario	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario introduce correo electrónico, contraseña y URL de REPXOS	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El sistema debe estar ejecutándose	
Postcondiciones	El usuario se ha autenticado correctamente en el sistema	
Flujo de eventos		
Flujo básico Autenticar Usuario		
Actor	Sistema	
1.	Introduce los datos correo electrónico, contraseña y URL de REPXOS y selecciona el botón "Entrar"	
2.		Valida los datos. Muestra la interfaz Subir Ficheros
3.		Termina el caso de uso

Flujos alternos	
2a Campos vacíos	
Actor	Sistema
1	Muestra el mensaje de información "El campo x no puede ser nulo" (x es el nombre del campo)
2b Datos incorrectos	
2b1a Campo URL de REPXOS incorrecto	
Actor	Sistema
1	Muestra el mensaje de información "REPXOS no disponible"
2a1b Campo correo electrónico o contraseña incorrecto	
Actor	Sistema
1	Muestra el mensaje de información "Usted no tiene acceso al recurso solicitado"
Prototipo elemental de interfaz gráfica de usuario	
	

Anexo 2.2: Subir Fichero

Objetivo	Subir los ficheros a DEPXOS
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario escoge los ficheros que desea subir a DEPXOS y termina cuando los ficheros se han subido a DEPXOS correctamente

Complejidad	Media	
Prioridad	Alta	
Precondiciones	El usuario está autenticado	
Postcondiciones	El usuario subió los ficheros a DEPXOS satisfactoriamente	
Flujo de eventos	Flujo básico Subir Ficheros	
Actor	Sistema	
1	Selecciona la opción subir ficheros	
2		Muestra una interfaz que permite seleccionar ficheros
3	Selecciona los ficheros que va a subir	
4		Muestra una interfaz con la cantidad de ficheros seleccionados y la opción Enviar
5	Selecciona la opción Enviar	
6		Sube los ficheros al servidor y muestra un mensaje de información "Ficheros enviados correctamente"
Relaciones	CU incluidos	-
Prototipo elemental de interfaz gráfica de usuario		
		

Anexo 2.3: Gestionar Fichero

Objetivo	Gestionar los Ficheros en DEPXOS	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario desea listar o eliminar los ficheros que se encuentran en DEPXOS y termina cuando el usuario ha listado o eliminado los ficheros correctamente	
Complejidad	Media	
Prioridad	Media	
Precondiciones	El usuario ha sido autenticado y ha subido como mínimo un fichero	
Postcondiciones	Se lista o elimina uno o más ficheros	
Flujo de eventos		
Flujo básico Gestionar Ficheros		
Actor	Sistema	
Indica que desea listar o eliminar los ficheros.	a) Si el usuario decide listar los ficheros Ver Sección 1: Listar Ficheros b) Si el usuario decide eliminar los ficheros Ver Sección 2: Eliminar Ficheros	
Sección 1: Listar Fichero		
Flujo básico Listar Fichero		
Actor	Sistema	
1	Selecciona la opción Administrar Ficheros	
2		Muestra en la interfaz Administrar Fichero la tabla listado de ficheros con todos los ficheros existentes en el servidor pertenecientes a ese usuario, de los ficheros se muestran los datos: Fichero (nombre), Tamaño, Tipo, Fecha y Opciones. Las opciones que se brindan son: <ul style="list-style-type: none"> • eliminar cada fichero (ver el flujo alternativo 2a eliminar fichero) • extraer metadatos de cada fichero (ver el flujo alternativo 2b extraer metadatos)
Flujos alternos		

2a Eliminar Fichero		
Actor		Sistema
1		Muestra el mensaje de confirmación "¿Seguro que desea eliminar el elemento?"
2	Presiona el botón Eliminar	
3		Elimina el fichero y actualiza la tabla
2a2a Cancelar Eliminar Fichero		
Actor		
1		Cancela la petición y muestra nuevamente la interfaz Administrar Fichero
2b Extraer Metadatos		
Actor		Sistema
1	Selecciona la opción que permite extraer los metadatos	
2		Extrae los metadatos del fichero, muestra la interfaz Listar Fichero y actualiza la tabla
Sección 2: Eliminar Ficheros		
Flujo básico Eliminar Ficheros		
Actor		Sistema
1	Selecciona el botón Eliminar todos	
2		Muestra el mensaje de confirmación "¿Seguro que desea eliminar todos los elementos?"
3	Selecciona el botón Eliminar todos	
4		Elimina los ficheros y actualiza la tabla
3a Cancelar Eliminar todos		
Actor		
1		Cancela la petición y muestra nuevamente la interfaz Administrar Fichero

Relaciones	CU incluidos	-																																																										
Prototipo elemental de interfaz gráfica de usuario																																																												
<table border="1"> <thead> <tr> <th>Fichero</th> <th>Tamaño</th> <th>Tipo</th> <th>Fecha</th> <th>Opciones</th> </tr> </thead> <tbody> <tr> <td>10.1.1.34.226.pdf</td> <td>159.04 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>1.pdf</td> <td>85.37 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>236-887-1-PB.pdf</td> <td>244.15 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>251109_09a.pdf</td> <td>201.02 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>32-275-1-PB.pdf</td> <td>280.21 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>5136-8376-1-PB.pdf</td> <td>3.37 MB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>763lecture2.ppt</td> <td>359.5 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>ADA483465.pdf</td> <td>207.99 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>Best Practices for TEI in Libraries.htm</td> <td>386.34 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>Bibliog.rar</td> <td>8.75 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> </tbody> </table>						Fichero	Tamaño	Tipo	Fecha	Opciones	10.1.1.34.226.pdf	159.04 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	1.pdf	85.37 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	236-887-1-PB.pdf	244.15 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	251109_09a.pdf	201.02 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	32-275-1-PB.pdf	280.21 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	5136-8376-1-PB.pdf	3.37 MB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	763lecture2.ppt	359.5 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	ADA483465.pdf	207.99 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	Best Practices for TEI in Libraries.htm	386.34 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	Bibliog.rar	8.75 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>
Fichero	Tamaño	Tipo	Fecha	Opciones																																																								
10.1.1.34.226.pdf	159.04 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
1.pdf	85.37 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
236-887-1-PB.pdf	244.15 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
251109_09a.pdf	201.02 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
32-275-1-PB.pdf	280.21 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
5136-8376-1-PB.pdf	3.37 MB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
763lecture2.ppt	359.5 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
ADA483465.pdf	207.99 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
Best Practices for TEI in Libraries.htm	386.34 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
Bibliog.rar	8.75 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
<p> <input type="button" value="Eliminar todos"/> <input type="button" value="Extraer a todos"/> </p> <p> <input type="text" value="Buscar"/> Mostrar <input type="text" value="10"/> registros </p> <table border="1"> <thead> <tr> <th>Fichero</th> <th>Tamaño</th> <th>Tipo</th> <th>Fecha</th> <th>Opciones</th> </tr> </thead> <tbody> <tr> <td>236-887-1-PB.pdf</td> <td>244 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>32-275-1-PB.pdf</td> <td>280 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>Documento_completo_.pdf</td> <td>171 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> <tr> <td>IPIG-03- Instructivo_Evaluacion_Calidad_Documentacion_Metadatos_Geograficos_V2_0_2011.pdf</td> <td>588 KB</td> <td></td> <td>hace unos segundos</td> <td><input type="checkbox"/> <input checked="" type="checkbox"/></td> </tr> </tbody> </table>						Fichero	Tamaño	Tipo	Fecha	Opciones	236-887-1-PB.pdf	244 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	32-275-1-PB.pdf	280 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	Documento_completo_.pdf	171 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>	IPIG-03- Instructivo_Evaluacion_Calidad_Documentacion_Metadatos_Geograficos_V2_0_2011.pdf	588 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																														
Fichero	Tamaño	Tipo	Fecha	Opciones																																																								
236-887-1-PB.pdf	244 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
32-275-1-PB.pdf	280 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
Documento_completo_.pdf	171 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								
IPIG-03- Instructivo_Evaluacion_Calidad_Documentacion_Metadatos_Geograficos_V2_0_2011.pdf	588 KB		hace unos segundos	<input type="checkbox"/> <input checked="" type="checkbox"/>																																																								

Anexo 2.4: Extraer Metadato

Objetivo	Extraer los metadatos de los ficheros seleccionados
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario desea extraer metadatos y termina cuando se han extraído los metadatos a todos los ficheros
Complejidad	Alta
Prioridad	Media
Precondiciones	El usuario está autenticado, existen ficheros en la base de datos
Postcondiciones	Se extrajeron los metadatos a todos los ficheros satisfactoriamente
Flujo de eventos	
Flujo básico Extraer Metadatos	
Actor	Sistema
1	Selecciona el botón Extraer a todos
2	Extrae los metadatos de todos los ficheros, muestra la interfaz Listar Fichero y actualiza la tabla

Relaciones	CU incluidos	-
Prototipo elemental de interfaz gráfica de usuario		

Anexo 2.5: Gestionar Ítem

Objetivo	Gestionar los Ítems en DEPXOS	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario desea listar, editar o eliminar los ítems que se encuentran en DEPXOS	
Complejidad	Media	
Prioridad	Media	
Precondiciones	El usuario ha sido autenticado y ha extraído los metadatos como mínimo a un fichero	
Postcondiciones	Se lista, edita o elimina los ítems	
Flujo de eventos		
Flujo básico Gestionar ítem		
Actor	Sistema	
Indica que desea listar, editar o eliminar mínimo un ítem	a) Si el usuario decide listar ítem Ver Sección 1: Listar Ítem b) Si el usuario decide eliminar ítem Ver Sección 2: Eliminar Ítem	
Sección 1: Listar Ítem		
Flujo básico Listar Ítem		
Actor	Sistema	
1	Selecciona la opción Administrar Ítem	
2		Muestra una interfaz con una tabla con todos los

		<p>ítems existentes en el servidor pertenecientes a ese usuario, de los ítems se muestran los datos: Ítem (nombre), Tamaño, Tipo, Fecha, Evaluación y Opciones. Las opciones que se brindan son:</p> <ul style="list-style-type: none"> • editar cada ítem (ver el flujo alternativo 2a editar ítem) • eliminar ítem (ver el flujo alternativo 2b eliminar ítem) • depositar en REPXOS cada ítem (ver el flujo alternativo 2c depositar en REPXOS)
Flujos alternos		
2a Editar Ítem		
Actor		Sistema
1	Selecciona la opción que permite editar el ítem	
2		Muestra la interfaz Editar Ítem que permite editar los metadatos del ítem
3	Selecciona el botón Aceptar	
4		Valida los nuevos metadatos del ítem y muestra un mensaje de información "El ítem ha sido editado"
2a3a Cancelar Editar Ítem		
Actor		
1		Cancela la petición y muestra nuevamente la interfaz Administrar Ítem
2a3b Añadir nuevo metadato		
Actor		
1		Añade un nuevo metadatos con los campos necesarios
2a4a Valores vacíos		
Actor		Sistema

1		Muestra el mensaje de información "El envío contenía valores inválidos"
2b Eliminar Ítem		
Actor		Sistema
1	Selecciona la opción que permite eliminar el ítem	
2		Muestra un mensaje de confirmación "¿Seguro que desea eliminar el elemento?"
3	Selecciona el botón Eliminar	
4		Elimina el ítem y actualiza la tabla
2b3a Cancelar Eliminar Ítem		
Actor		
1		Cancela la petición y muestra nuevamente la interfaz Administrar Ítem
2c Depositar en REPXOS		
Actor		Sistema
1	Selecciona la opción que permite depositar en REPXOS	
2		Muestra una interfaz para seleccionar la colección donde se depositarán los ítems. Muestra el mensaje de información "Seleccione la colección a la cual desea hacer el envío"
3	Selecciona el botón Depositar	
4		Deposita el ítem en REPXOS y muestra la interfaz Administrar Ítem
2c3a Cancelar Depositar en REPXOS		
Actor		
1		Cancela la petición y muestra nuevamente la interfaz Administrar Ítem
Sección 2: Eliminar Ítem		
Flujo básico Eliminar Ítem		

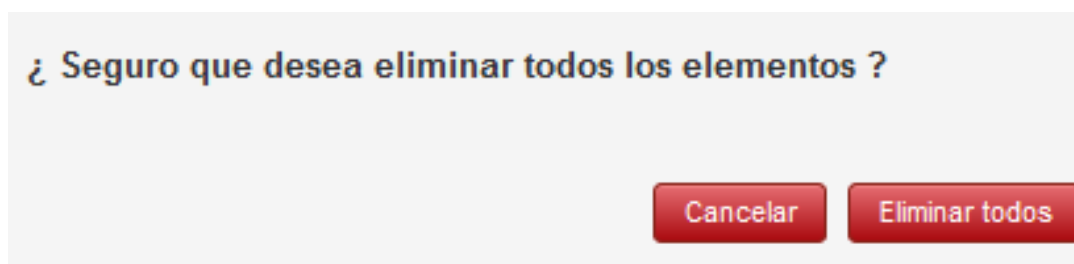
Actor		Sistema
1	Selecciona el botón Eliminar todos	
2		Muestra el mensaje de confirmación "¿Seguro que desea eliminar todos los elementos?"
3	Selecciona el botón Eliminar	
4		Elimina todos los ítems y actualiza la tabla

3b Cancelar Eliminar Ítem

Actor		Sistema
1		Cancela la petición y muestra nuevamente la interfaz Administrar Ítem

Relaciones CU incluidos -

Prototipo elemental de interfaz gráfica de usuario

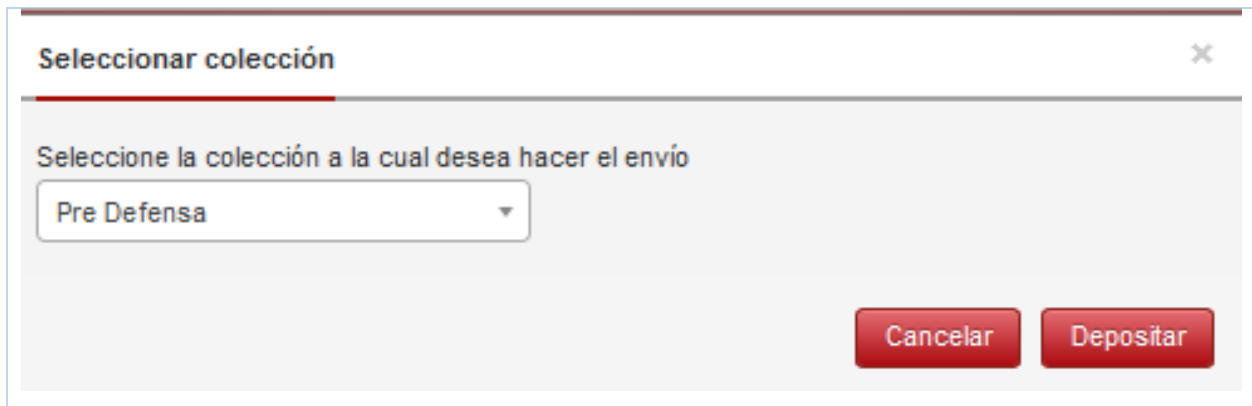


Ítem	Tamaño	Tipo	Fecha	Evaluación	Opciones
validar metadatos (by ME).doc	19 KB		hace 2 minutos	☆☆☆	
uciencia-2012-t46-p1035-ponencia-1462.pdf	335.13 KB		hace 2 minutos	★☆☆	
tesisEnGrails1.pdf	2.3 MB		hace 2 minutos	★☆☆	
tei-print.css	2.79 KB		hace 2 minutos	☆☆☆	
tei.css	16.47 KB		hace 2 minutos	☆☆☆	
TD_Deulert.pdf	2.02 MB		hace 2 minutos	★☆☆	
SWORD Portuges.pdf	1.79 MB		hace 2 minutos	☆☆☆	
sword.pdf	267.67 KB		hace 2 minutos	★☆☆	
sflgo.png	2.06 KB		hace 2 minutos	☆☆☆	
Seleccion deposito (ME).doc	2.58 KB		hace 2 minutos	☆☆☆	

Anexo 2.6: Depositar Ítem en REPXOS

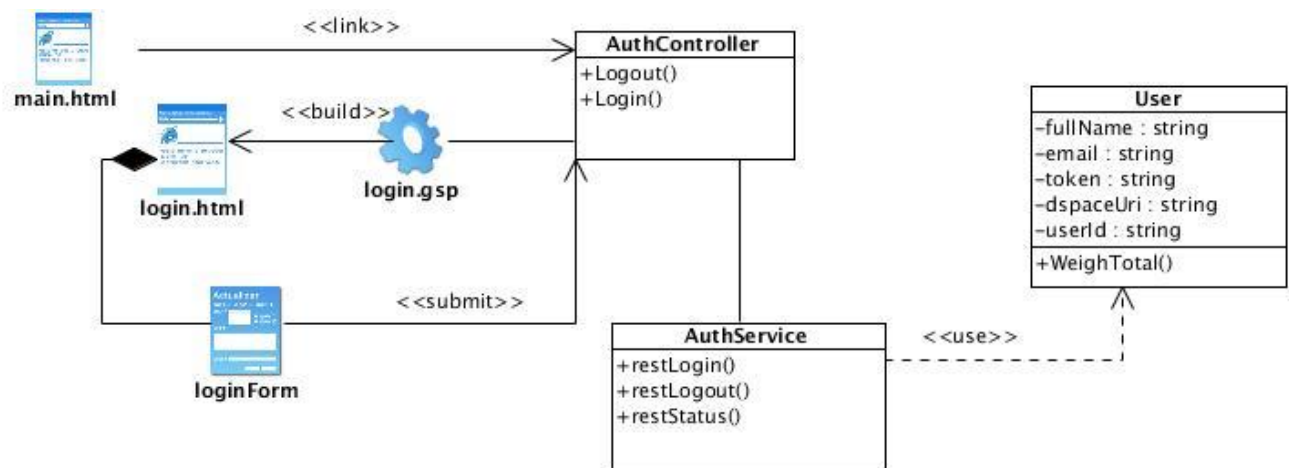
Objetivo	Depositar los ítems en REPXOS
Actores	Usuario

Resumen	El caso de uso inicia cuando el usuario selecciona la opción Depositar todos y termina una vez estén depositados en REPXOS	
Complejidad	Media	
Prioridad	Media	
Precondiciones	Existen ítems en la base de datos	
Postcondiciones	Se depositaron satisfactoriamente los ítems seleccionados a REPXOS	
Flujo de eventos		
Flujo básico Depositar Ítems a REPXOS		
	Actor	Sistema
1	Selecciona la opción Administrar Ítems	
2		Muestra una interfaz para seleccionar la colección donde se depositarán los ítems. Muestra el mensaje de información "Seleccione la colección a la cual desea hacer el envío"
3	Selecciona la colección donde se depositarán los ítems y presiona el botón Depositar todos	
4		Deposita todos los ítems en REPXOS y muestra la interfaz Administrar Ítem
Flujos alternos		
3a Cancelar Depositar en REPXOS		
	Actor	Sistema
1		Cancela la petición y muestra nuevamente la interfaz Administrar Ítem
Relaciones	CU incluidos	-
Prototipo elemental de interfaz gráfica de usuario		

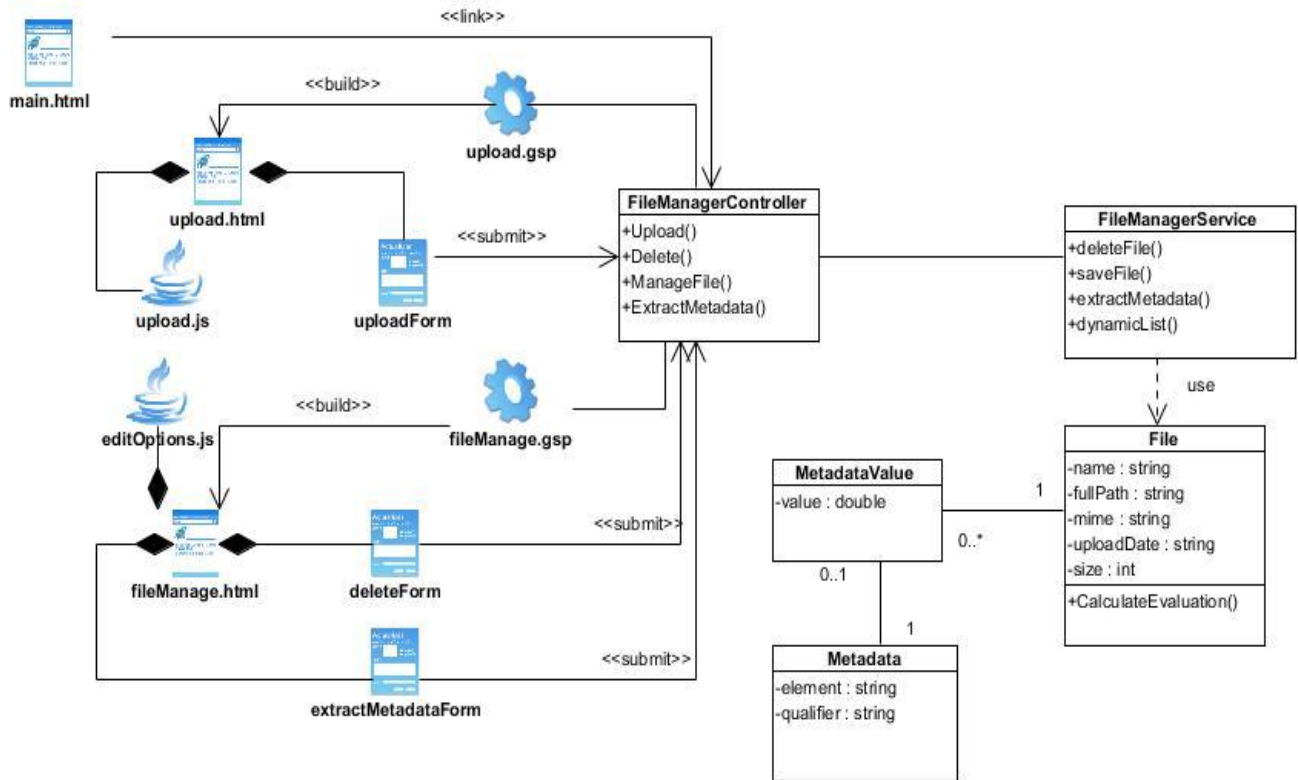


Anexo 3 Diagramas de clases del diseño utilizando estereotipos web

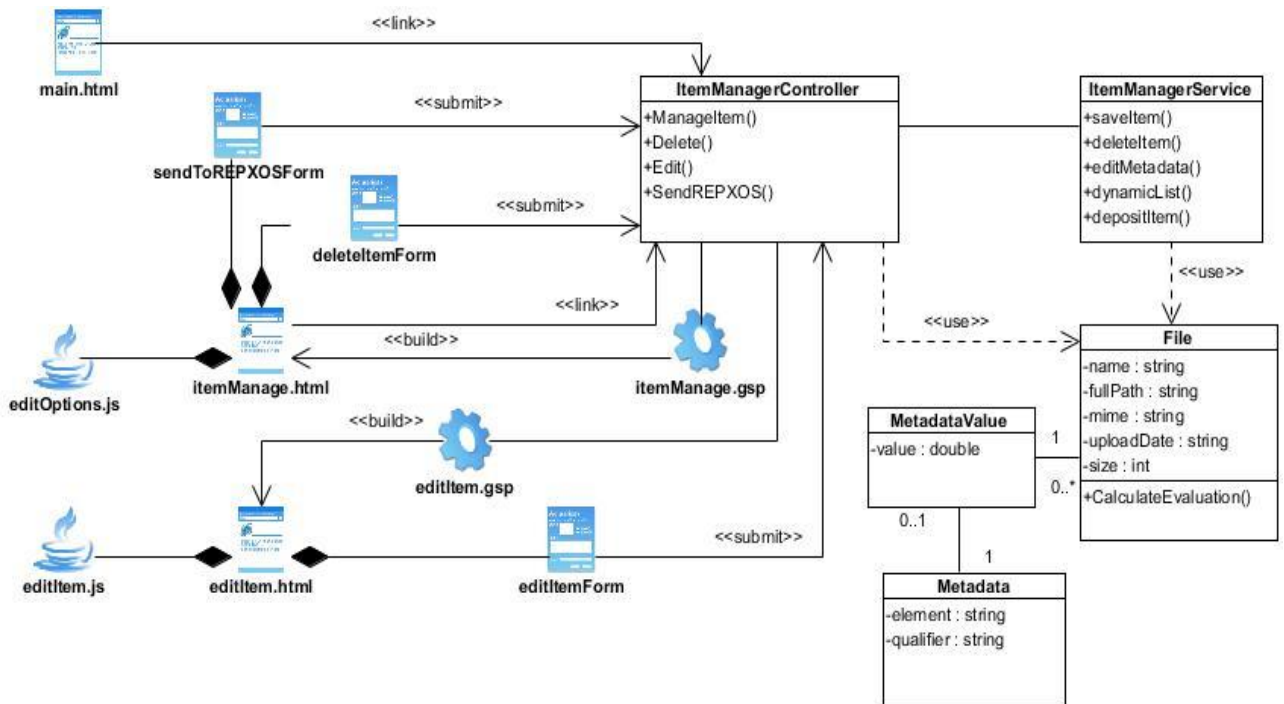
Anexo 3.1: Autenticar Usuario



Anexo 3.2: Subir y Gestionar Fichero

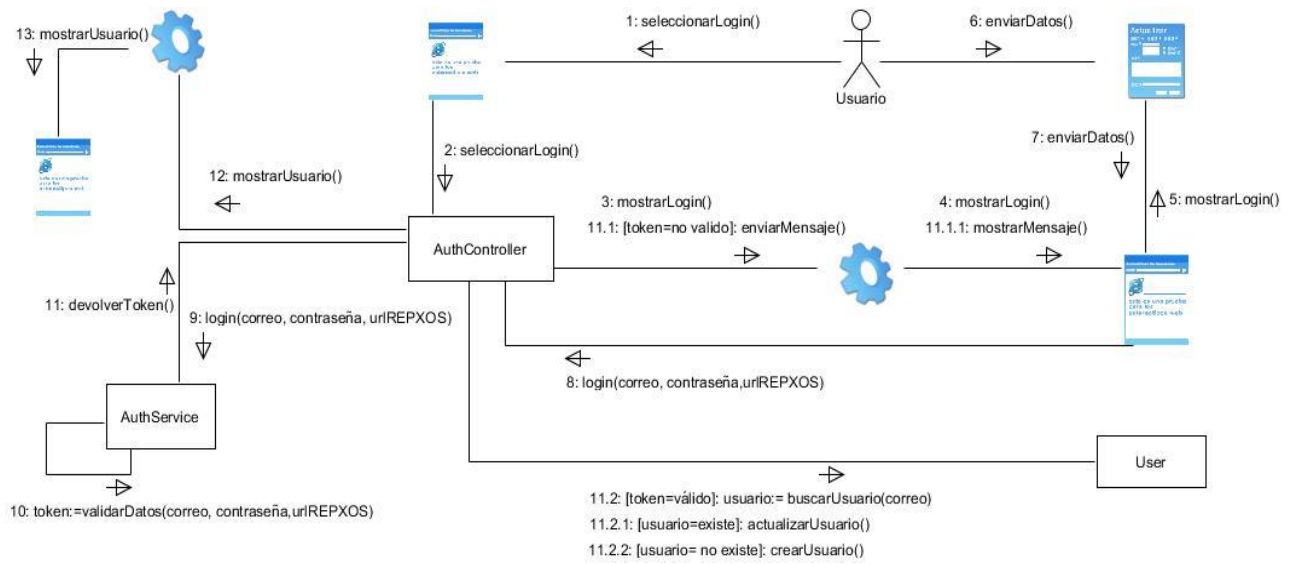


Anexo 3.3: Gestionar Ítem y Depositar Ítem en REPXOS

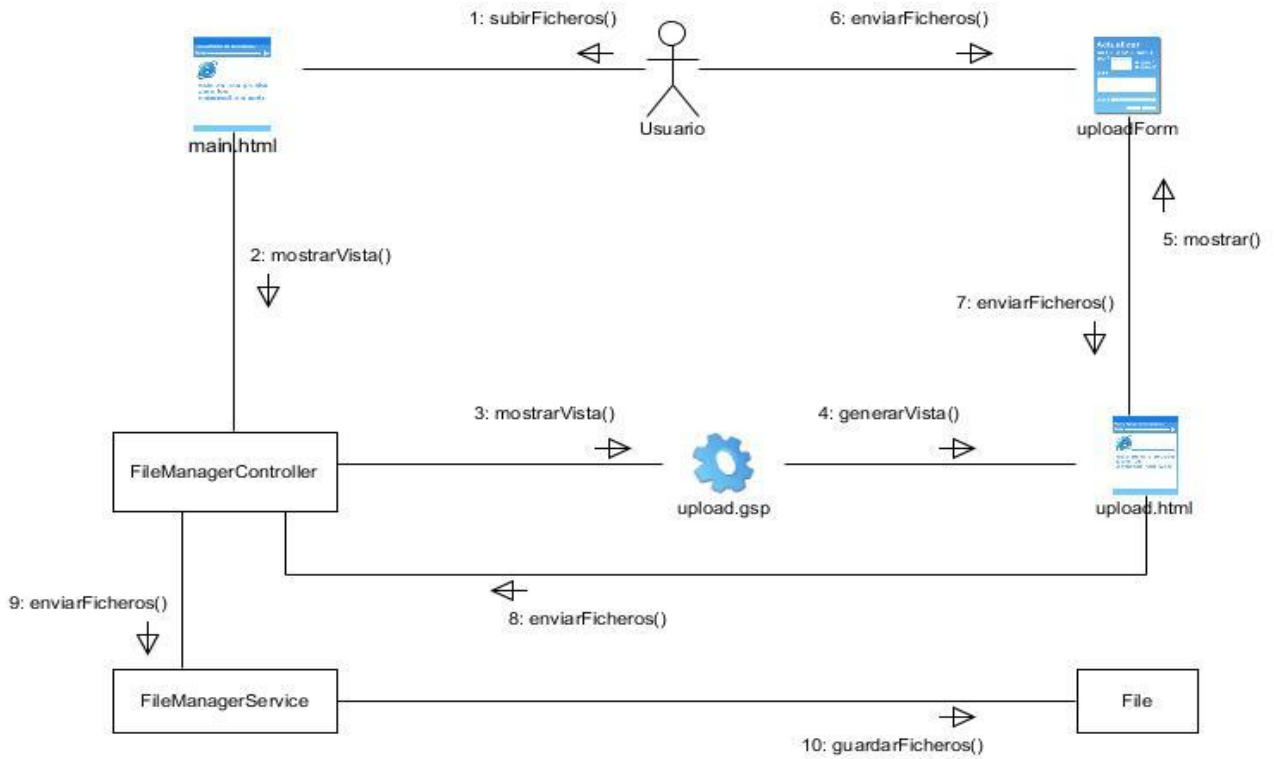


Anexo 4: Diagramas de Colaboración

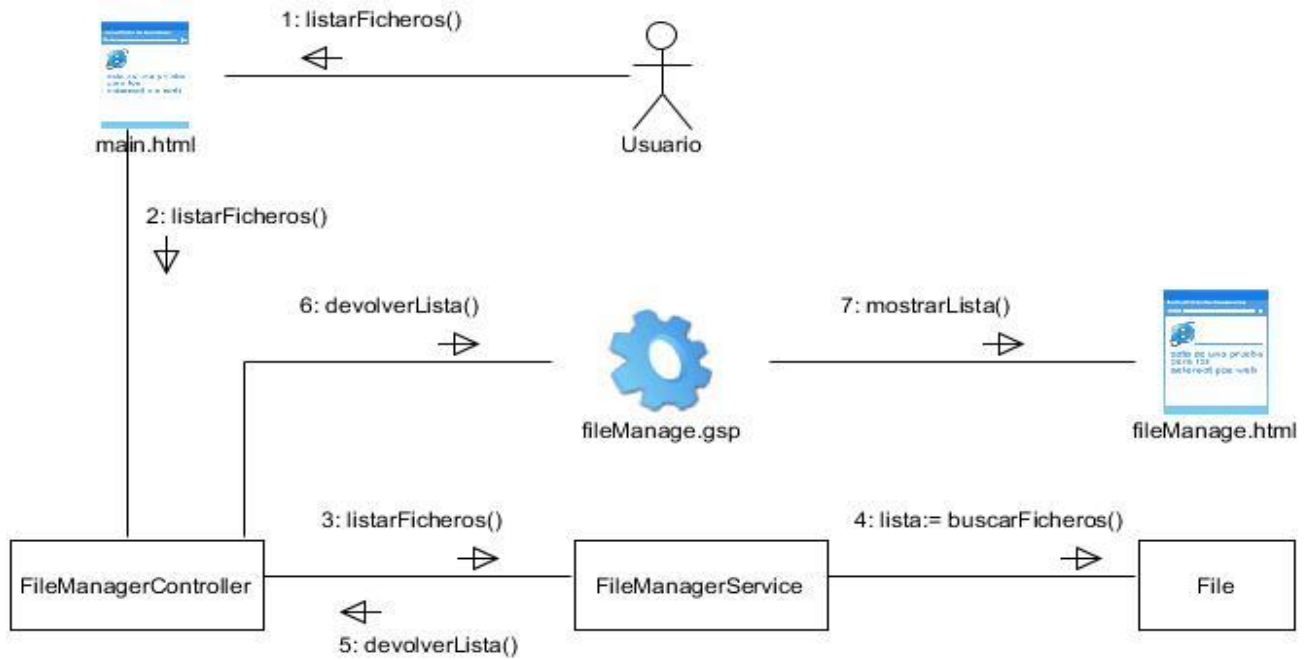
Anexo 4.1: Autenticar Usuario



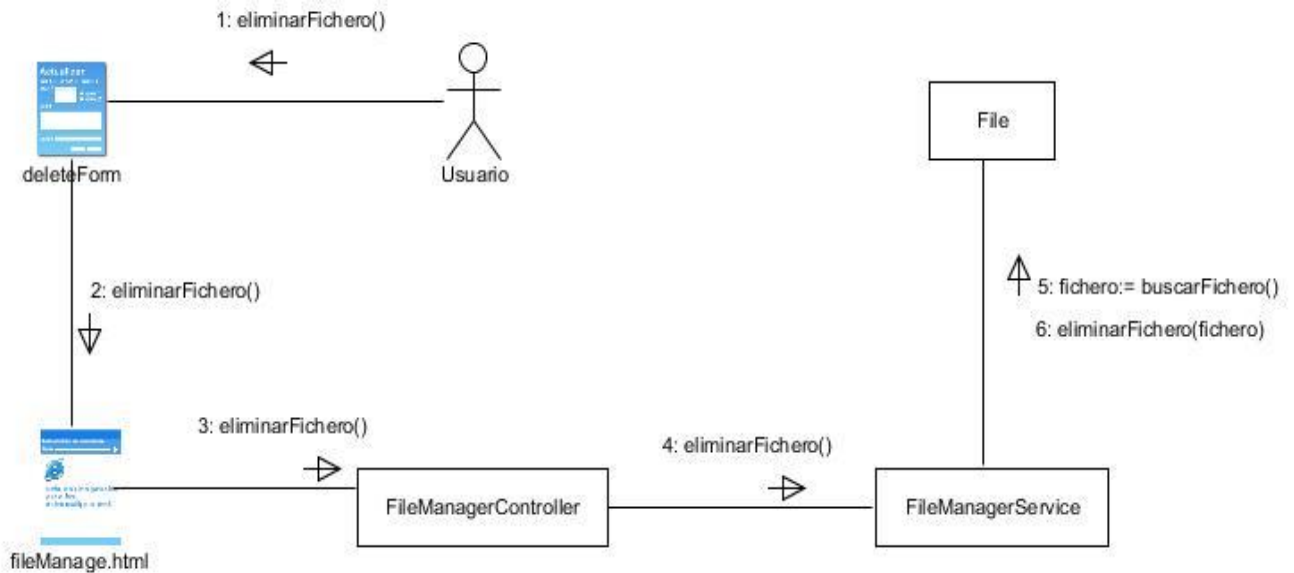
Anexo 4.2: Subir Fichero



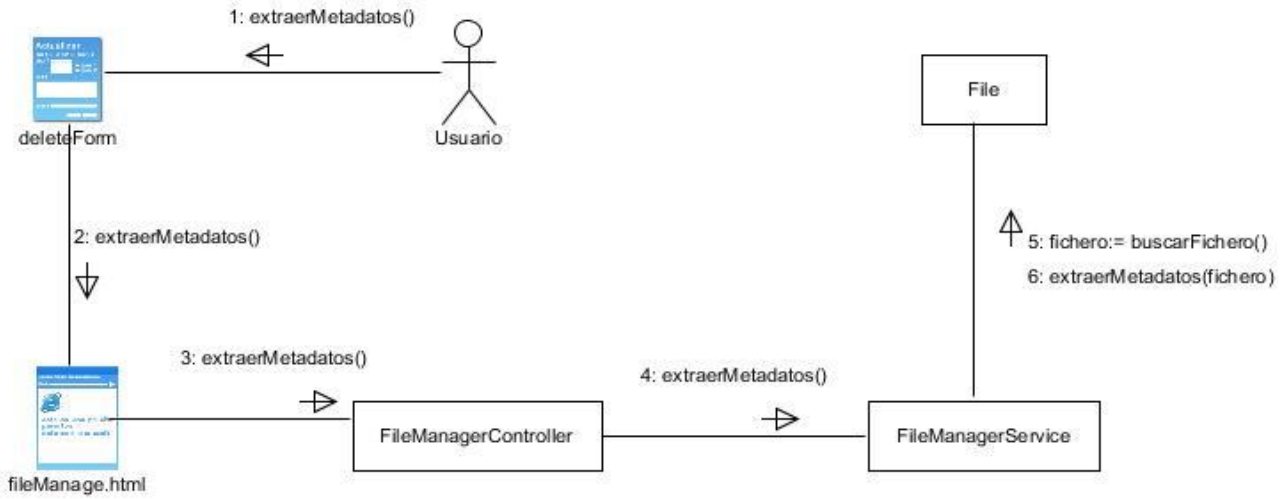
Anexo 4.3: Listar Fichero



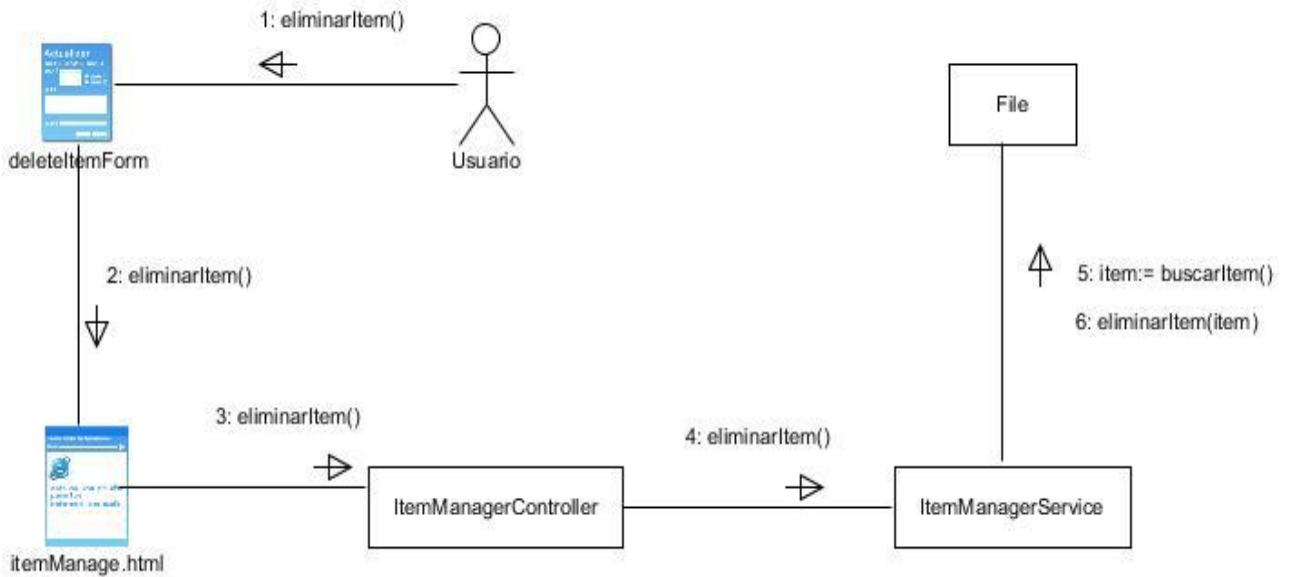
Anexo 4.4: Eliminar Fichero



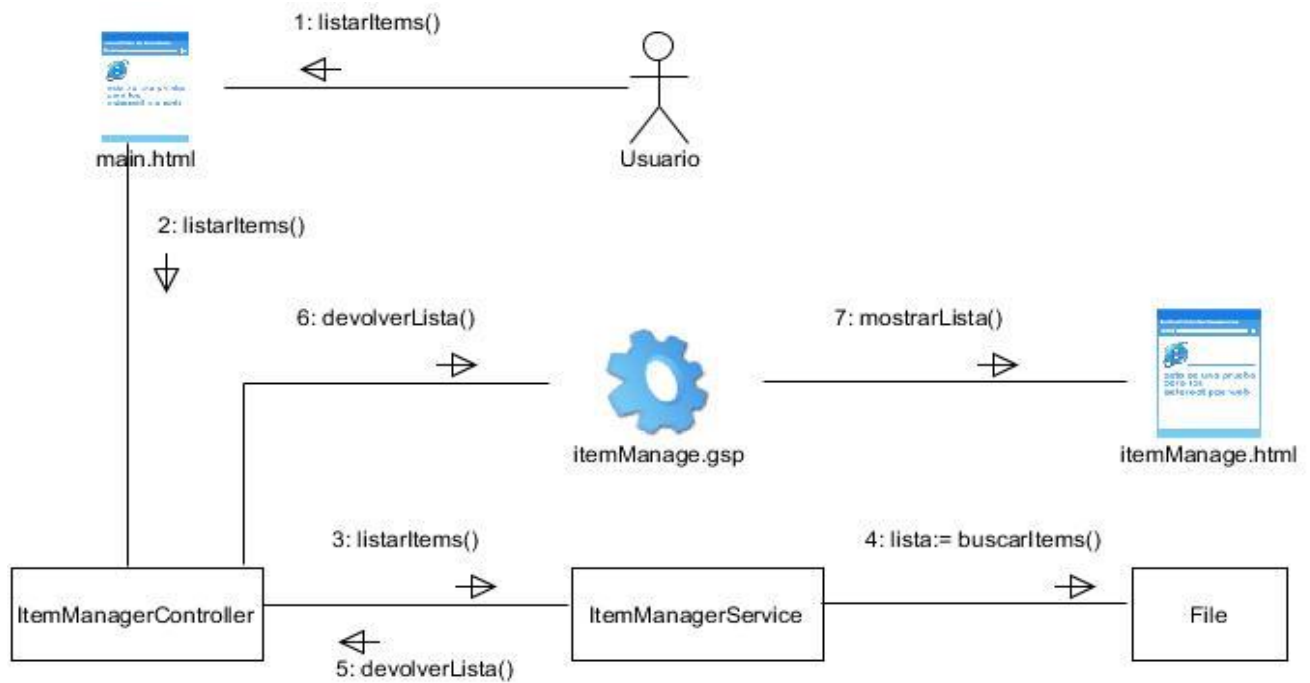
Anexo 4.5: Extraer Metadatos



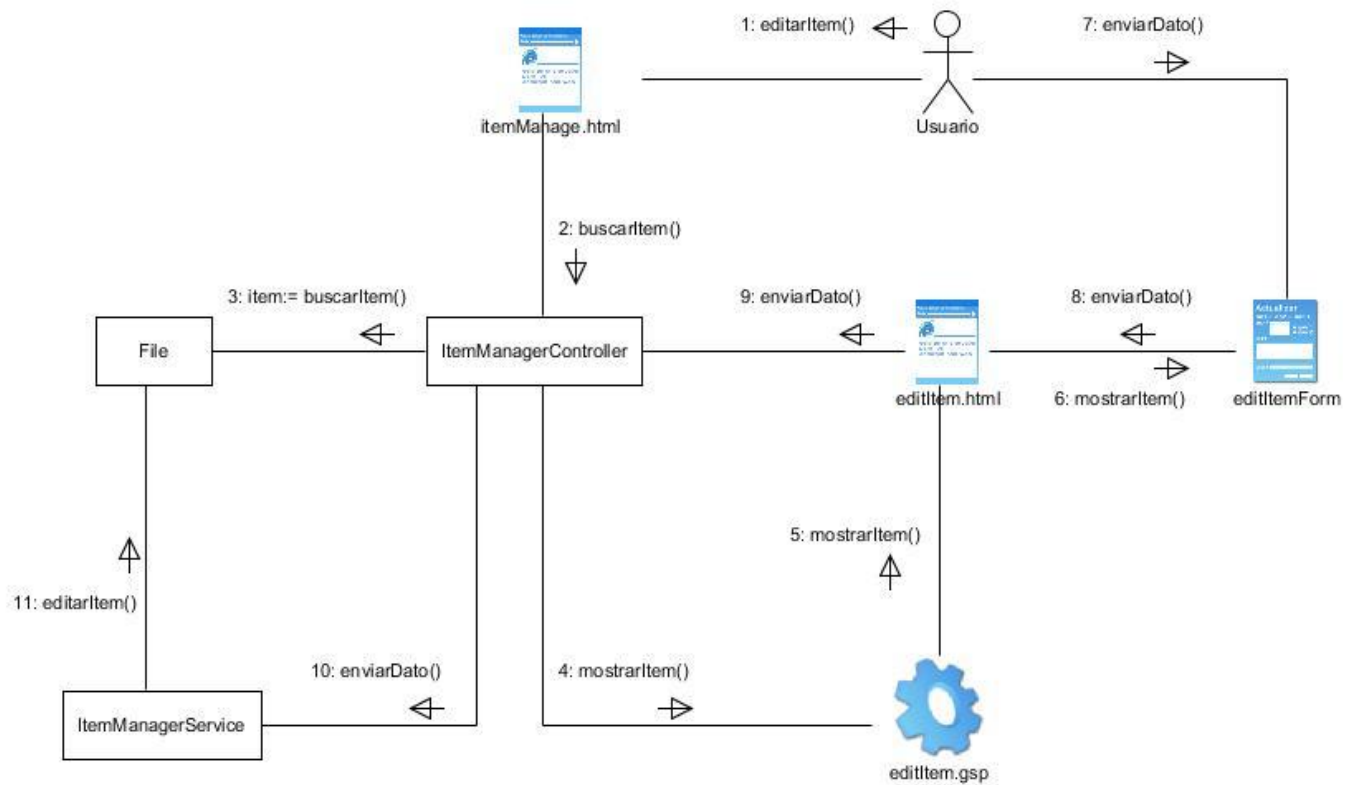
Anexo 4.6: Eliminar Ítem



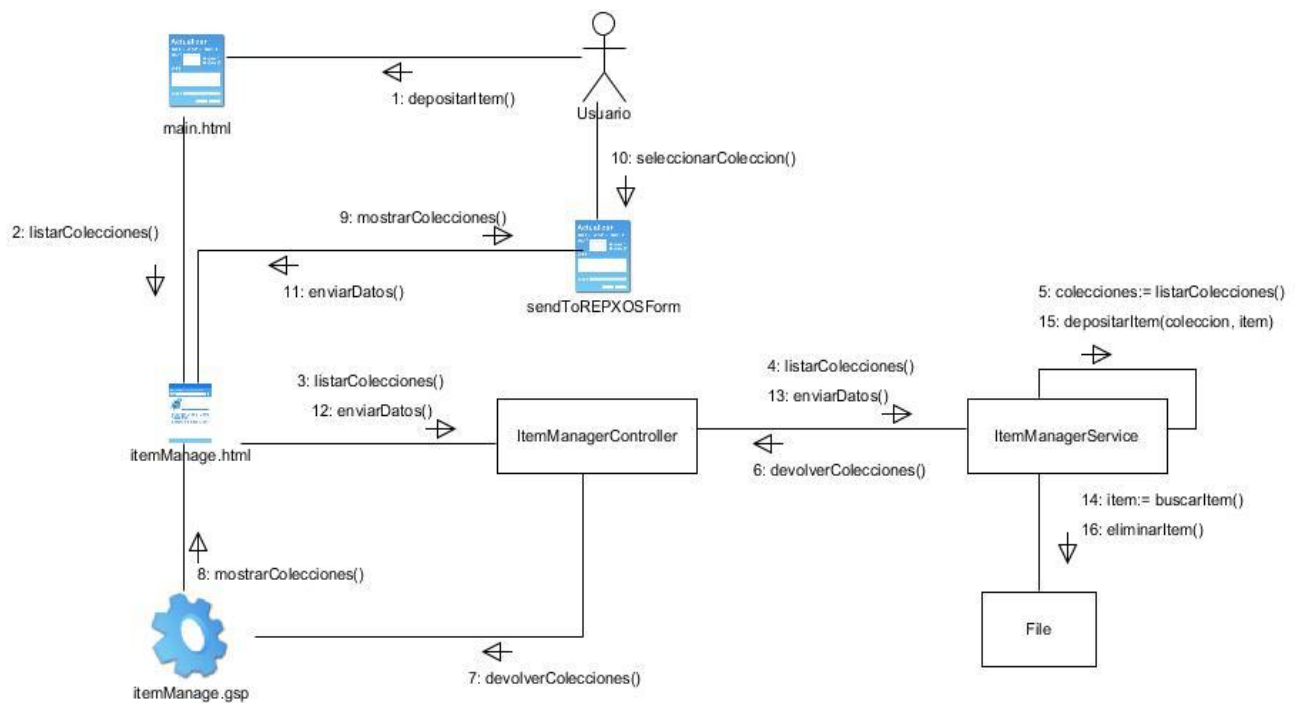
Anexo 4.7: Listar Ítem



Anexo 4.8: Editar Ítem



Anexo 4.9: Depositar Ítem en REPXOS



Anexo 5 Casos de Prueba

Anexo 5.1: Gestionar Fichero

SC1 Listar Fichero

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Listar Fichero	Permite al usuario listar los ficheros subidos por él	Brinda la posibilidad de listar todos los ficheros correspondientes al usuario en la tabla Listado de ficheros	1- Seleccionar del menú superior la opción "Administrar Ficheros"

SC2 Eliminar Fichero

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Eliminar Fichero	Permite al usuario eliminar los ficheros subidos por él	Brinda la posibilidad de eliminar uno a uno los ficheros correspondientes al usuario. Muestra el mensaje: "¿Seguro que desea eliminar el elemento"	1- Seleccionar del menú superior la opción "Administrar Ficheros" 2- Seleccionar en la columna Opciones la primera opción correspondiente al fichero

SC3 Eliminar todos los Ficheros

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Eliminar todos los Ficheros de una vez	Permite al usuario eliminar todos los ficheros de una vez subidos por él	Brinda la posibilidad de eliminar de una vez todos los ficheros correspondientes al usuario. Muestra el mensaje: "¿Seguro que desea eliminar todos los elementos"	1- Seleccionar del menú superior la opción "Administrar Ficheros" 2- Seleccionar en el menú superior de la tabla Listado de ficheros el botón "Eliminar todos"

Anexo 5.2: Autenticar Usuario

SC1 Autenticar Usuario

El caso de prueba presenta como variables el campo URL, Correo y Contraseña, además se pueden encontrar los escenarios de datos vacíos y valores inválidos. Se brindan los ejemplos de datos válidos e inválidos y lo que devuelve la aplicación en caso de cada ejemplo.

Escenario	Descripción	URL	Correo	Contraseña	Respuesta del sistema	Flujo central
EC 1.1 Autenticar Usuario	Permite al usuario autenticarse exitosamente en el sistema	V	V	V	Brinda la posibilidad de autenticar un usuario, para ello debe introducir: • correo electrónico y contraseña de usuario además de la URL de REPXOS	1- Introducir los datos correctamente 2- Seleccionar la opción "Entrar"
		http://192.168.1.2:8080/rest	avmachado@estudiantes.uci.cu	aray12		
EC 1.2 Datos vacíos	Existen campos vacíos y no se puede registrar correctamente la sanción	I	V	V	Muestra el mensaje de error "El campo x no puede ser nulo"	1- Introducir los datos y dejar algún campo vacío 2- Seleccionar la opción "Entrar"
		vacío	avmachado@estudiantes.uci.cu	aray12		
		V	V	V		
EC 1.3 Valores inválidos	Valida que no existan campos incorrectos	http://192.168.1.2:8080/rest	avmachado@estudiantes.uci.cu	vacío	Muestra el mensaje de error "Usted no tiene acceso al recurso solicitado"	1- Introducir datos erróneos 2- Seleccionar la opción "Entrar"
		V	V	V		
		http://192.168.1.2:8080	avmachado@estudiantes.uci.cu	aray12		
		http://192.168.1.2:8080/rest	avmachado	aray12		
		http://192.168.1.2:8080/rest	avmachado@estudiantes.uci.cu	1234567890		

Anexo 5.3: Subir Fichero

SC1 Subir Fichero

Descripción	Respuesta del sistema	Flujo central
Permite al usuario subir los ficheros al sistema	Brinda la posibilidad al usuario de seleccionar los ficheros que desea subir a REPXOS Muestra el mensaje: "Ficheros enviados correctamente"	1- Seleccionar la opción "Clic para buscar" 2- Seleccionar los archivos 3- Seleccionar la opción "Subir"

Anexo 5.4: Extraer Metadatos

SC1 Extraer Metadatos

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Extraer metadatos	Permite al usuario extraer los metadatos de cada uno de los ficheros subidos por él	Brinda la posibilidad de extraer los metadatos de cada uno de los ficheros subidos por él	1- Seleccionar del menú superior la opción "Administrar Ficheros" 2- Seleccionar en la columna Opciones la segunda opción correspondiente al fichero

SC2 Extraer todos los Metadatos

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Extraer los metadatos de los Ficheros de una vez	Permite al usuario extraer de una vez los metadatos de todos los ficheros subidos por él	Brinda la posibilidad de extraer de una vez los metadatos de todos los ficheros subidos por él. Dirige al usuario a la interfaz Administrar Ítem	1- Seleccionar del menú superior la opción "Administrar Ficheros" 2- Seleccionar en el menú superior de la tabla Listado de ficheros el botón "Extraer a todos"

Anexo 5.5: Gestionar Ítem

SC1 Listar Ítem

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Listar Ítem	Permite al usuario listar los ítems subidos por él	Brinda la posibilidad de listar todos los ítems correspondientes al usuario en la tabla Listado de ítems	1- Seleccionar del menú superior la opción "Administrar ítems"

SC2 Eliminar Ítem

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Eliminar Ítem	Permite al usuario eliminar los ítems subidos por él	Brinda la posibilidad de eliminar uno a uno los ítems correspondientes al usuario. Muestra el mensaje: "¿Seguro que desea eliminar el elemento"	1- Seleccionar del menú superior la opción "Administrar ítems" 2- Seleccionar en la columna Opciones la primera opción correspondiente al ítem en la tabla Listado de ítems

SC3 Eliminar todos los Ítems

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 3.1 Eliminar todos los ítems de una vez	Permite al usuario eliminar todos los ítems de una vez subidos por él	Brinda la posibilidad de eliminar de una vez todos los ítems correspondientes al usuario. Muestra el mensaje: "¿Seguro que	1- Seleccionar del menú superior la opción "Administrar ítems"

		desea eliminar todos los elementos"	2- Seleccionar en el menú superior de la tabla Listado de ítems el botón "Eliminar todos"
--	--	-------------------------------------	---

SC4 Editar Ítem

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 4.1 Editar Ítem	Permite al usuario editar todos los metadatos de cada ítem subido por él	Brinda la posibilidad de editar los metadatos de cada ítem correspondiente al usuario. Muestra el mensaje: "El ítem ha sido editado"	1- Seleccionar del menú superior la opción "Administrar ítems" 2- Seleccionar en la columna Opciones la segunda opción correspondiente al ítem en la tabla Listado de ítems

GLOSARIO DE TÉRMINOS

Bytecode: Es un conjunto de instrucciones altamente optimizadas que son interpretados por el intérprete de Java (Libro: Fundamentos de programación en Java 2 por Mc Herbert Shildt).

CamelCase: Tipo de escritura que se caracteriza porque las palabras van unidas entre sí sin espacio, con la peculiaridad de que la primera letra de cada término se encuentra en mayúscula para hacer más legible el conjunto.

Closures: Un closure en groovy es un bloque de código que puede tomar argumentos, devolver valores y ser asignado a una variable.

GPS: Es un sistema de radionavegación, basado en el espacio, que proporciona servicios fiables de posicionamiento, navegación, y cronometría a usuarios civiles en todo el mundo. A todo el que cuente con un receptor del GPS, el sistema le proporcionará su localización y la hora exacta en cualesquiera condiciones atmosféricas, de día o de noche, en cualquier lugar del mundo y sin límite al número de usuarios simultáneos.

GROBID: Generación de Datos Bibliográficos.

Disecionar: Examinar o analizar de forma minuciosa y detallada.

Interface: Es un conjunto de declaraciones de funciones.

PDF (Portable Document Format): Es un formato de almacenamiento de documentos digitales.

Procedimientos almacenados: En PostgreSQL se puede definir como un programa, procedimiento o función, el cual está almacenado en la base de datos y listo para ser usado.

REST: Transferencia de Estado Representacional.

TEI: Es un estándar para la representación de textos en formato digital mediante la utilización de un marcado semántico.

Triggers; Son objetos que se asocian con tablas y se almacenan en la base de datos. Su nombre se deriva por el comportamiento que presentan en su funcionamiento, ya que se ejecutan cuando sucede algún evento sobre las tablas a las que se encuentra asociado. Los eventos que hacen que se ejecute un trigger son las operaciones de inserción (INSERT), borrado (DELETE) o actualización (UPDATE), ya que modifican los datos de una tabla.