Universidad de las Ciencias Informáticas
Facultad 1

“Gestión de metadatos asociados a las tipologías documentales definidas en el Gestor de Documentos Administrativos eXcriba”

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

Autora: Daliana Noa Heredia

Tutores: Ing. Dayelis Blanco Hernández
Ing. Reinier Elejalde Chacon

Ciudad de la Habana, Junio 2012

Declaración de autoría

DECLARO: Que soy el único autor de este trabajo y autorizo al Centro de Informatización Universitaria de la Universidad de las Ciencias Informáticas, para que hagan el uso que estimen pertinente con este trabajo.

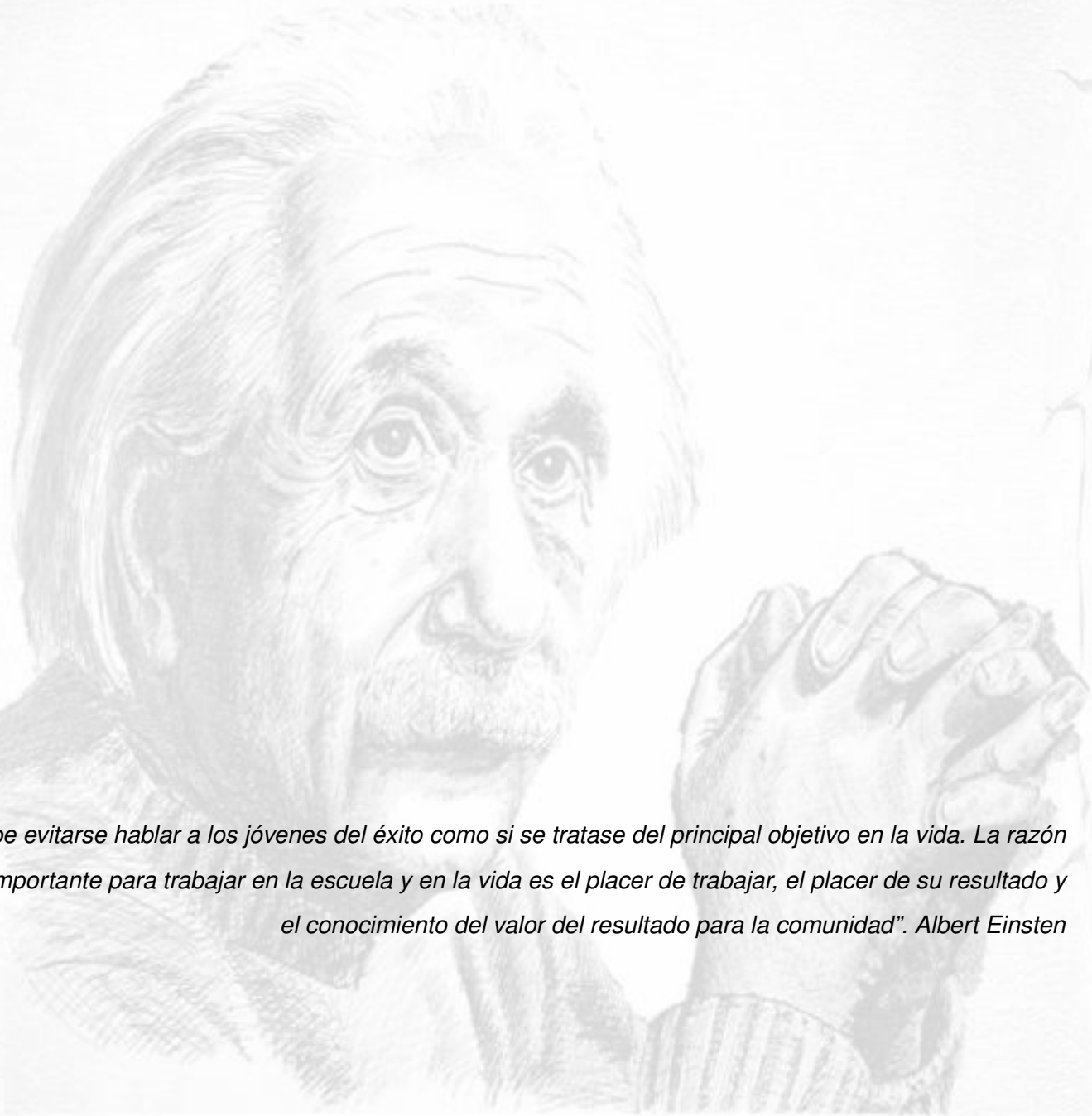
Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Daliana Noa Heredia

_____ _____

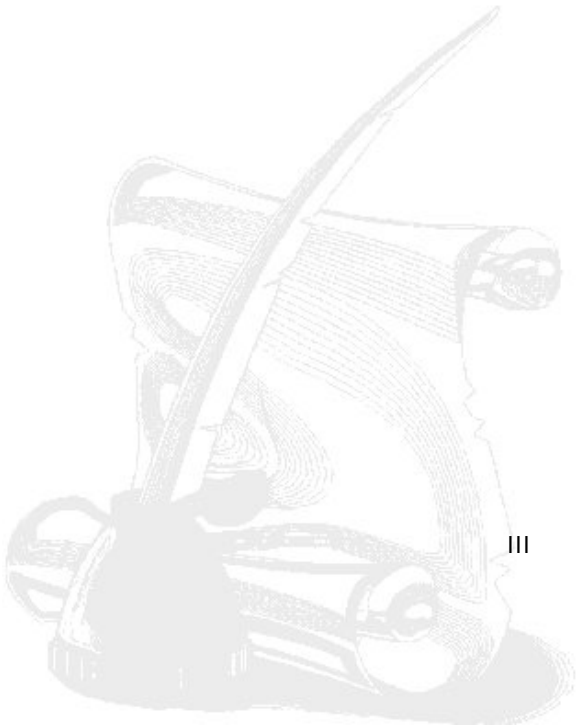
Ing. Dayelis Blanco Hernández Ing. Reinier Elejalde Chacon



“Debe evitarse hablar a los jóvenes del éxito como si se tratase del principal objetivo en la vida. La razón más importante para trabajar en la escuela y en la vida es el placer de trabajar, el placer de su resultado y el conocimiento del valor del resultado para la comunidad”. Albert Einsten

Agradecimientos

En primer lugar quiero agradecer a mis padres, a mi hermana y a mis abuelos por su confianza y apoyo.
A mis tutores Reinier y Dayelis, ya que gracias a su ayuda hoy logro uno de mis mayores sueños.
A mi novio Amaury Viera por darme todo su cariño, amor y comprensión a lo largo de mi carrera.
A mis tías Carmen y Julia, que son dos madres más para mí.
Agradecer además a varias personas, con las cuales comparto diariamente en el proyecto y han aportado su granito de arena a esta tesis (Pedro, Michel, Misael y Lisandra).
A mi amiga Dailys y a mi antiguo grupo 10108, especialmente a Rubén Reinaldo que nunca olvidaré lo que hizo por mí.
También quiero agradecer a Rodain Terrer por ayudarme cuando lo necesité.
A mis compañeras de apartamento, que siempre voy a recordar sus locuras (Lázara, Diana, Analaicy, Mairim, Yanet, Lianet y Ariannis).
A todos mi amigos del grupo 10205 y 1507, especialmente a Lissete, Aliuska, Mirley, Alberto, Liset y David.
Por último y no menos importante a Yoel Rivera Suárez por la ayuda prestada en esta tesis.
A todos muchísimas gracias.....



Dedicatoria

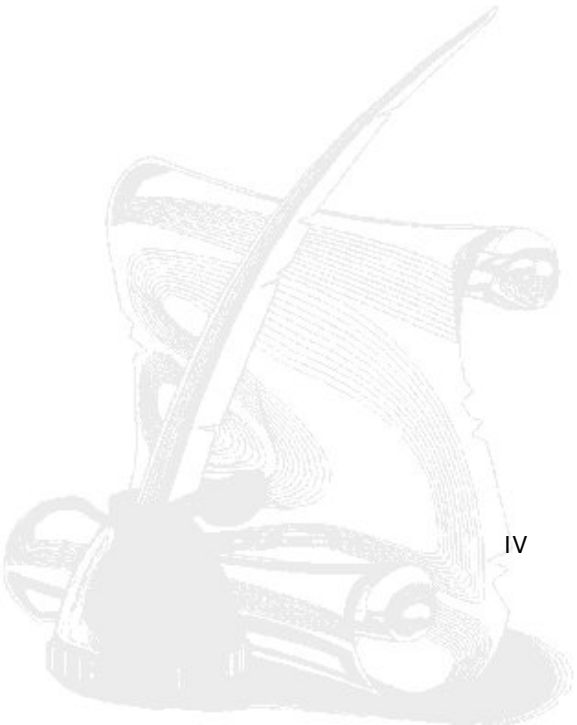
Esta tesis va dedicada especialmente a mi gran amigo Reinier Elejalde Chacon, por siempre estar presente en mi vida, por los momentos compartidos y por su apoyo incodicional.

A mi novio Amaury Viera Hernández, el cual ha sido mi guía a lo largo de mi carrera, por su amor y comprensión, por aguantarme mis malcriadeces y enseñarme que para lograr cosas buenas en la vida hay que perseverar y sacrificarse.

A mi madre y mi abuela, que son mis más grandes tesoros y a ellas debo lo que soy ahora.

A mi padre por su cariño y amor.

A mi toda mi familia, especialmente a mi hermana, a Leunam, a mi tía Carmen y Julia.



Resumen

Cada día se produce más información, y cada día es más importante identificarla y describirla, la mayoría de las organizaciones que poseen un Sistema de Gestión Documental necesitan acceder y consultar de forma frecuente la información archivada, es aquí donde entran a desempeñar un papel fundamental los metadatos asegurando la existencia y perdurabilidad a través de los años de la información.

Debido a la importancia que tienen los metadatos en los procesos documentales y a la necesidad de extender las funcionalidades del sistema eXcriba se desarrollará la presente investigación en función de que se pueda garantizar la correcta gestión de los metadatos asociados a las diferentes tipologías documentales existentes, además de facilitar la incorporación de metadatos a los documentos, independientemente de la norma que se use para describirlos. Contribuyendo a que se mejoren los procesos en los que intervienen los documentos que el eXcriba maneja.

Palabras clave: información, documentos, metadatos.

Índice general

Índice de figuras

Índice de tablas

Introducción

El mundo empresarial se ha caracterizado por la necesidad de tomar control total sobre los procesos que se llevan a cabo dentro de una institución, lo cual deriva en la gestión eficaz de la información que da vida a dichos procesos. Hasta hace unas décadas, la documentación en formato duro producida en cada organización como resultado de transacciones, procesos estratégicos, administrativos y los propios del negocio se almacenaba en gavetas, estantes y archivos. Así, el constante crecimiento del volumen de documentos afectaba directamente la disponibilidad y recuperación de la información, incluyendo los gastos por concepto de remodelación o adquisición de nuevos locales para el almacenamiento.

La alineación de las administraciones con las normativas y procedimientos propuestos en ese entonces por la gestión documental constituyó el paso fundamental en la consolidación de la continuidad del negocio en dichas instituciones. El cumplimiento de las pautas propuestas por esta área les proveía a sus ejecutores una forma más eficiente de aprovechar el verdadero valor de la información impresa en los documentos, agilizando los procesos y de cierta forma elevando el nivel de satisfacción de clientes y trabajadores.

Lamentablemente, en cierta forma para estas instituciones, la estabilidad alcanzada con la adopción de las normativas y procedimientos propuestos por la gestión documental en su accionar no duraría para siempre; en paralelo evolucionaban vertiginosamente las Tecnologías de la Información y las Comunicaciones (TIC). Poco a poco, los documentos se producían con más frecuencia en formato electrónico, sin contar los grandes logros alcanzados en el campo de la digitalización, proceso capaz de convertir la información soportada en papel a formato electrónico. Sin dudas, el carácter revolucionario y transformador de las TIC se introdujo en las administraciones como una necesidad ante el cambio de la nueva sociedad.

Esta explosión de la nueva información digital transformó y amplió considerablemente el alcance de las responsabilidades de los tecnólogos de la información. Constituiría un reto para ellos garantizar la disponibilidad, seguridad, privacidad y el cumplimiento de las normas en cualquier contexto. La complejidad de los entornos tecnológicos actuales ha aumentado considerablemente; si bien el volumen de información crece rápidamente, las administraciones siguen siendo las responsables de gestionar gran parte de esta.

Bajo estos preceptos, las tecnologías de la información deben poder utilizarse para administrarla de manera eficaz e inteligente.

Una forma eficiente de gestionar la documentación como un recurso global es desplegar una infraestructura de información que permita almacenar, proteger, optimizar y potenciar la información, de manera que se eviten los riesgos y se reduzcan a la vez los costos de inversiones en gestión y el crecimiento del presupuesto de las instituciones. Al mismo tiempo que se rentabilice al máximo su verdadero valor para el negocio. Superar este desafío resultó ser la incorporación de los Sistemas de Gestión Electrónica de Documentos (SGED) en el centro de la infraestructura de las TIC de cualquier empresa.

Un componente fundamental dentro de la gestión de documentos es la captura y mantenimiento de la información contextual asociada a estos últimos durante su ciclo de vida. Esta información vincula los documentos con el contexto administrativo y funcional (actividades y procesos) en que han sido producidos así como con otros documentos, constituyendo los metadatos parte esencial dentro de la misma[1]. Las definiciones más difundidas de este término los describen como datos sobre datos, informaciones sobre datos, datos sobre informaciones, información referente a información, entre otros. Los metadatos poseen gran importancia cuando se maneja documentación, a la vez que permiten verificar la autenticidad, fiabilidad e integridad de los documentos a lo largo del tiempo.

A partir del 2009, en la Universidad de las Ciencias Informáticas (UCI) se comenzó a desarrollar el Gestor de Documentos Administrativos eXcriba. Esta solución informática constituye hoy una de las alternativas cubanas para la gestión electrónica de documentos administrativos basada en normas, estándares y metodologías internacionales en el ámbito de la gestión documental. Ya en su versión 2.0 la solución cuenta con gran parte de las funcionalidades básicas recomendadas para una adecuada gestión documental, sin embargo, otras no menos importantes, como es el caso de la gestión de metadatos, han pasado a un segundo plano.

Inicialmente en el momento de creación de documentos en el sistema solo es posible describir los mismos mediante un conjunto de metadatos genéricos (nombre, título, autor, descripción y fecha de creación). Además, en aras de cumplir con las directrices propuestas por la norma ISAD-G[1] se

[1] Norma Internacional General de Descripción Archivística, que establece directrices para describir los documentos.

implementaron los componentes necesarios para que desde la propia interfaz web fuera posible describir los documentos a partir de los metadatos propuestos por dicha norma. Luego, con la incorporación de los módulos para la gestión del cuadro de clasificación y la gestión de expedientes se dio soporte a la aplicación para que permitiera gestionar los metadatos específicos para estas áreas. También es posible en el sistema clasificar los documentos a partir de las tipologías[2] documentales definidas, sin embargo, se imposibilita la descripción de dichos documentos a partir de los metadatos asociados a la tipología documental seleccionada. Una vez que son incorporados los documentos al sistema no existe ninguna funcionalidad que permita adicionarle nuevos metadatos.

Por otra parte, todas las organizaciones no manejan las mismas tipologías documentales para clasificar sus documentos y aun cuando diferentes organizaciones contemplen denominaciones similares puede darse el caso, que los metadatos asociados a dichas tipologías son diferentes por ejemplo, una minuta de reunión en una organización puede contemplar determinados metadatos mientras que en otra organización puede ser de interés describirla con otros. En el caso de los documentos que por sus características o condiciones de uso no constituyen documentos de archivo también se hace necesario darles un tratamiento documental mediante la descripción de metadatos específicos.

Lo anteriormente expuesto evidencia la dificultad que existe en la utilización y comprensión de la finalidad y el contexto en el que los documentos son creados. Se dificulta además el proceso de personalización de la solución en organizaciones en las cuales no se describen los documentos a partir de la norma ISAD-G. En estos casos, los desarrolladores deben hacer modificaciones al código fuente de la aplicación para adaptar el sistema a la norma que se emplea en la organización. Por otra parte, se dificulta el desencadenamiento de los procesos de gestión en los que interviene un documento, lo que imposibilita justificar su existencia, gestión y disposición a lo largo de su ciclo de vida. El sistema tampoco provee los mecanismos necesarios para recuperar los documentos a partir de los metadatos asociados a una tipología documental.

La situación antes descrita da lugar al siguiente **problema científico**: ¿Cómo facilitar los procesos de incorporación, asignación y uso de los metadatos asociados a los documentos producidos o incorporados al repositorio de contenidos del Gestor de Documentos Administrativos eXcriba?

[2]Forma específica o documental en la que se plasma o refleja una función, actividad o tarea de un sujeto productor. Ejemplos: carta, acta, informe, expediente[2].

Así mismo se asume como **objeto de estudio** la Gestión Documental.

El **objetivo general** de la investigación es desarrollar un módulo sobre el repositorio de contenidos del Gestor de Documentos Administrativos eXcriba que permita gestionar los metadatos, contribuyendo a que los documentos permanezcan accesibles y disponibles a lo largo del tiempo.

Como **campo de accción** se define: la gestión de metadatos para la descripción de documentos y se formulan las siguientes **tareas de investigación:**

1. Fundamentación de los elementos teóricos, conceptuales y metodológicos relacionados con la gestión documental y de metadatos.
2. Valoración de los mecanismos empleados por diferentes sistemas actuales para la gestión de metadatos, identificando sus dos componentes esenciales: modelo de datos y representación visual.
3. Selección de las metodologías, herramientas, tecnologías y lenguajes a utilizar durante el desarrollo del módulo para la gestión de metadatos.
4. Especificación de los requisitos que serán implementados en el marco de este trabajo.
5. Evaluación de la arquitectura del cliente web del GDA eXcriba.
6. Implementación de la infraestructura de servicios necesarios para la gestión de metadatos según los requerimientos identificados en el repositorio de contenidos del GDA eXcriba.
7. Incorporación de las funcionalidades identificadas para la gestión de los metadatos.
8. Diseño de los casos de prueba de caja negra y ejecución de las pruebas correspondientes.

Para guiar la presente investigación se utilizaron los siguientes **métodos científicos**:

- Métodos teóricos:
    - Histórico-lógico: posibilita mediante la búsqueda de bibliografía realizar un estudio de los fundamentos teóricos asociados a la gestión de metadatos en sistemas de gestión electrónica de documentos permitiendo que queden plasmadas las bases que sustentarán la presente investigación.

- Analítico-sintético: posibilita mediante el análisis de las definiciones y conceptos fundamentales descomponer la captura de metadatos de distintas soluciones de gestión electrónica de documentos, determinando características y las relaciones que se establecen entre ellas, logrando que su estudio fuera de fácil entendimiento y así poder realizar una mejor propuesta de solución que cumpla la situación problemática planteada.

Justificación de la Investigación:

Tradicionalmente, la actividad administrativa ha tenido un carácter documental, desde el punto de vista de que los documentos administrativos constituyen el testimonio de su actividad y son el soporte y la expresión externa de los actos de la administración. La necesidad de registrar los procesos administrativos seguirá siendo tan indispensable en el mundo digital tal y como lo ha sido en el mundo burocrático basado en el papel. Seguirá siendo necesario crear archivos que puedan servir como evidencia, que sean completos, manejables y accesibles a través del tiempo.

Razones de carácter administrativo, legal, económico o cultural justifican – además de su valor – la necesidad de organizar, describir y recuperar la información archivística. En este contexto, la creación o asignación de metadatos, la necesidad de etiquetar, describir o de definir aspectos informativos del documento electrónico junto al propio documento o de aportar datos sobre los propios datos es la solución más defendida en el ámbito de la información electrónica. Además es una tendencia teórica íntimamente relacionada a la recuperación de información y una solución técnica aplicable y adaptable a cada comunidad particular de información o a cada tipo de documento digital.

Según la norma ISO[3] (*International Organization for Standardization*) – 23081 cualquier sistema o combinación de sistemas de gestión documental que se use, debe ser capaz de usar y suministrar metadatos para gestionar documentos de forma responsable y efectiva. Los sistemas de gestión de documentos se deberían diseñar e implementar con la infraestructura necesaria para generar, incorporar y gestionar los metadatos correctamente y siempre que sea posible, realizarlo mediante procesos automatizados, asegurando que tanto los documentos como sus metadatos permanezcan accesibles, auténticos, fiables y disponibles frente a cualquier tipo de cambio de sistema.

[3]Organización Internacional de Normalización. Es una federación mundial de organismos nacionales de normalización.

El contenido de esta investigación se encuentra estructurado por cuatro capítulos. A continuación se muestra una breve descripción de cada uno:

Capítulo 1: fundamentación teórica, se describen los conceptos teóricos fundamentales que permiten entender temas relacionados con la gestión dinámica de metadatos. Se realiza un estudio de homólogos con el objetivo de identificar características y aspectos que puedan ser útiles en la propuesta de solución así como una breve descripción de la metodología, herramientas y tecnologías a usar para el desarrollo de la aplicación.

Capítulo 2: características del sistema, este capítulo tiene como objetivo describir detalladamente cómo se le irá dando respuesta a las peticiones del cliente, especificando para ello los requerimientos funcionales de la propuesta, así como los respectivos casos de uso que se generan y que darán solución al problema, sirviendo como punto de partida que permitirá ejecutar correctamente los próximos flujos de trabajo.

Capítulo 3: diseño del sistema, se procede a la realización de un análisis crítico de la arquitectura del GDA eXcriba, así como las relaciones que se establecen entre los componentes que conforman dicha arquitectura. Además, se elaboran los diagramas de clases del diseño, los cuales posibilitarán la construcción de las funcionalidades definidas en el capítulo anterior. Para la mejor comprensión de estos diagramas se describe detalladamente las clases fundamentales y los patrones de diseños empleados.

Capítulo 4: implementación y prueba, se representan los diagramas de despliegue y de componentes del módulo y una vez finalizada la implementación, es sometido mediante tres iteraciones a pruebas funcionales de caja negra empleando el método de partición equivalente, mostrándose los resultados obtenidos.

Capítulo 1

Fundamentación teórica

“Solo es útil el conocimiento que nos hace mejores”. Sócrates

Se presenta en este capítulo una guía que contiene los elementos teóricos que rigen la investigación, permitiendo la mejor comprensión de la gestión de metadatos a través de un estudio realizado a distintas soluciones para la Gestión Documental. Se describe además la metodología, tecnologías y herramientas que se utilizan durante el análisis, diseño e implementación de la solución.

1.1. Gestión Documental

La decisión de utilizar una solución de gestión documental se toma a menudo tras una crisis o una situación tensa causada por la gestión de la información, situación que requiere el establecimiento de una organización más estructurada, de una trazabilidad o una mejor usabilidad. La crisis se manifestaba de diferentes formas: la imposibilidad de encontrar la última versión de un documento en el que se han invertido días de trabajo y las dificultades causadas por el tiempo perdido. El tiempo perdido en búsquedas de documentos y la reutilización de documentos existentes también es problemática cuando se intercambian muchos documentos en la empresa y la eficacia del trabajo puede verse mermada por la falta de organización, lo cual evidencia pérdida de productividad[3].

En 1950 la *Federal Act* institucionalizó el término *records management* como forma idónea para el tratamiento documental en los Estados Unidos, definiéndola como el *área de la gestión administrativa general que se ocupa de garantizar la economía y eficiencia en la creación, mantenimiento, uso y disposición de los documentos a lo largo de todo su ciclo de vida, con el objetivo de asegurar una documentación adecuada, evitar lo no esencial, simplificar los sistemas de creación y uso del papeleo, mejorando la forma de cómo se organizan y recuperan los documentos*[2].

Luego, la ISO la define como el *área de gestión responsable de un control eficaz y sistemático de la creación, la recepción, el mantenimiento, el uso y la disposición de documentos de archivo, incluidos los*

procesos para incorporar y mantener en forma de documentos la información y prueba de las actividades y operaciones de la organización[4].

1.1.1. Sistemas informáticos para la Gestión Documental

El nacimiento de las tecnologías de la información y las comunicaciones trajo consigo la necesidad de capturar y conservar en formato electrónico los documentos. La generación y tramitación de los mismos en formato de papel que se originaban dentro de las organizaciones se multiplicaba considerablemente, apareciendo para dar respuesta a este problema los Sistemas de Gestión Documental. Estos sistemas representaron un nuevo salto en cuanto a la forma de pensar de los administradores y archiveros, ya que posibilitaba la disponibilidad y accesibilidad de un documento en forma fácilmente aprovechable, además de la reducción considerable del imparable incremento del cúmulo de papel.

Un sistema de Gestión Documental se define como un *sistema de información que incorpora, gestiona y facilita el acceso a los documentos a lo largo del tiempo*[4], para poder realizar estas funciones contiene aspectos de almacenamiento, recuperación, clasificación, seguridad, retención, distribución, creación y autenticación. Estos suponen un paso de avance para cualquier entidad debido a que proporcionan controles sobre la documentación con que esta cuenta, así como soporte digital para un adecuado almacenamiento sistemático, garantizando su recuperación de forma rápida y segura, además de que la misma no se deteriora como resultado de constante manipulación, evitándose así su pérdida. La implementación de estos sistemas trae también como beneficio que la información se pueda transmitir o intercambiar fácilmente.

1.1.2. Ventajas de un sistema de gestión de documentos

Teresa Allepuz Ros y Carmen Gutiérrez La Rubia consideran que las ventajas de implantar un SGED puede dividirse en tres grandes bloques[5]:

1. Estratrégicas, que incluyen consideraciones sobre la marcha general de la empresa y afectan a la mayoría de sus áreas, ejemplo de estas ventajas se encuentran:
    - Hacer más rápidos los procesos de trabajo en los que interviene el manejo de documentación.

- Asegurarse de que la documentación no es al factor que retrasa los procesos de trabajo.
- Incrementar la posibilidad de crecimiento, que está limitada por la capacidad de procesar más documentación en papel.
- Mejorar el servicio al usuario.
- Mejorar la satisfacción en el trabajo del personal.
- Mejorar la seguridad de los documentos contra incendios o inundaciones.

2. Financieras, que inciden en la reducción de costes y aumento de los ingresos:

   - Incrementar la productividad, reduciendo el tiempo necesario para realizar determinados procesos: reduciendo los recursos necesarios, produciendo más con los mismos o aumentando la calidad con los mismos recursos.
   - Ahorrar espacio físico, eliminando todos los espacios dedicados al archivo de documentos en papel.
   - Incrementar el volumen de negocio al poder poner los productos antes en el mercado, o al mejorar la calidad o al aumentar la capacidad de creación de nuevos productos.

3. Técnicos, que presentan la instalación de un SGED como la solución a una serie de problemas existentes dentro de la organización:

   - Deficiente gestión de la documentación en papel, que supone dificultades y demoras en la recuperacion de la información, documentos de gran valor que no son explotados en su totalidad, pérdida de documentos, entre otros.
   - Lenta comunicación de la información.
   - Duplicidad en el tratamiento de la información que es procesada en sistemas distintos.
   - Retrasos en el procesamiento de la información.

1.2. Metadatos

Como resultado del aumento de grandes cantidades de datos digitales disponibles en la red aparece la sobrecarga de información, situación por la cual se comienzan a usar los metadatos para facilitar la recuperación y catalogación de la misma. Existen diferentes definiciones para acuñar el término metadato, comúnmente son llamados datos sobre datos o información referente a información, otros conceptos se muestran a continuación:

Es la suma total de lo que se puede decir acerca de cualquier objeto de información. En este contexto, un objeto de información es algo que puede ser dirigido y manipulado como una entidad discreta por un ser humano o un sistema de información[6].

Es la información que permite clasificar los documentos en una estructura predefinida, limitar o expandir el acceso y uso, identificar las acciones de disposición o preservación previstas para uno determinado o recoger los eventos que han sucedido a un documento a lo largo del tiempo[7].

Son datos significativos que representan otros objetos de datos discretos, descripciones estructuradas de un objeto informático y estructuras de organización de la información – *legibles por una máquina* – cuya finalidad es hacer útiles los datos de distintas formas, según las necesidades concretas de cada servicio de información digital (los archivos digitales también) y según la aplicación que se les otorgue[8].

Los metadatos pueden ser definidos también como información estructurada que describe, explica, localiza o de lo contrario hace que sea más fácil recuperar, utilizar o administrar determinado recurso de información. Este término puede ser usado para referirse a información comprensible de una máquina, mientras que otros los utilizan solo para describir los registros electrónicos, aunque en ambientes bibliotecarios pueden ser comúnmente usados para la descripción de recursos que sean aplicables a cualquier tipo de objeto digital o no digital siendo la catalogación una forma de uso de los metadatos[9].

En el ámbito de la gestión de documentos, los metadatos se definen como datos que describen el contexto, contenido y estructura de los documentos, así como su gestión a lo largo del tiempo[4].

Según lo observado hasta ahora se puede hacer referencia a un metadato ya sea cuando se hable de un catálogo de biblioteca o cuando se quiera describir un sistema electrónico, por lo que se utiliza la definición de conjunto estructurado de datos cuyo objetivo es proporcionar información, para ello describen características,

estructuras y contenido de un determinado objeto de información, incrementando conocimiento de forma tal que se pueda administrar dicho objeto.

1.2.1. ¿Cómo categorizar los metadatos?

Existen diversidad de criterios en cuanto a las tipologías actualmente existentes de metadatos, porque estos dependen de muchos factores como el tipo de información que describen, el nivel de estructuración de esta información, el lugar donde se encuentren los metadatos, su ámbito de aplicación, el tipo de usuario que los utiliza y sus finalidades, entre otros. Por tanto, son múltiples también las clasificaciones de metadatos realizadas hasta ahora por diferentes autores, entre ellas se encuentran[9]:

- Metadatos descriptivos: describe un recurso con fines de descubrimiento e identificación, pueden incluir elementos tales como título, resumen, autor y palabras clave.
- Metadatos estructurales: facilitan la navegación y la presentación de los recursos además de proporcionar información sobre la estructura interna de los documentos, así como la relación entre ellos, ejemplos de estos están los rótulos de estructuración como página de título, tabla de contenidos, capítulos, partes, índice, fotografía de un periódico, entre otros.
- Metadatos administrativos: ayuda a la administración de determinado recurso, para ello responde preguntas de tipo cómo y cuándo fue creado el recurso, el tipo de archivo que posee este, así como otras técnicas para proporcionar información y acceso al mismo, ejemplos de estos tipos de metadatos se encuentran los datos técnicos tales como tipo y modelo de escáner, resolución, profundidad de bit, espacio de color, formato de archivo, compresión, fuente de luz, propietario, fecha del registro de derecho de autor, entre otros.

Al comparar los atributos de distintos esquemas[1], se distinguen dos tipos de metadatos[11]:

Metadatos intrínsecos: están incrustados dentro del documento, es decir, no pueden residir fuera del documento que describen, tanto el documento como el metadato presentan un vínculo inquebrantable al ser imposibles de modificar de forma separada, no se puede alterar uno sin que esto afecte al otro, los casos más comunes de este tipo son los que describen documentos generados en forma dinámica y con características

[1] Plan lógico que muestra las relaciones entre los distintos elementos de un conjunto de metadatos[10].

que varían frecuentemente a lo largo del tiempo. Incluyen atributos como: materia, título, autor, editor, lugar de publicación, fecha, tipo de objeto de información, forma del identificador (URN, ISBN), relación, fuente, idioma, cobertura, resumen, versión, notas, firma, clasificación, nivel de seguridad y descriptores.

Metadatos extrínsecos: estos tipos de metadatos para describir el objeto usan un enlace que persiste de forma separada, son flexibles y escalables a la hora de alterarlos, dado que las modificaciones que se efectúan sobre ellos no afectan a los datos principales. Abarcan información como: requerimientos del sistema, modo de acceso, accesibilidad, coste, control, extensión o tamaño del documento, descripción codificada y descriptores de la revisión.

Aunque existen varios enfoques en la tipificación de los elementos de metadatos, se reconocen generalmente cinco tipos de metadatos para describir un objeto de información digital. Se presentan a continuación dichos tipos de metadatos, prestando especial atención en su significado para la descripción archivística y la gestión de documentos electrónicos[8].

1. Metadatos descriptivos: son aquellos que dependen del propio documento y sirven para representar o identificar los objetos de información digital en su fase de organización, por ejemplo a través de instrumentos de descripción o URN *(Uniform Resource Name/Number)* y contemplan elementos de tipo: título, creador, identificador, entre otros.

2. Metadatos administrativos o metadatos para la gestión de recursos: se utilizan para la propia gestión y administración de los *Document Like Object*[2] (DLO por sus siglas en inglés). La importancia de este tipo de elementos reside en que permiten comprobar el mantenimiento, como por ejemplo el control de las distintas versiones de un registro.

3. Metadatos técnicos: son aquellos elementos creados por, o generados para, un sistema automatizado y se refieren al funcionamiento del sistema de *recordkeeping*[3], por ejemplo, datos sobre los tiempos de respuesta o los requisitos técnicos de acceso a un determinado tipo de información (autentificación, existencia de un *software* particular, entre otros.)

[2]Expresión ampliamente reconocida en la literatura sobre metadatos para aludir a los documentos de internet (texto, imagen, audio y video) y se utiliza para referirse a algo así como una unidad electrónica documental[8].

[3]Término que refiere a las políticas y procedimientos para gestionar los registros desde su creación hasta su distribución final[8].

4. Metadatos de uso: generalmente creados de forma automática, relativos al nivel de utilización y al tipo de usuarios de un determinado servicio de información.

5. Metadatos para la conservación: son aquellos metadatos destinados a gestionar la preservación de las fuentes de información para resolver los problemas básicos de la información digital (la fragilidad de los DLO, el problema de la obsolescencia de los medios informáticos, entre otros).

1.2.2. Funciones de los metadatos

Los metadatos tienen como función básica[6]:

1. La creación, multiversionado, la reutilización y recontextualización de objetos de información: los objetos entran en un sistema de información digital por ser creados digitalmente o por convertirse en formato digital. Las distintas versiones de un mismo objeto pueden ser usadas para la conservación, investigación, exhibición, difusión o incluso en el desarrollo de productos con fines opuestos.

2. Organización y descripción: una función primordial de los metadatos es la descripción y el orden de los objetos originales en un repositorio o en una colección, los objetos de información se organizan de forma automática o manual dentro de la estructura del sistema de información digital y pueden incluir descripciones generadas por el creador original, los metadatos adicionales pueden ser creados por profesionales de la información a través de los registros, las catalogaciones y los procesos de indexación.

3. Validación: los usuarios escudriñan los metadatos y otros aspectos de los recursos recuperados a fin de determinar la autoridad y la credibilidad de esos recursos.

4. Búsqueda y recuperación: la buena descripción de metadatos es esencial para buscar y recuperar información relevante a los objetos de información.

5. Utilización y preservación: los objetos digitales también tienen que estar sujetos a un régimen de conservación continua y se someten a procesos como la comprobación de integridad para asegurar su continua disponibilidad y para documentar cualquier cambio que pudiera haber ocurrido con el objeto de información durante los procesos de conservación.

6. Disposición: los metadatos son un componente clave en la disposición de la documentación, además ayudan a descubrir con que frecuencia se accede a una determinada información, ya sea a la que está disponible como la que se encuentra inactiva o ya no es necesaria y esta pueda ser descartada.

En resumen se plantea que poseen funciones como la de garantizar la autenticidad, disponibilidad de los datos y el acceso a los documentos a lo largo del tiempo de forma controlada y eficiente ya que se conoce con precisión el objeto descrito, también adhieren contenido, contexto y estructura a los documentos permitiendo el intercambio de información sin la necesidad de que implique el intercambio de los propios recursos. Los metadatos posibilitan además el uso de la documentación por parte de futuras generaciones, son esenciales para sostener el crecimiento y conservación de la información a mayor escala permitiendo búsqueda, integración y recuperación. Además, se pueden combinar para actuar como un conjunto de datos de identificación diferenciando un objeto de otro con fines de validación. Todo lo anterior mencionado se resume en tres funciones básicas:

- Gestión de datos.
- Acceso a datos.
- Análisis de datos.

1.2.3. Importancia de los metadatos

Los metadatos describen los atributos esenciales de un objeto de información asegurando que estos sobrevivan y sean accesibles en un futuro. Debido a que la información digital es frágil y puede ser dañada o alterada en cualquier momento los metadatos desempeñan un importante papel al poder garantizar la preservación de la información. Con el creciente número de recursos que en la actualidad son basados en la web, la cantidad innumerable de portales, sitios, páginas, sistemas que se agregan en internet y se construyen diariamente, se hace vital la adecuada gestión de los datos que estas contienen, siendo indispensable para ello el uso de metadatos. Estos últimos posibilitan la organización de los recursos contenidos en los sistemas, manteniendo los datos organizados y clasificados, permitiendo garantizar el acceso más eficiente a los grandes conjuntos de datos incluidos en los mismos.

Se concluye que los metadatos son elementos que poseen vital importancia ya que mediante su uso la búsqueda se hace eficaz al incrementarse la accesibilidad a los objetos de información. Permiten describir el contexto de los documentos y su relación con los demás objetos. Ayudan en la documentación, el mantenimiento, la autenticidad y el grado de integridad de los objetos de información, promueven también la mejora en el desarrollo de los sistemas, ya que se posibilita documentar los cambios en el uso de los contenidos y mediante este resultado se pueden retroalimentar las decisiones. Finalmente, se puede decir que los metadatos son necesarios a la hora de registrar cualquier proceso administrativo debido a que ayudan a la preservación y persistencia de cualquier objeto que sea descrito a través de los mismos, implicando que estos sean accesibles y manejables a lo largo del tiempo.

1.3. Metadatos para la gestión de documentos

La utilización de metadatos bien estructurados que especifiquen el contenido, estructura y contexto de los registros documentales, así como las necesidades esenciales de gestión constituyen el fundamento tecnológico de los sistemas de información constituidos en torno a los nuevos archivos digitales administrativos[8].

La gestión de los documentos implica de por sí la gestión de los metadatos ya que asegura la autenticidad, la fiabilidad, la disponibilidad y la integridad de los objetos de información a lo largo del tiempo, ya sean estos físicos o digitales posibilitando su gestión y comprensión. Los metadatos son la información asociada a los documentos gestionados, sirven para calificar los documentos, dotarlos de información complementaria utilizable, pueden utilizarse directamente como información, pero sobre todo, son la base de las funciones de búsqueda aplicadas a los documentos. Para contar con funciones avanzadas, un sistema de gestión electrónica de documentos (SGED) debe permitir asociar a los documentos informaciones estructuradas utilizables, siendo este el papel de los metadatos[12].

La necesidad de registrar los procesos administrativos sigue siendo tan indispensable en el mundo digital como lo ha sido siempre en el mundo burocrático basado en el papel, sigue siendo necesario crear archivos que puedan servir como evidencia, que sean completos, manejables y accesibles a través del tiempo. Una de las maneras de lograr este nivel de control y seguridad en el medio electrónico es poner en práctica estrategias de metadatos[8]. Con el fin de conservar los documentos es necesario crear metadatos

que faciliten el desencadenamiento de los procesos de gestión de documentos o la documentación de los mismos. En el contexto de los documentos electrónicos se debe ser cuidadosamente selectivo a la hora de usar los metadatos, ya que estos se encuentran de forma implícita dentro de los mismos y no se pueden deducir a simple vista, como es el caso de los documentos en papel, por esa razón los documentos electrónicos presentan fuerte dependencia de un contexto administrativo bien documentado y de la descripción aportada por los metadatos sobre cómo se producen los documentos.

Cuando se hace referencia a los documentos electrónicos la gestión de los metadatos se convierte en la verdadera gestión documental. Si se entiende como documentos electrónicos los ficheros que contienen información (un pdf, un excel, una imagen, un video, un registro de base de datos, entre otros) y se denomina metadatos a toda la información que es necesaria para gestionarlos, se percibe que cualquier proceso documental es en realidad un proceso de captura, gestión y explotación de metadatos[7]. En este ámbito los metadatos son parte imprescindible de los registros documentales, ya que cada documento electrónico depende de estos para garantizar tanto su acceso e intercambio como su existencia a través del tiempo. Por ese motivo se hace indispensable precisar, caracterizar, describir y estructurar los documentos electrónicos a través de los metadatos, permitiendo mejorar la gestión de la información de los procesos administrativos, lo que implica un mejor funcionamiento de los SGED.

Los metadatos para la gestión de documentos pueden usarse para identificar, autenticar y contextualizar tanto los documentos como agentes, procesos y sistemas que los crean, gestionan, mantienen y utilizan, así como las políticas que los rigen, definiendo a los documentos desde el mismo momento que son incorporados a este, fijándole en su contexto y estableciendo el control de su gestión, además de permitir que los documentos sean utilizados por una aplicación o sistema de información haciéndolos legibles y comprensibles. Por otra parte, los objetos de información no solo adquieren los metadatos en el momento en que son creados o incorporados al sistema, sino que a medida que se muevan a través de todo su ciclo de vida se le pueden ir añadiendo nuevas capas de metadatos, permitiendo estos últimos acumular y actualizar información para que puedan seguir utilizándose cuando no sean necesarios para la gestión pero sean conservados para facilitar la investigación[10].

1.3.1. Utilidad y beneficios de los metadatos para la gestión de documentos

Los metadatos apoyan los procesos de trabajo y los procesos de gestión de documentos[10]:

- Protegiendo los documentos como prueba y asegurando su accesibilidad y disponibilidad a lo largo del tiempo.
- Facilitando la comprensión de los documentos.
- Sirviendo de base y garantizando el valor probatorio de los documentos.
- Contribuyendo a garantizar la autenticidad, fiabilidad e integridad de los documentos.
- Respaldando la gestión del acceso, la privacidad y los derechos de propiedad intelectual.
- Sirviendo de base para una recuperación eficiente.
- Respaldando las estrategias de interoperabilidad, permitiendo que se incorporen oficialmente al sistema documentos creados en diversos entornos administrativos y técnicos y que se mantengan durante tanto tiempo como sea necesario.
- Proporcionando vínculos lógicos entre los documentos y su contexto de creación, manteniéndolos de forma estructurada, fiable e inteligible.
- Facilitando la identificación del entorno tecnológico en que los documentos digitales fueron creados o se incorporaron al sistema y la gestión del entorno tecnológico en el que se han mantenido, de modo que puedan ser reproducidos como documentos auténticos cuando se necesiten.
- Facilitando la migración eficiente y exitosa de documentos electrónicos de un entorno o plataforma informática a otro o cualquier otra posible estrategia de conservación.

Los metadatos son indispensables para cualquier entidad que desee gestionar adecuadamente y de manera eficaz su documentación, en la gestión de documentos pueden existir diferentes perspectivas entorno a los metadatos, una de ellas es en la gestión de la administración donde estos apoyan los procesos de negocio, el otro se encuentra en la gestión de documentos donde los metadatos capturan las

características de los documentos y su contexto y respaldan su gestión a lo largo del tiempo, por último, está la perspectiva de uso dentro y fuera del contexto de la administración donde posibilitan la recuperación, la comprensión y la interpretación de los documentos, siendo recomendable en todo momento que los metadatos no sean estáticos, que presenten una activo mantenimiento y estén actualizados para evidenciar la naturaleza de los cambios de las fuentes de información a las cuales los metadatos se refieren[10].

1.4. Estado actual de la gestión de metadatos en el GDA eXcriba

El Gestor de Documentos Administrativos eXcriba es una alternativa cubana para la gestión electrónica de documentos que surge como parte del proceso de informatización de la sociedad que se ha estado llevando a cabo en el país desde principios del siglo XX. Desde su concepción, se ha trabajado en el desarrollo de un producto genérico centrado en el cumplimiento de los requerimientos agrupados en las normas ISO-15489, ISAD-G, el estándar EAD[4] (*Encoded Archival Description*), así como en la especificación de Moreq[5] (Model Requirements for the Management of Electronic Records). La solución integral se compone de tres elementos fundamentales: físicos o de *hardware*, lógicos o de *software* y metodológicos.

La base tecnológica fundamental sobre la cual está implementada la solución es el ECM (*Enterprise Content Management*)[6] Alfresco. En esencia, Alfresco provee un repositorio de contenidos «en el cual se almacenan los documentos, sus metadatos y demás objetos de negocio» y un conjunto de servicios sobre dicho repositorio que permiten acceder y gestionar los contenidos. De esta manera, la mayor parte de las funcionalidades que hoy brinda el sistema se basan en los servicios de repositorio ajustados a determinadas necesidades. Muchas han sido las funcionalidades que sobre esta base se han incorporado a la interfaz web, sin embargo, otras no menos importantes han pasado a un segundo plano, tal es el caso de la gestión de metadatos.

La gestión de metadatos constituye una de las principales áreas por explotar de dicha solución. En aras de cumplir con las directrices propuestas por la norma ISAD-G desde un inicio se le dio tratamiento a la

[4] El EAD (Descripción Archivística Codificada) es un estándar para codificar instrumentos de descripción archivística[13].

[5] Moreq es una especificación formal de requerimientos para un SGED[14].

[6] Son las estrategias, métodos y herramientas utilizadas para capturar, administrar, almacenar, preservar y entregar contenido a los documentos relacionados con los procesos organizativos.

aplicación para que permitiera gestionar los metadatos especificados en dicha norma. Asimismo, es posible especificar y modificar un conjunto de metadatos que le son asignados a todos los contenidos una vez que se crean o incorporan al sistema, entre ellos, el nombre, título, descripción, autor y la fecha de creación. En otros casos «muy particulares» como los expedientes o demás elementos del cuadro de clasificación, también se le comenzó a dar soporte a los metadatos específicos para esta área.

Esto significa que una de las debilidades actuales que presenta el GDA eXcriba es que aún no tiene soporte para la gestión del resto de los metadatos definidos para cada una de las tipologías documentales especificadas en el repositorio. Aun cuando la norma ISAD-G constituye una guía oficial para la descripción de los documentos de archivo, cada administración «según su contexto y necesidades» define además un conjunto de metadatos que le impregnan información de algún valor a los documentos en algún momento de su ciclo de vida, principalmente a aquellos documentos que no son de archivos, pero que son necesarios durante los procesos de Gestión de Documental.

1.4.1. Gestión de metadatos en Alfresco

Como la base tecnológica del eXcriba es el ECM Alfresco representa vital importancia analizar la gestión de metadatos en esta herramienta.

Alfresco es un sistema de gestión de contenido empresarial de código abierto que incluye la gestión de documentos y la gestión de contenidos web[15], está desarrollado sobre la plataforma J2EE que ofrece diferentes productos y servicios integrados a su suite ECM: Alfresco Explorer, Alfresco Share, entre otros. Gracias al uso de componentes de código abierto en la conformación de su escalable arquitectura, Alfresco facilita la creación de aplicaciones centradas en el contenido, comúnmente de gestión documental, gestión de documentos de archivo, gestión de contenido web, entornos colaborativos, entre otros.

En el centro de la arquitectura de Alfresco se encuentra el repositorio de contenidos. Se trata de un conjunto de componentes y servicios que garantizan el almacenamiento, acceso, gestión y recuperación de los contenidos, sus metadatos y relaciones. Para ello, cuenta con un diccionario de datos que opera sobre dos elementos fundamentales: nodo y contenido. En este marco, contenido es todo portador de información, un objeto, un documento, una regla de negocio. Por otra parte, los nodos constituyen la

estructura de dato fundamental para el almacenamiento de los objetos. Cada contenido almacenado en el repositorio es representado a través de un nodo.

De esta manera, el repositorio de contenidos se convierte en un árbol de nodos, en el cual cada nodo soporta un conjunto de propiedades (metadatos) cuyos valores pueden ser de cualquier tipo de dato. Como mínimo, cada nodo tiene un nodo padre (excepto el nodo raíz del árbol) y a su vez tantos nodos hijos como sea necesario. Además, se pueden establecer relaciones entre los diferentes nodos. El propósito general de este tipo de estructura de datos es especificar cómo serán almacenados los nodos en el repositorio y en consecuencia, cómo operar con estos, de ahí la importancia del modelado de datos en el trabajo con los contenidos.

La gestión de metadatos en Alfresco, contempla dos componentes fundamentales: el modelado de contenidos – *a partir del cual se define la estructura de cada nodo en el repositorio de contenido* – y el mecanismo para la configuración de la interfaz web de la aplicación. Alfresco Explorer permite la personalización de la interfaz web en función de las necesidades de los clientes. Ninguna de las configuraciones requiere de implementación ni conocimientos técnicos de alto nivel; todas se hacen a través de ficheros XML[16].

En estos ficheros XML se pueden definir por cada tipo de contenido qué metadatos se desean gestionar desde la interfaz web y en qué componentes (combos de selección, botones de selección, cuadros de texto, entre otros) así como una configuración personalizada de dichos componentes. De esta manera, se garantiza una separación entre la estructura de los datos con la presentación de los mismos en la interfaz web, lo que supone un beneficio sustancial sobre todo cuando se deseen realizar modificaciones en alguno de los casos, en el modelo de datos o en la presentación[17].

1.5. Gestión de metadatos en los sistemas de Gestión Documental

Como Alfresco es la plataforma sobre la cual está desarrollada el Gestor de Documentos Administrativos eXcriba se analizan a continuación algunos sistemas similares a este gestor de contenido empresarial, con el objetivo de identificar las características que poseen estas soluciones para la gestión de metadatos y en función de esto ver los principales aportes a tener en cuenta para realizar una propuesta de implementación para la gestión de metadatos en el GDA eXcriba.

1.5.1. Nuxeo

Nuxeo es una empresa francesa que ofrece diversas soluciones de gestión documental (Nuxeo DM), gestión de activos digitales (Nuxeo DAM), gestión de casos (Nuxeo CMF), entre otros. Se trata de una solución completa de gestión de contenido empresarial, implementada sobre la plataforma Java. Posee una amplia gama de tipos de documentos, metadatos, flujos de trabajo avanzado, gestión de categorías, funciones de colaboración, búsqueda, gestión de contenido complejo (web, multiarchivos, estructurados) y gestión multibases[3].

La gestión de metadatos en Nuxeo DM se puede realizar desde la misma interfaz web por lo que proporciona mayor extensibilidad y flexibilidad al sistema. Por defecto posee una gran variedad de tipos documentales disponibles, así como un conjunto de componentes (combos de selección, radios y áreas de textos) que le conceden gran potencia a la interfaz. Además, permite la definición de objetos y colecciones documentales y gestionar contenidos sin la necesidad de incorporar ficheros al sistema, útil para soluciones que brindan servicios de gestión de documentos en formato duro[18].

1.5.2. Jahia

Jahia es una solución integrada de gestión de contenidos, que responde a las necesidades de gestión de contenido web y documental, distinguiéndose de otros por su interfaz de administración de contenidos incorporada al propio sitio. Además, permite gestionar manualmente los metadatos asociados a los documentos (categorías, palabras clave, entre otros) así como extraerlos automáticamente con el fin de incorporarlos al conjunto de estos definidos manualmente. Permite la búsqueda avanzada de archivos y capacidades de filtrado y un motor de reglas incorporadas para automatizar acciones en sus documentos[19].

Todos los contenidos en esta solución son tratados como objetos con propiedades que corresponden a un tipo. Con Jahia, cada tipo de contenido que se almacena en el repositorio se denomina nodo. Los nodos son objetos definidos por su tipo, el cual establece a su vez las propiedades que un nodo puede tener. Por cada tipo de nodo puede haber un número ilimitado de nodos creados en el repositorio[20].

1.5.3. Knowledge Tree

Es una solución de Gestión Electrónica de Documentos desarrollada por la sociedad sudafricana JamWarehouse. Es un gestor documental capaz de ser integrado mediante servicios web y cuenta con varios mecanismos de diseminación de información. Posee una gestión de documentos simple y eficaz, destacándose varios puntos positivos, tales como una búsqueda avanzada que ofrece respuestas satisfactorias, modos de navegación virtual además de gestionar los metadatos desde aplicaciones ofimáticas.

La gestión de metadatos se realiza desde la propia interfaz web. Por defecto todos los documentos tienen asociados una nube de etiquetas que acepta varias palabras clave que el usuario elige para describirlos. El tipo de metadato que un usuario puede agregar a un documento se basa en el tipo de documento. Cuando un documento se agrega al repositorio, el usuario tiene que elegir que tipo de documento es y en función de esto serán los metadatos que contendrá dicho documento[21].

Cada tipo de documento se puede asociar con uno o más conjuntos de campos (*fieldsets*) y cada *fieldset* a su vez puede estar asociado con cualquier cantidad de tipos de documentos. En los *fieldsets* se almacenan los metadatos que contienen los documentos que pueden ser de dos tipos: genéricos y específicos. Los genéricos forman parte de todos los documentos en el repositorio mientras que los específicos solo se utilizan cuando se asocian con un tipo de documento[22].

1.5.4. Resultado

Knowledge Tree posee un conjunto de metadatos genéricos y específicos, donde los genéricos forman parte de todos los documentos en el repositorio mientras que los específicos solo se utilizan cuando se asocian con un tipo de documento. Esto es un aspecto clave que posibilita una visión de los requisitos que se deben capturar para permitir la correcta gestión de los metadatos en el eXcriba, lo cual es muy útil ya que si en determinado momento algún usuario necesita describir un único documento y no todos en el sistema tendrá la posibilidad de hacerlo, así como cuando estos metadatos ya no fueran necesarios para ese documento se le permitirá eliminarlos. Lo explicado anteriormente da lugar a la definición de requisitos para el módulo que se pretende desarrollar, requisitos que serán descritos detalladamente en próximos capítulos.

Por otra parte Jahia y Nuxeo posibilitan el correcto diseño de los prototipos de interfaces que serán desarrollados para los usuarios finales, al analizar cómo estos sistemas realizan el manejo de la información de entrada, el tipo de información manipulada y las distintas salidas que se producen, prestando esencial atención en los componentes de interfaces (combos de selección, radios, áreas de texto, entre otros) que se utilizan para la interacción usuario-sistema. Estas dos herramientas permiten de manera general tener una visión de cómo se implementan las vistas a mostrar, de modo que las interfaces que se desarrollen para el módulo de gestión de metadatos en el eXcriba sean lo más simples posibles y que el contenido que se manipule sea el que el usuario necesite.

1.6. Tecnologías, herramientas y metodología

1.6.1. Lenguajes de programación

1.6.1.1. PHP

PHP*(Hypertext Preprocessor)*: Es un lenguaje de "código abierto interpretado", de alto nivel, embebido en páginas HTML y ejecutado en el servidor. La mayoría de su sintaxis es similar a C, Java y Perl y es fácil de aprender. PHP es un lenguaje de programación simple y eficiente, hecho pensando en la web[23].

Se decide utilizar PHP ya que es el lenguaje propuesto por la arquitectura del sistema sobre el cual se ha desarrollado el cliente web de eXcriba, por lo que al desarrollarse las diferentes funcionalidades que le darán cumplimiento a la presente investigación se debe seguir esta restricción de diseño, para que la integración de estas funcionalidades al sistema se realice de manera satisfactoria y el resultado obtenido sea el esperado.

1.6.1.2. JavaScript

La API de JavaScript de Alfresco provee una de las interfaces de programación más usadas en la actualidad para el desarrollo de aplicaciones centradas en el contenido sobre el repositorio de contenidos del ECM Alfresco. Esta API permite personalizar y desarrollar nuevas funcionalidades de manera más sencilla y rápida que si se hiciera con Java. A diferencia del JavaScript del lado del cliente – *aquel que se*

ejecuta en el navegador – la API de JavaScript es una implementación de JavaScript del lado del servidor implementada con el motor de JavaScript Mozilla Rhino[7][25].

Alfresco extendió la implementación de Rhino con algunos objetos globales y funciones que permiten manipular los diferentes objetos del repositorio. Otro elemento importante a tener en cuenta es que la ejecución de un servicio sobre JavaScript implica el uso de los servicios de transacciones y de seguridad, lo que significa que se mantiene intacto el modelo de seguridad y además que si durante la ejecución del servicio algunas de las operaciones dentro de este fallan, entonces el estado final del sistema será el mismo que tenía antes de comenzar la ejecución del servicio.

En el contexto del presente trabajo de diploma, aunque los servicios fueron implementados en Java, se hizo una revisión de las diferentes API de JavaScript que provee Alfresco así como de los servicios implementados por el equipo de desarrollo. El objetivo fundamental de esta revisión fue totalmente identificativo, o sea, verificar la posibilidad de manipular el diccionario de datos de Alfresco a través de alguna de las API o de los servicios implementados por el equipo de desarrollo del GDA eXcriba.

1.6.1.3. Java

Java es un lenguaje de programación creado por Jim Gosley, en concordancia con numerosos autores y desarrolladores experimentados en este lenguaje, es un potente lenguaje de programación orientado a objetos que además se caracteriza por ser distribuido, enfocado en la seguridad, de arquitectura neutral y portable. Este compendio de buenas características potenciales de Java es lo que hace que sea uno de los lenguajes preferidos para el desarrollo de aplicaciones empresariales de alto nivel: tal es el caso de Alfresco[25].

Sin embargo, en lo que a este trabajo respecta, Java se usa para la implementación de webcripts, o sea, webscripts cuyo controlador está implementado en Java. Para desarrolladores identificados con el negocio cabría lugar a una pregunta como la siguiente ¿por qué usar Java para implementar el controlador cuando la API de JavaScript parece cubrir todos los requerimientos? El hecho es que Java se usa en este contexto cuando:

[7] Rhino es una implementación de código abierto de JavaScript escrito enteramente en Java. Normalmente se incrusta en las aplicaciones Java para ofrecer secuencias de comandos para los usuarios finales[24].

- El acceso a determinados servicios no está disponible desde la API de JavaScript que provee Alfresco, tal es el caso del manejo del diccionario de datos.
- Interactuar con sistemas cuyas API solamente están expuestas vía Java.
- El rendimiento es absolutamente crítico.

A diferencia de la API de JavaScript, con Java sería posible acceder directamente a las API y servicios que ofrece Alfresco para la manipulación de los diferentes objetos del repositorio y no solamente al subconjunto de interfaces disponibles a través de JavaScript. En su lugar, se emplean e implementan librerías para darle cumplimiento a los requerimientos identificados.

1.6.2. Marcos de trabajo

1.6.2.1. jQuery

jQuery es un *framework* – *conjunto de herramientas* – para el lenguaje JavaScript de código abierto creado por John Resig que sirve como base para la programación avanzada de aplicaciones. Implementa un conjunto de clases que permiten al desarrollador abstraerse de uno de los principales problemas a los que se enfrenta el desarrollo de una aplicación web: la incompatibilidad entre navegadores. Otra de las características favorables a su elección es el tamaño de la librería, tamaño bastante razonable, que no retrasa mucho la carga de las páginas[26].

jQuery soluciona algunos problemas que se presentan frecuentemente en el desarrollo de aplicaciones web actuales, entre ellos: acceso a parte de la página, modificación de la apariencia de la misma, alterar el contenido de una página o parte de ella, responder a la interacción de los usuarios con la aplicación, incorporar animaciones, obtener información del servidor sin refrescar la página, simplificación de algunas de las operaciones comunes que se usan en JavaScript[27].

Aprovechando estas y otras características de jQuery, la mayor parte de la capa de presentación del GDA eXcriba está escrita o hace uso de este *framework*. A partir de un conjunto de sugerencias realizadas por el equipo de desarrollo especializado y de una evaluación y análisis crítico de las características del *framework* y mediante las necesidades identificadas se comprobó que efectivamente jQuery podía ser utilizado para el desarrollo de las funcionalidades que se incorporarían con el nuevo módulo.

1.6.2.2. WebScript

Un webscript es simplemente una URI unida a un servicio usando algunos de los métodos HTTP estándar: GET, PUT, POST, DELETE. Alfresco WebScript fue introducido oficialmente en el año 2006 como parte de la arquitectura del sistema (Alfresco) y desde entonces ha obtenido popularidad entre los desarrolladores e integradores de sistemas que usan Alfresco como Gestor de Contenido Empresarial, implementando nuevos servicios sobre su repositorio. Tanto es así que las últimas implementaciones y proyectos de Alfresco, entre ellas Surf, Share, Web Studio y los servicios CMIS, han sido desarrollados usando webscripts[28].

Son precisamente, la elegancia, simplicidad y usabilidad que proporciona este *framework* en el desarrollo de servicios las razones principales por las que el arquitecto del proyecto – *con el apoyo del equipo de desarrollo* – excitó el desarrollo de una capa de comunicación con el repositorio de contenidos a través de servicios REST (*Representational State Transfer*)[8] antes del lanzamiento de la segunda versión de la solución: “eXcriba 2”. De esta manera, se convertiría en una restricción de diseño heredada para todos los desarrollos posteriores hacer uso de esta capa y por otra parte de las potencialidades que suministran los servicios REST. En estas condiciones, para el desarrollo de los servicios identificados se propone la implementación y uso de servicios REST a través del *framework* WebScript[17].

1.6.2.3. CodeIgniter

CodeIgniter, originalmente desarrollado por Rick Ellis, es un *framework* para desarrollar aplicaciones o sitios web usando PHP. Su objetivo es facilitar el desarrollo de proyectos mucho más rápido que lo que se podría hacer comenzando desde cero, proveyendo un conjunto de bibliotecas para tareas comunes, una interfaz sencilla y una estructura lógica para acceder a esas bibliotecas[30]. Se decide usar este marco de trabajo ya que es una restricción de diseño heredada, es decir, fue propuesto en la arquitectura del sistema por estar bien equipado para el desarrollo de aplicaciones escritas en el lenguaje usado: PHP.

[8]Término introducido en la tesis doctoral de Roy Fielding – uno de los principales autores de la especificación de HTTP– en 2000. REST o la Transferencia de Estado Representacional es es un estilo de arquitectura de software para sistemas hipermedias distribuidos tales como la web[29].

1.6.3. Herramientas

1.6.3.1. Visual Paradigm

Para realizar el modelado del sistema se utilizó como herramienta CASE Visual Paradigm para UML (VP-UML). VP-UML ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del *software* a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de utilidad para el analista. Diseñada para una amplia gama de usuarios interesados en la construcción de sistemas de forma fiable a través de la utilización de un enfoque Orientado a Objetos[31].

1.6.4. Metodología

1.6.4.1. RUP (*Rational Unified Process*) con nivel 2 de CMMI

El Proceso Unificado es un proceso de desarrollo de *software* que se distingue por tres aspectos fundamentales: dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. RUP divide el desarrollo del *software* en cuatro fases[32]:

- Inicio: el objetivo de esta fase es desarrollar la visión del proyecto.
- Elaboración: en esta fase se especifican la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema.
- Construcción: completar el desarrollo del sistema basado en la línea base de la arquitectura. El objetivo de la misma es obtener la Capacidad Operacional Inicial.
- Transición: se corrigen los problemas e incorporan mejoras sugeridas por los usuarios, en sí se garantiza que el *software* esté listo para entregar al usuario. El objetivo es conseguir el release del producto.

RUP es una infraestructura flexible de desarrollo de *software* que proporciona prácticas recomendadas probadas y una arquitectura configurable. Es un proceso práctico que posee nueve flujos de trabajo, seis conocidos como flujos de ingeniería (modelado de negocio, requisitos, análisis y diseño, implementación,

prueba y despliegue) y tres como flujos de apoyo (gestión de la configuración y los cambios, gestión de proyecto y gestión del entorno) en el que se definen[32]:

- Trabajador: papel que un individuo puede desempeñar en el desarrollo de *software* .
- Actividad: es una unidad de trabajo que se asigna a un trabajador.
- Artefactos: es un término general para cualquier tipo de información creada, producida, cambiada o utilizada por los trabajadores en el desarrollo del sistema.

Se decide utilizar RUP ya que es la metodología usada para el desarrollo de *software* en el proyecto eXcriba. Además, RUP se caracteriza, como se explica anteriormente, por ser iterativo e incremental, estar centrado en la arquitectura y guiado por casos de uso, elementos importantes para el éxito tanto del proyecto como del producto. Además, RUP actualmente en la Universidad de las Ciencias Informáticas (UCI) presenta un proceso de mejora basado en el segundo nivel de CMMI[9] (*Capability Madurity Model Integration*), el cual posibilita que la institución como productora de software que es, sea reconocida internacionalmente al desarrollar productos con mayor calidad y eficiencia. Esta metodología complementada con el segundo nivel de CMMI será una excelente combinación que garantiza tener bien documentado el producto así como reducir riesgos que puedan existir en el desarrollo del mismo.

1.6.5. Tecnologías

1.6.5.1. FreeMarker

FreeMarker es un motor de plantillas[10] de código abierto creado por Benjamin Geer y Mike Bayer. Comparado con otros lenguajes como PHP o JSP, FreeMarker como lenguaje de plantillas que es, no permite la implementación de procedimientos de negocio complejos dentro de sus plantillas, de hecho, solamente incluye un conjunto de instrucciones o directivas que proveen elementos básicos de programación (condicionales, asignación de variables, iteración sobre listas)[28].

[9]Modelo de Capacidad y Madurez Integrado. CMMI es un modelo para la mejora de procesos, que proporciona a las organizaciones, los elementos esenciales para procesos eficaces.

[10]Una plantilla de presentación es un documento de texto que, aplicado a un modelo de datos produce una salida (un nuevo documento) que tiene la estructura definida en la plantilla.

Este motor es usado por el framework WebScript de Alfresco para proveer un mecanismo de generación de respuestas preformateadas una vez finalizada la ejecución de un webscript, o sea, una vez finalizada la ejecución de una lógica de negocio se emite una respuesta cuya salida queda determinada por la estructura de la plantilla de presentación correspondiente al servicio en ejecución.

Una de las principales ventajas que usa el *framework* WebScript de este motor de plantillas es que permite generar diferentes vistas y representaciones como resultado de la ejecución de un mismo servicio. De esta manera, se logra la separación definitiva entre la lógica de negocio que se ejecuta durante una petición y el formato de salida de la respuesta generada[17].

A través de los resultados arrojados por el estudio del análisis bibliográfico que se expone en este capítulo, se pudo evidenciar los aspectos principales que comprenden la gestión de metadatos y su estrecha relación con la gestión documental, presentándose el marco teórico a partir del cual se solidifica el proceso de desarrollo logrando que se cumplan los objetivos planteados. Se especifican además algunas de las características de las herramientas, tecnologías y metodología a usar en el desarrollo del *software* permitiendo establecer los cimientos que fundamentan la propuesta de solución.

Capítulo 2

Características del sistema

"Dedicamos mucho tiempo – la mayoría del esfuerzo del proyecto – no a la implementación ni a las pruebas, sino a tratar de decidir qué es lo que se va a construir". Brian Lawrence

La captura de requisitos es el proceso de averiguar, normalmente en circunstancias difíciles, lo que se debe construir[32]. En el presente capítulo se exponen las características del sistema a implementar, describiéndose la propuesta de solución y el contexto en el que se desarrolla el problema mediante un modelo de dominio. Se identifican además, los requisitos funcionales y las técnicas utilizadas para su obtención así como la especificación de casos de uso del sistema.

2.1. Propuesta de solución

A continuación se propone un conjunto de cambios que deben llevarse a cabo para solventar los problemas derivados de la ineficiente gestión de metadatos en el GDA eXcriba. Como parte de la propuesta se decidió, teniendo en consideración criterios de la norma ISO-23081, dividir el proceso en dos momentos fudamentales:

- Momento de la incorporación de los documentos.
- Posterior a la incorporación de los documentos.

2.1.1. Momento de incorporación de los documentos

En el momento de incorporación de los documentos al sistema, los metadatos pueden incluir, según sea el caso y las necesidades, información sobre el contexto de creación del documento, de la organización y de los agentes implicados entre otros atributos relativos a la tipología documental, estructura y formato. Estos metadatos permitirían entender el entorno en que los documentos fueron creados, con qué finalidad y la actividad que se llevó a cabo durante su producción. De esta manera los metadatos formarán parte integral

de los documentos generados[10]. Por la importancia y las características que le confieren estos metadatos a los documentos se propone:

- Tal y como se realiza actualmente, el sistema debe permitir la clasificación de los documentos a partir de alguna de las tipologías documentales definidas, esto es, durante la creación o incorporación del documento al sistema, el usuario tendrá la posibilidad de definir la tipología documental a la que responde dicho documento, para ello primeramente sería necesario definir el conjunto de tipologías documentales que se producen y manejan en la organización.

- Descripción de los metadatos según la tipología documental seleccionada por el usuario. Una vez seleccionada la tipología documental, el sistema debe proveer un listado de los metadatos asociados a tal tipología de modo que sea posible incluir la información o atributos relativos al contexto de creación, de la organización y de gestión del mismo.

- Se propone además, a petición de algunos clientes, entre ellos los especialistas del Archivo Nacional de Cuba, ajustar el mecanismo para describir los documentos a partir de la norma ISAD-G. A su consideración, los documentos no necesariamente deben incluir los metadatos asociados a esta norma desde el propio momento de creación del documento, en su lugar, estos metadatos deben ser añadidos paulatinamente, a medida que el documeto transite por su ciclo de vida. Como beneficio, la solución a este problema será el punto de partida para la incorporación de nuevas normas de descripción de metadatos.

2.1.2. Posterior a la incorporación de los documentos

Los procesos de gestión de documentos llevados a cabo sobre uno o un conjunto de estos, deben quedar documentados con el fin de garantizar la fiabilidad de los mismos al tiempo que facilitan su desencadenamiento. Estos metadatos, llamados metadatos sobre actividades de gestión, incluyen información sobre los procesos de gestión que han sido o serán aplicados a cada uno de los documentos. Con vista a asegurar la autenticidad, fiabilidad, disponibilidad e integridad de los documentos a lo largo del tiempo se propone:

- Facilitar la incorporación de metadatos a los documentos en algún momento de su ciclo de vida posterior a la creación o incorporación del mismo al sistema, contribuyendo de esta manera a la conservación de dichos documentos. Del mismo modo será necesario que los usuarios tengan la posibilidad, según el permiso conferido, de modificar los valores de los metadatos asociados a cada uno de los documentos.
- Permitir la eliminación de metadatos asociados a los documentos, esto es, que el usuario pueda eliminar un conjunto de metadatos asociados a un documento una vez que este o un comité de expertos determine que dichos metadatos ya no son, ni serán útiles.

Para llevar a cabo la propuesta, se decidió hacer uso de dos conceptos que se introducen en el diccionario de datos de Alfresco: tipos de contenido y aspectos. Los tipos de contenidos se usarán para la clasificación de cada uno de los contenidos (documentos o carpetas). Los aspectos, en cambio, permitirán asociarle a los contenidos un compendio de metadatos ya sea en el momento de la creación de los documentos como en la gestión.

2.2. El contexto del problema

2.2.1. Modelo de dominio

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema[32].

En la Figura 2.1 se representa el modelo de dominio construido mediante un diagrama UML, similar a como se modelan los diagramas de clase. El propósito que persigue el mismo es identificar las entidades del negocio relevantes en el contexto que se desarrolla el problema, además de los eventos que se desencadenan alrededor de la gestión de metadatos en el sistema eXcriba.

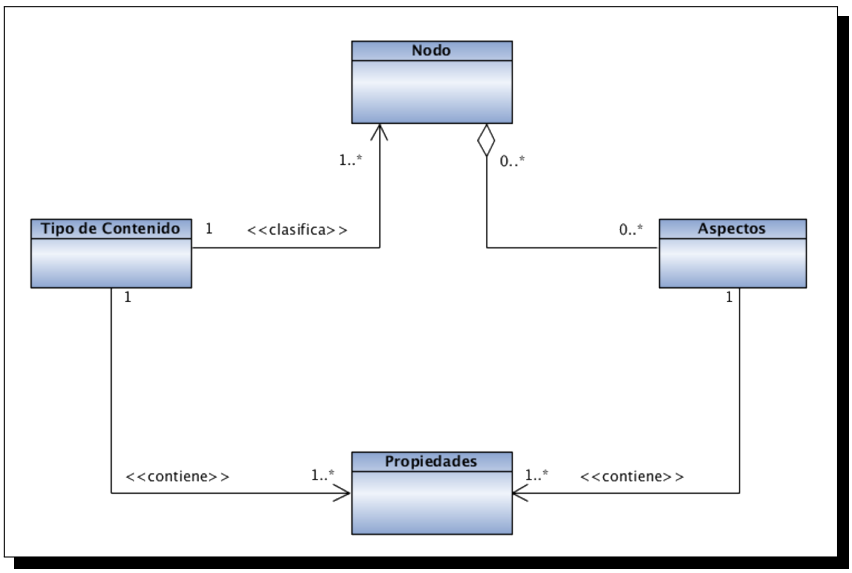


Figura 2.1: Diagrama del modelo de dominio

Los nodos se clasifican a partir de los tipos de contenido, para los cuales se definen un conjunto de propiedades que se le añaden automáticamente a los nodos una vez clasificados los mismos. Además, se le pueden añadir nuevos metadatos a partir de la definición de aspectos.

2.2.1.1. Descripción de las clases del modelo de dominio

- **Nodo:** es la estructura de datos que usa Alfresco para gestionar los contenidos. En el contexto de este trabajo un contenido solo hará referencia a documentos o carpetas.
- **Tipo de contenido:** es un concepto que introduce el diccionario de datos de Alfresco para clasificar los nodos.
- **Aspectos:** es un concepto que introduce el diccionario de datos de Alfresco para adicionar nuevos metadatos en tiempo de ejecución.
- **Propiedades:** son los metadatos que se le asocian a los nodos por dos vías: tipos de contenido o aspectos.

2.3. Captura de requisitos

Un requisito es algo que el sistema tiene que hacer o una cualidad que debe tener. Un requisito existe ya sea porque el tipo de producto exige ciertas funciones o cualidades o porque el cliente quiere que el requisito sea parte del producto entregado[33]. La especificación de requisitos de software establece la base para el acuerdo entre los clientes y los desarrolladores, de lo que deberá hacer el producto de software, así como lo que no se espera que este haga.

2.3.1. Definición de las técnicas de obtención de requisitos

Los requisitos son la base para todo lo que hay que seguir en el ciclo de desarrollo de un producto, por lo tanto, es lógico pensar que estos deben ser entregados a diseñadores y desarrolladores correctamente descritos, de ahí la vital importancia del uso de una adecuada técnica para la captura de requisitos de manera eficaz, que ayude a obtener una visión suficientemente buena del sistema. Las técnicas utilizadas para la captura de requisitos del módulo son las siguientes:

1. Sistemas existentes: esta técnica consiste en analizar distintos sistemas ya desarrollados que estén relacionados con el sistema a ser construido. Se pueden analizar las interfases de usuario, observando el tipo de información y cómo se maneja. También es útil analizar las distintas salidas que los sistemas producen (listados, consultas), porque siempre pueden surgir nuevas ideas sobre la base de estas. Esto puede ser útil para descubrir requisitos importantes, que tal vez el cliente o el usuario hayan fallado en comunicar[34].

2. Brainstorming (Lluvia de ideas): una sesión de tormenta de ideas es una reunión de personas interesadas, cuya misión es generar nuevas ideas para el producto final. Esta técnica aprovecha el efecto de grupo, es decir, reúne a un grupo de personas para generar tantas ideas como sea posible para el nuevo producto. Las ideas que se exponen en esta técnica son todas aceptables y siempre debe existir el espacio para criticarlas o debatirlas[33].

3. Prototipos: durante la actividad de extracción de requerimientos, puede ocurrir que algunos requerimientos no estén demasiado claros o que no se esté muy seguro de haber entendido

correctamente los requerimientos obtenidos hasta el momento, todo lo cual puede llevar a un desarrollo no eficaz del sistema final. Entonces, para validar los requerimientos hallados, se construyen prototipos, los cuales son simulaciones del posible producto que luego son utilizados por el usuario final[35].

Estas tres técnicas combinadas dieron lugar a la captura de los requisitos del módulo. Primeramente se hizo un estudio de homológos, donde se realizó un análisis de como se gestionan los metadatos en varios sistemas de gestión electrónica de documentos. Este estudio posibilitó observar como la mayoría de estos sistemas definen metadatos para describir sus documentos tanto genéricos como específicos, elemento esencial que dió lugar a la obtención de importantes requerimientos funcionales. Lo explicado anteriormente evidencia el uso de la técnica de sistemas existentes.

Posteriormente de acuerdo con Suzanne y James Robertson – *quienes recomiendan que todos los proyectos debieran programar al menos una sesión de tormenta de ideas* – se realizó una lluvia de ideas con varios participantes del sistema eXcriba, tales como el arquitecto, un analista y dos desarrolladores. En esta reunión surgieron nuevas ideas que se convirtieron en nuevos requisitos a incorporarle al módulo, además de que se tomaron sugerencias de como serían las interfaces de usuario donde se manejará la información que será expuesta al usuario final.

Finalmente con los requerimientos encontrados mediante el uso de las dos técnicas mencionadas anteriormente se hizo un esbozo de los prototipos de usuario, mediante estos prototipos se consiguió una importante retroalimentación en cuanto a si el sistema diseñado en base a los requerimientos recolectados le permite al usuario realizar su trabajo de manera eficiente y efectiva, dando lugar a la aparición de nuevos requisitos que pasaron desapercibidos. Concluida la aplicación de esta técnica se formalizaron los requisitos funcionales, los cuales se enuncian en el apartado siguiente.

2.3.2. Especificación de requisitos funcionales

Los requerimientos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de como se debe comportar en situaciones particulares, es decir, estos describen lo que el sistema debe hacer[36].

Con el objetivo de presentar los requisitos funcionales se realiza la especificación de los mismos, definiendo de cada requisito la prioridad del mismo otorgada por el cliente y la complejidad del desarrollo de dicha funcionalidad. Dichas especificaciones se muestran a continuación.

No.	Nombre	Descripción	Prioridad	Complejidad
R1	Mostrar metadatos de un contenido.	El sistema permite al usuario visualizar los metadatos asociados al elemento sobre el cual está trabajando, ya sea un documento o una carpeta. La información que mostrará el sistema como resultado de esta acción serán las propiedades asociadas al documento o carpeta según su tipo de contenido.	Alta	Baja
R2	Editar metadatos de un contenido.	El sistema muestra un formulario al usuario donde se podrá editar los valores de las propiedades que se asocien al elemento, siempre y cuando se tenga permisos para realizar la operación. La información que mostrará el sistema como resultado de esta acción serán las propiedades asociadas al elemento con las modificaciones realizadas.	Alta	Baja
R3	Adicionar aspecto a un contenido.	El sistema permite que el usuario si tiene los permisos requeridos pueda adicionar metadatos al documento o carpeta según estime conveniente, esta acción se realizará siempre que exista un aspecto asociado al documento o carpeta seleccionada previamente.	Baja	Media

R4	Eliminar aspectos de un contenido.	El sistema permite que el usuario si tiene los permisos requeridos, pueda sustraer propiedades a un documento o carpeta cuando las mismas no sean necesarias.	Baja	Media
R5	Mostrar aspectos.	El sistema posibilita que se listen los aspectos existentes en el repositorio.	Baja	Baja

Tabla 2.1: Especificación de requisitos

Para visualizar los prototipos de interfaz de los requisitos funcionales remitirse al **anexo A** de este documento.

2.4. Especificación de casos de uso

2.4.1. Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema muestra las funcionalidades que el sistema debe tener para aportar un resultado de valor para sus actores. A partir de los requerimientos que han sido identificados en el apartado 2.3 se propone el diagrama que se muestra a continuación, el cual es una representación gráfica de los actores y los respectivos casos de uso que estos inician, de esta manera queda plasmado la forma en que es usado el sistema por los actores.

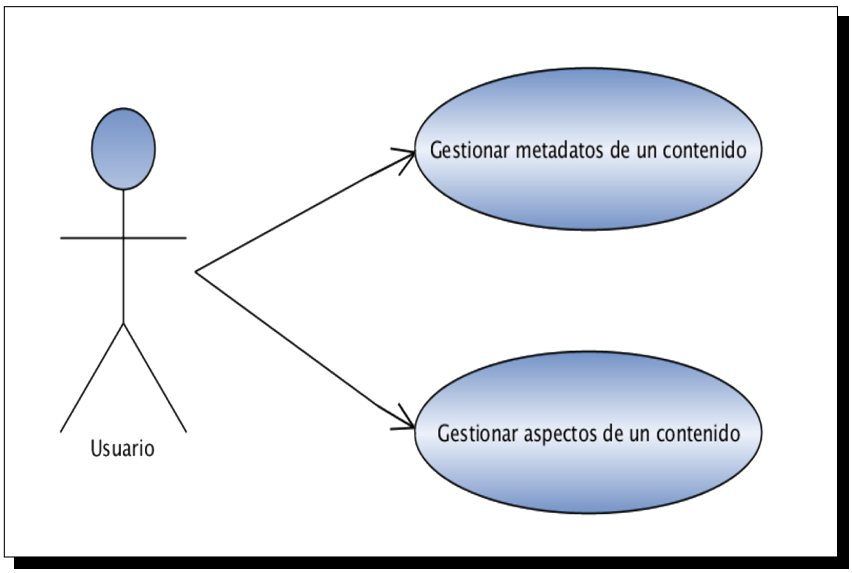


Figura 2.2: Diagrama de casos de uso

2.4.2. Definición de actores del sistema

Actor	Justificación
Usuario	Es la persona que visualiza o modifica los metadatos asociados a un documento o carpeta específico del sistema, así como icorpora o elimina nuevas propiedades a este.

Tabla 2.2: Definición de actor del sistema

2.5. Descripción de los casos de uso

Caso de uso	**Gestionar metadatos de un contenido**
Actor	Usuario

Resumen	El caso de uso inicia cuando el usuario procede a obtener o modificar los metadatos que le son asignados a un documento o carpeta del sistema y finaliza cuando se ejecuta la operación deseada.
Complejidad	Baja
Prioridad	Alta
Referencias	R1, R2
Precondiciones	El usuario ha sido autenticado en el sistema.
	El usuario tiene los permisos necesarios para realizar la acción.
Poscondiciones	
Flujo de eventos	
Sección: “mostrar metadatos asociados a un contenido”	
Acción del actor	**Respuesta del sistema**
1. Selecciona el documento o carpeta del cual desea visualizar sus metadatos y escoge la opción “Propiedades” en la barra lateral derecha.	2. Muestra los metadatos asociados al documento o carpeta seleccionada.
Sección:“editar metadatos asociados a un contenido”	
Acción del actor	**Respuesta del sistema**
1. Selecciona el documento o carpeta del cual desea editar sus metadatos.	2. Muestra los metadatos correspondientes al elemento seleccionado.
3. Modifica el valor del metadato que estime conveniente.	4. Valida si el campo esta vacío.
	5. Actualiza los cambios realizados.
	6. Termina el caso de uso.
Flujo alterno	
4.a Existen campos vacíos.	
Acción del actor	**Respuesta del sistema**

	4.a.1 Muestra un mensaje notificando al usuario que que existen campos vacíos.

Tabla 2.3: Descripción textual del CU. Gestionar metadatos de un contenido

Caso de uso	**Gestionar aspectos de un contenido**
Actor	Usuario
Resumen	El caso de uso inicia cuando el usuario desea añadir o eliminar un conjunto de metadatos a un documento o carpeta seleccionada en el sistema y finaliza cuando se añaden o se sustraen respectivamente estas propiedades del elemento especificado.
Complejidad	Media
Prioridad	Baja
Referencias	R3, R4, R5
Precondiciones	El usuario ha sido autenticado en el sistema.
	El usuario tiene los permisos necesarios para realizar la acción.
Poscondiciones	
Flujo de eventos	
Sección:“adicionar aspecto a un contenido”	
Acción del actor	**Respuesta del sistema**
1. Selecciona el documento o carpeta al cual le desea añadir nuevos metadatos.	2. Muestra una lista de acciones básicas. ○ Adicionar Aspecto. ○ Eliminar Aspectos. ○ Copiar. ○ Cortar. ○ Eliminar.
3. Selecciona la acción “Adicionar aspecto”.	4. Se muestra una lista con los aspectos definidos en el repositorio.

5. Selecciona el aspecto a adicionar.	6. Muestra las propiedades correspondientes al aspecto seleccionado.
7. Introduce los valores de las propiedades mostradas.	8. Añade las nuevas propiedades contenidas dentro del aspecto seleccionado al documento o carpeta especificada.
Sección:"eliminar aspectos de un contenido"	
Acción del actor	**Respuesta del sistema**
1. Selecciona el documento o carpeta al cual le eliminará uno o varios aspectos.	2. Muestra una lista de acciones básicas. ○ Adicionar Aspecto. ○ Eliminar Aspectos. ○ Copiar. ○ Cortar. ○ Eliminar.
3. Selecciona la acción "Eliminar aspectos".	4. Muestra una lista con los aspectos asociados al elemento seleccionado.
5. Selecciona el o los aspectos a sustraer del documento o carpeta previamente escogida.	6. Sustrae las propiedades contenidas dentro de los aspectos seleccionados del documento o carpeta especificada.
	7. Termina el caso de uso.

Tabla 2.4: Descripción textual del CU. Gestionar aspectos de un contenido

2.5.1. Plan de iteración de casos de uso

Iteración.	Descripción	CU a implementar	Duración total
1	En esta iteración se va a implementar el caso de uso que tiene alta prioridad para el cliente.	CU. Gestionar metadatos de un contenido.	01/10/2011 - 20/12/2011 (2 meses con 20 días)
2	En esta iteración se va a implementar el caso de uso que tiene baja prioridad para el cliente.	CU. Gestionar aspectos de un contenido.	20/01/2012 - 20/04/2012 (3 meses)

En el contenido de este capítulo se especifican los requisitos funcionales de la solución con su complejidad y prioridad para el cliente, además de varios artefactos tales como: modelo de dominio, para la concepción y conceptualización del entorno donde se desarrolla el problema y el diagrama de casos de uso del sistema. Se describen detalladamente los casos de uso a partir de las transacciones que ocurran durante la interacción usuario - sistema y los prototipos de interfaz de usuario, los que proporcionan una visión inicial de cómo pudiera quedar el sistema en términos de interfaz. Este resultado servirá como punto de partida para que la ejecución de los próximos flujos de trabajo se realice correctamente y lograr así su total entendimiento.

Capítulo 3

Diseño del sistema

"Puedes usar un borrador en la tabla de diseño o un martillo en el sitio de construcción". Frank Lloyd

El presente capítulo abarca el diseño que se propone como parte de la solución del sistema, donde se representan a partir de la descripción detallada de los casos de uso los diagramas de clases del diseño utilizando estereotipos web. Se explica además la arquitectura del sistema eXcriba y la de sus módulos, así como la descripción de los componentes que la conforman y los patrones de diseño utilizados.

El diseño del software es la última acción de la ingeniería correspondiente dentro de la actividad de modelado, la cual establece una plataforma para la construcción (generación de código y pruebas). Permite al ingeniero de software modelar el sistema o producto que se va a construir, además de ser la única forma en que, de manera exacta, un requisito del cliente se puede convertir en un sistema o producto de software terminado. Este flujo de trabajo es de gran importancia, ya que sin su realización se corre el riesgo de construir un sistema inestable, prospenso a fallos a la hora de realizar cambios, por muy pequeños que estos sean, además de ser la etapa donde se establece y fomenta la calidad del software[37].

3.1. Arquitectura del sistema

La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones de los componentes con cada uno de los otros y con el entorno y los principios que orientan su diseño y evolución[38].

Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que restringe los roles o rangos de los elementos arquitéctonicos y las relaciones permitidas entre esos elementos dentro de la arquitectura que se conforma a ese estilo[29].

La arquitectura basada en capas es un miembro de la familia de estilos de llamada y retorno, esta arquitectura se define como una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior[39]. En una

arquitectura multicapas lo importante y que siempre se debe recordar es que las capas inferiores brindan servicios a las capas superiores (independientemente del nivel en que se encuentren). La clave de su desarrollo es que una capa solamente debe utilizar lo que la interfaz de la o las capas inferiores le brindan, de este modo se puden intercambiar las capas respetando la interfaz, que viene a ser como un “contrato entre capas”.

Durante el diseño de la presente investigación se utilizó una arquitectura basada en capas o ***n-capas***, como también se le denomina por ser la arquitectura definida para el proyecto GDA eXcriba, ademas dicha arquitectura brinda ventajas como el poder aislar en componentes independientes aspectos del desarrollo, tales como las cuestiones de presentación, lógica de negocio y mecanismos de almacenamiento, que se pueden después reutilizar en otros sistemas.

3.1.1. Análisis general de la Arquitectura del GDA eXcriba

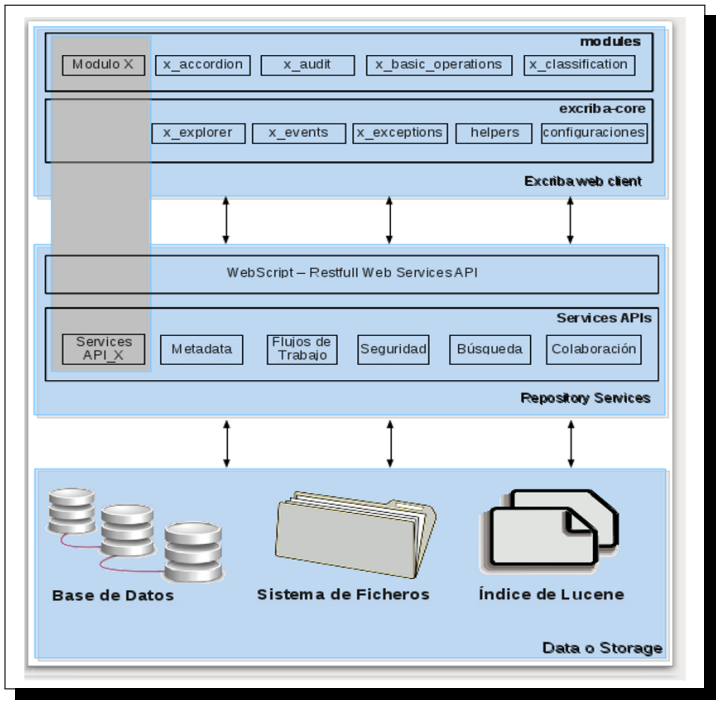


Figura 3.1: Vista de la arquitectura de eXcriba

Como se puede apreciar en la Figura 3.1 el GDA eXcriba se compone de tres capas arquitectónicas fundamentales (Cliente Web, Servicios de Repositorio, Almacenamiento de Datos) que interactúan entre sí durante la realización de cada una de las funcionalidades que brinda el sistema.

La capa superior, “Cliente Web” se compone de un conjunto de módulos que dan soporte a las diferentes operaciones o funcionalidades con las que interactúan los diferentes usuarios del sistema. Su principal componente es el módulo “excriba-core” el cual brinda un conjunto de servicios, interfaces, configuraciones y funcionalidades genéricas de las cuales dependen (o pueden ser usadas) por el resto de los módulos que conforman la aplicación web.

Las otras dos capas, forman parte del núcleo del sistema: el Sistema de Gestión de Contenido Empresarial (ECM) Alfresco. La capa de servicios de acceso al repositorio “Servicios de Repositorio”, a través del Framework WebScript provee un API completa de objetos de repositorio y servicios basados en el estilo arquitectónico REST. Estos servicios juegan un papel fundamental en la incorporación de las funcionalidades al sistema al tiempo que garantizan que desde el cliente web se pueda acceder a los datos que se almacenan en el repositorio de contenido.

La capa de almacenamiento de datos, la más baja de la arquitectura, constituye precisamente los cimientos sobre los cuales se construye o desarrolla el resto de la aplicación, esta capa se compone de la base de datos (en la cual se almacena de manera estructurada toda la información referente a los archivos binarios «documentos» en forma de metadatos, a los usuarios y demás objetos de negocio) y un sistema de ficheros que se compone de documentos que persisten en formato binario.

3.1.1.1. ¿Cómo interactúan los componentes?

Cómo se pueden observar en la Figura 3.1 cuando se implementa un nuevo módulo, Módulo X por ejemplo, este pasa a formar parte del subsistema “modules” dentro de la primera capa, “excriba web client”. Por lo general, estos módulos hacen uso de algunas de las funcionalidades o configuraciones que provee el núcleo (excriba-core). Para acceder a los datos del repositorio se hace uso de los servicios que se encuentran disponibles en el subsistema “services” dentro de la capa “Repository Services”, servicios que en la mayoría de los casos hacen uso de las APIs que provee Alfresco para el desarrollo de determinadas funcionalidades.

3.1.2. Arquitectura del cliente web de eXcriba. Diseño de la arquitectura del módulo

El módulo para la gestión de metadatos en el eXcriba responde a una arquitectura basada en dos capas fundamentales (capa de presentación y capa de aplicación) y una capa secundaria (capa de acceso al repositorio) con la que se interactúa para obtener los datos que serán procesados por dicho módulo.

- Capa de Presentación: Se compone de los recursos básicos que serán procesados, por lo general, por el cliente (navegador web) para generar la salida que será expuesta al usuario en forma de interfaces sencillas. En este conjunto se encuentran las páginas HTML, las hojas de estilos y los ficheros javascript que se ejecutan del lado del cliente. En esta capa el módulo para la Gestión de Metadatos hace uso de los framework jQuery y de la API de eXcriba para el procesamiento de las interfaces de usuario a través del lenguaje JavaScript.

- Capa de Aplicación: Agrupa la lógica del negocio que se ejecuta del lado del servidor (Apache en este caso). En función del módulo que se trate, en esta capa se validan los datos que provienen del cliente, ejecuta ciertas operaciones con dichos datos y delega el envío de los mismos al repositorio a través de las capas inferiores, así mismo, en determinados casos prepara los datos obtenidos de las capas inferiores para conformar las vistas a mostrar. Para el desarrollo del módulo se hace uso en esta capa de el subsistema de Control de Acceso, el cual es utilizado para validar la sesión de un usuario previamente autenticado, el subsistema de Gestión de Acciones y el de Gestión de Eventos interactúan entre sí para llevar a cabo las acciones y eventos que se desencadenan en el sistema una vez que el usuario ejecuta determinada funcionalidad y el subsistema de Comunicación con la Capa de Acceso a Datos posibilita hacer la llamada a los servicios.

- Capa de Acceso al Repositorio: En esta capa se encuentran disponibles los servicios que serán empleados por el módulo para acceder o modificar los datos que persisten en el repositorio de contenido a través de las diferentes APIs que provee Alfresco (Alfresco WebScript Framework). Además, en esta capa el módulo hace uso del subsistema Servicios del Módulo Gestión de Metadatos, en el cual se encuentran los servicios que fueron implementados para la realización del mismo, dichos servicios se explicarán en detalle en apartados siguientes. En el subsistema Servicio de Nodos se encuentran los servicios que permiten manipular los nodos, se utilizó de este subsistema

el servicio para modificar las propiedades asociadas a un nodo y el servicio para adicionar aspectos a un nodo, finalmente el subsistema Servicio de Control de Acceso se usa para verificar que el usuario pueda ejecutar las diferentes acciones que se disponen en el nuevo módulo.

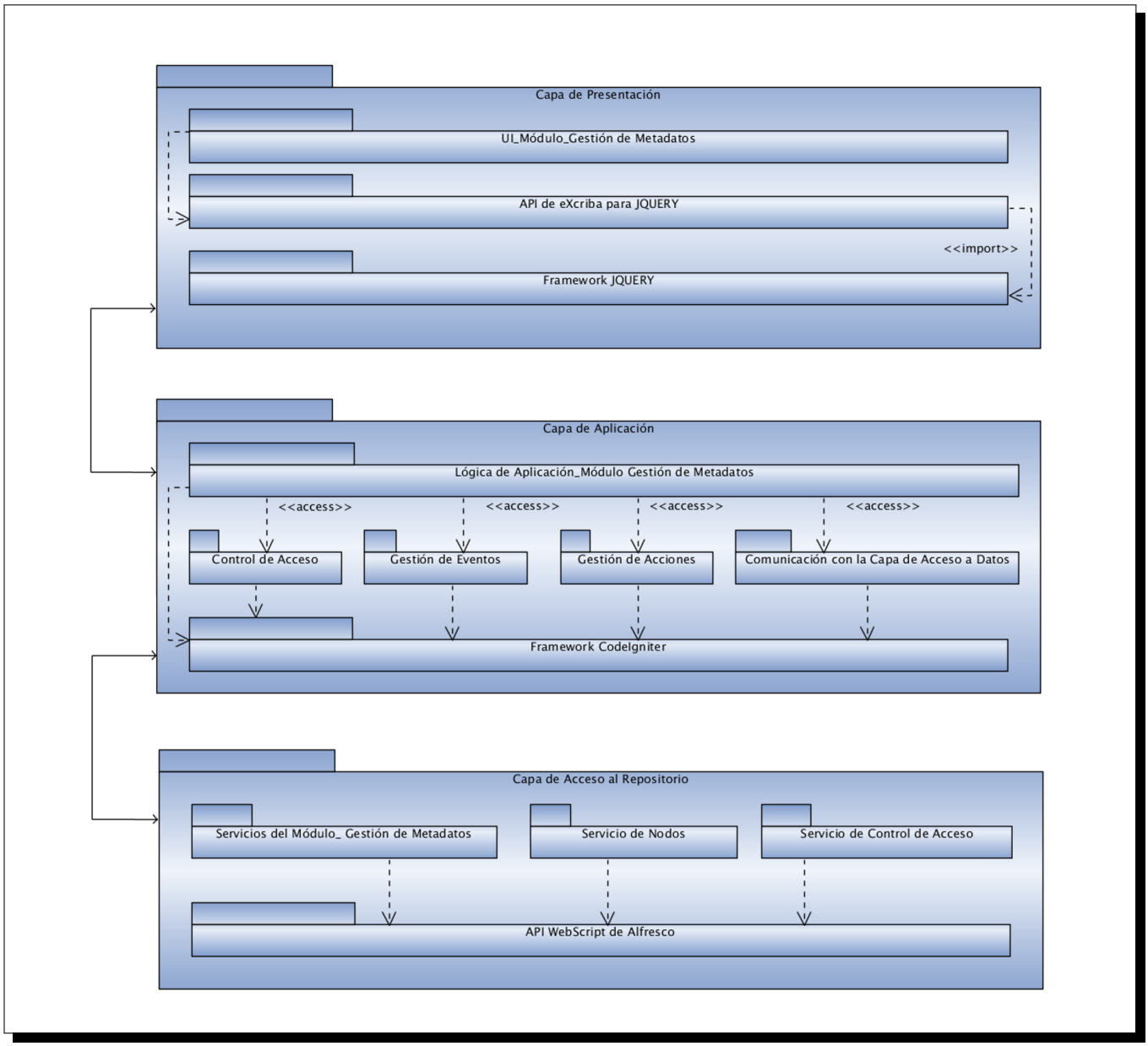


Figura 3.2: Vista en subsistemas de la arquitectura del módulo

3.2. Patrones de Diseño

Un patrón es una pareja de problema/solución con un nombre y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas[40]. En el diseño de esta investigación se utilizaron los patrones GRASP, por lo que se hará referencia específicamente a estos en este apartado.

GRASP es un acrónimo que significa General Responsibility Asignment Software Patterns (Patrones Generales de Software para Asignar Responsabilidades), donde una responsabilidad no es más que un método en los artefactos UML, por lo que los patrones GRASP describen los principios fundamentales de la asignacion de responsabilidades a objetos, expresados en forrna de patrones. Esta familia está compuesta por nueve patrones: experto, creador, alta cohesión, bajo acomplamiento, controlador, polimorfismo, fabricación pura, indirección y no hables con extraños[40]. A continuación se explica brevemente los patrones GRASP utilizados en el diseño del módulo.

3.2.1. Experto

El patrón Experto es el que se encarga de asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. Se debe tener en cuenta que la responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos), es decir, una clase contiene toda la información necesaria para realizar la labor que tiene encomendada[40].

Ejemplo: La clase Wscript_node es el experto en la manipulación de los nodos, la cual se encarga, a partir de un conjunto de entradas conformar la url para realizar la llamada al servicio y demás datos necesarios para llevar a cabo cada una de las operaciones que hagan manejo de nodos.

3.2.2. Bajo Acoplamiento

Este patrón es el que se encarga de asignar una responsabilidad para mantener bajo acoplamiento. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con las que conoce y recurre a ellas. Una clase con bajo o débil acoplamiento no depende de muchas otras y una con alto o fuerte recurre a muchas otras, esta último tipo de clase no es muy conveniente, ya que presentan los siguientes problemas[40]:

1. Los cambios de las clases afines ocasionan cambios locales.

2. Son más difíciles de entender cuando están aisladas.

3. Son más difíciles de reutilizar porque se requiere la presencia de otras clases de las que dependen.

El bajo acoplamiento es muy importante si se busca la reutilización, se evidencia este patrón en la separación de las diferentes capas y la comunicación entre estas a partir de interfaces genéricas.

Ejemplo: La clase controladora x_basic hace uso de una instancia de NodeService (interfaz que contiene los métodos necesarios para la manipulación de nodos) que en estos momento solo tiene una implementación concreta, Wscript_node, esta última se comunica con el repositorio de contenidos a través de servicios REST. Sin embargo, si en determinado momento, se decide cambiar a otra tecnología no sería necesario modificar el código fuente asociado a las capas superiores, en tal caso solo se reimplementaría la capa de acceso al repositorio.

3.2.3. Controlador

Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema y define para ello el método de su operación. La mayor parte de los sistemas reciben eventos de entrada externa, los cuales generalmente incluyen una interfaz gráfica para el usuario operado por una persona, para manejar estos eventos de entrada se eligen los controladores[40].

Ejemplo: En este caso la clase x_basic representa de alguna manera al sistema entero, siendo dicha clase el controlador que delega a otros el trabajo que ha de realizarse mientras coordina las actividades (funciones que se ejecutan en el módulo). Este controlador se encarga de:

- Validación de las entradas, método (setProperty), esto es verificar en caso de ser posible, que el valor especificado para modificar la propiedad seleccionada es correcto.

- Verificar que en el momento que el usuario ejecuta la operación aún tenga el permiso necesario para llevarla a cabo.

- Avisarle a los componentes necesarios que se está ejecutando dicha operación, por ejemplo que en el cliente web se actualice el valor de la propiedad.

3.3. Diseño del Caso de Uso gestionar metadatos de un contenido

Durante el diseño del caso de uso gestionar metadatos de un contenido se identificaron las clases del diseño necesarias para llevar a cabo las funcionalidades asociadas al mismo. En esta sección se definen cada una de estas clases y sus relaciones en términos de diagramas de clases del diseño y diagramas de interacción. Además, se describen cada una de estas clases y los servicios que deben implementarse o usarse para la completa realización del caso de uso.

3.3.1. Diagrama de clases del diseño

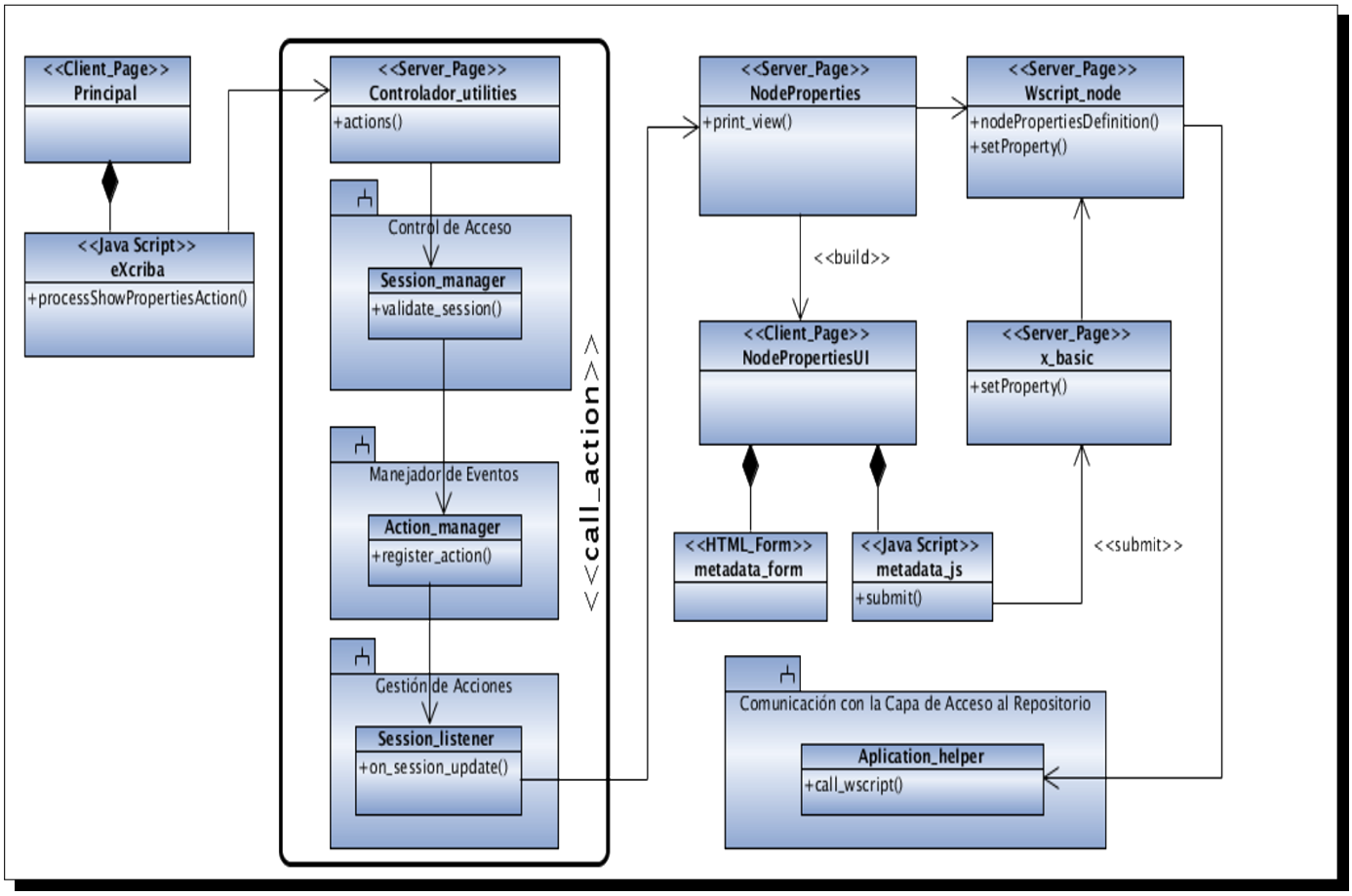


Figura 3.3: Diagrama de clases del diseño del CU Gestionar metadatos

3.3.2. Descripción de las clases del diseño

A continuación se exponen en forma de tablas las principales clases que intervienen en el caso de uso Gestionar metadatos de contenido, así como las funcionalidades identificadas como exclusivamente necesarias para la realización del mismo.

Nombre: NodeProperties	
Tipo de clase: Controladora	
Descripción: Coordina las operaciones que deben llevarse a cabo para generar el formulario donde el usuario podrá observar o modificar las propiedades asociadas a un documento o carpeta seleccionada por el usuario.	
Responsabilidades:	
Nombre	**Descripción**
print_view()	Genera la vista donde el usuario observará o editará los propiedades de un documento o carpeta del sistema.

Tabla 3.1: Descripción de la clase NodeProperties

Nombre: Wscript_node	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para hacer la llamada a los diferentes servicios disponibles.	
Responsabilidades:	
Nombre	**Descripción**
nodePropertiesDefinition()	Prepara los datos necesarios para hacer la llamada al servicio web nodePropertiesDefinition que devuelve las propiedades asociadas a un nodo especificado por parámetro.

setProperty()	Prepara los datos necesarios para hacer la llamada al servicio web setProperty el cual modifica el valor de un propiedad, para ello recibe como parámetros el identificador del documento o carpeta, el nombre de la propiedad y el nuevo valor que la misma contendrá.

Tabla 3.2: Descripción de la clase Wscript_node

Nombre: x_basic	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para interactuar con la capa de Acceso al Repositorio. Su función es actuar como la interfaz a través de la cual se comunica la capa de aplicación/negocio (específicamente las funcionalidades del módulo x_basic_operations) con la de Acceso al Repositorio.	
Responsabilidades:	
Nombre	**Descripción**
setProperty()	Coordina las operaciones necesarias que deben llevarse a cabo para modificar el valor de una propiedad, entre ellas interactuar con la Capa de Acceso al Repositorio.

Tabla 3.3: Descripción de la clase x_basic

3.3.3. Descripción de los servicios

Para la realización del caso de uso se hace necesario un servicio que permita obtener las propiedades que se asocian a un nodo, así como otro servicio que permita modificar el valor de una propiedad, para ello se propone la implementación del servicio **NodePropertiesDefinition**, descrito en la Tabla 3.4 y el uso del servicio **SetProperty** incorporado al subsistema de servicios de Nodos.

Servicio: NodePropertiesDefinition	
Descripción:	Devuelve las propiedades asociadas a un nodo.
URL:	http://<host>:<puerto>/alfresco/service/cu/uci/excriba /node/storeType/storeId/nodeUUID/definition
Requerimiento de Autenticación:	Usuario
Requerimiento de Transacción:	Requerida
Formato de Plantilla de Respuesta:	JSON
Paquete:	cu.uci.excriba.node
Método HTTP:	GET
Lenguaje de implementación del controlador:	Java
Clase:	cu.uci.excriba.wscripts.node.NodePropertiesDefinition
Parámetros o Argumentos de entrada	
Nombre y Descripción:	**Formato/Ejemplo**
storeType: identificador del tipo de almacén.	workspace
storeId: identificador del almacén (store).	SpacesStore
nodeUUID: identificador del nodo.	ab3eff9c-f868-44f0-b351-bdb657f6d57b
Parámetros o Argumentos de salida	
Nombre y Descripción:	**Formato/Ejemplo**

properties: listado de propiedades asociadas al nodo agrupadas por tipo de contenido.	[
	"tipo de contenido o aspecto" : {
	"nombre de la propiedad": {
	"name" : "nombre de la propiedad",
	"title" : "titulo de la propiedad",
	"description" : "descripción de la propiedad",
	"defaultValues" : "valores por defecto",
	"dataType" : "tipo de dato",
	"multiValued" : multivaluado (true/false),
	"mandatory" : obligatorio(true/false),
	"protected" : protegido(true/false),
	"value" : valor de la propiedad,
	"url" : "url de la propiedad"
	}
	}
	]

Tabla 3.4: Descripción del servicio NodePropertiesDefinition

El servicio **SetProperty** fue implementado por el equipo de desarrollo del GDA eXcriba con el objetivo de permitir modificar el valor a un metadato, recibe como entrada el identificador del nodo, el nombre de la propiedad que se desea modificar y el nuevo valor que se le asignará a dicha propiedad.

3.3.4. Diagramas de interacción del diseño

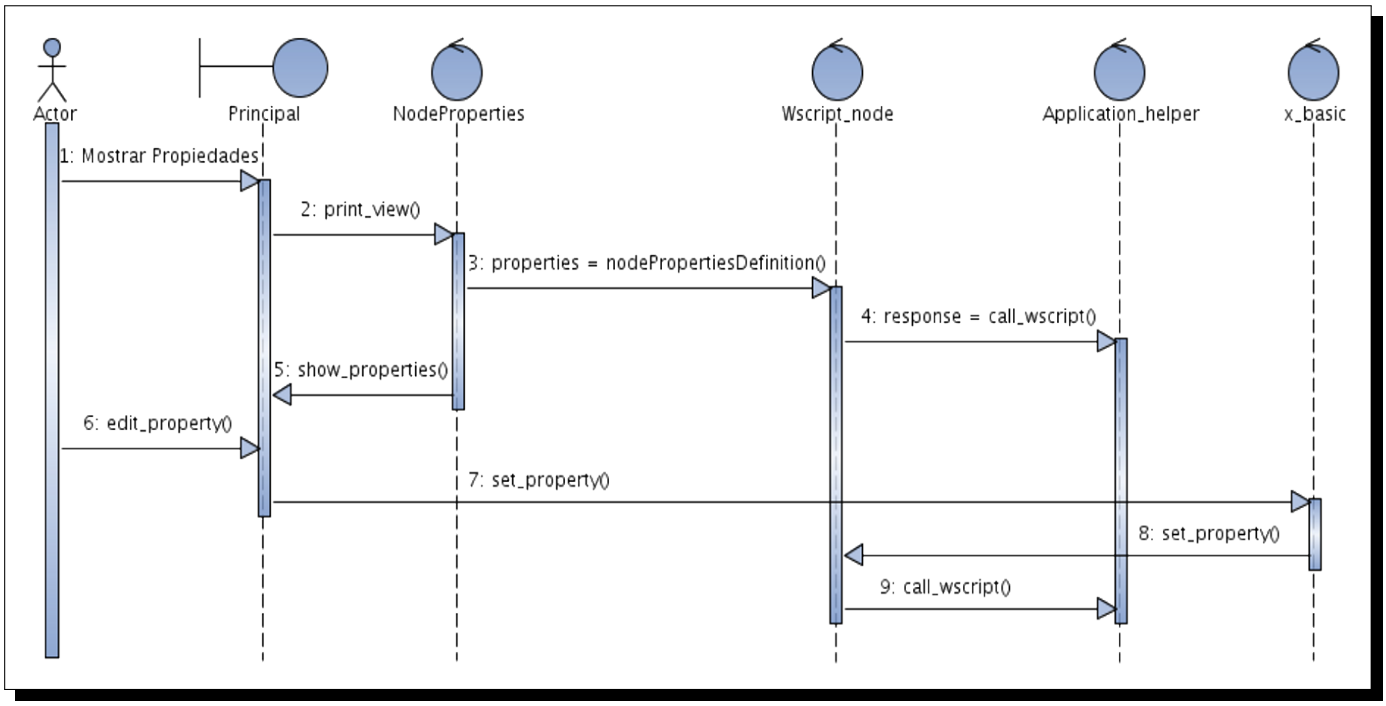


Figura 3.4: Diagrama de interacción del diseño del CU Gestionar metadatos

Para ver los restantes artefactos que se generaron durante el diseño del caso de uso gestionar aspectos de un contenido consultar los **Anexos B y C**.

Durante el transcurso de este capítulo se definió la arquitectura del módulo, se desarrollaron los diagramas de clases del diseño que permitirán dar respuesta a las funcionalidades descritas en el capítulo 2 en términos de requisitos funcionales, se describen además cada una de estas clases y los servicios que deben implementarse o usarse para desarrollar exitosamente los casos de uso que se definen, así como los diagramas de interacción de dichos casos de uso. Todo lo antes expuesto permitirá dar inicio a la implementación del módulo con el objetivo de que el resultado final de la construcción posea una alta calidad.

Capítulo 4

Implementación y prueba

"Todo programa hace algo correctamente, que puede no ser lo que necesitamos que haga". Anónimo

Este capítulo describe cómo los elementos del modelo de diseño son implementados en términos de componentes y cómo los mismos se organizan de acuerdo con los nodos referidos en el modelo de despliegue. Se exponen además las distintas pruebas realizadas a los casos de usos, siguiendo particularmente la técnica de prueba de caja negra y el método de partición de equivalencia.

4.1. Implementación

La implementación parte con el resultado del diseño y se implementa el sistema en términos de componentes, es decir ficheros de código, ejecutables, entre otros. Durante esta fase todos los componentes, características y requisitos deben ser implementados, integrados y probados en su totalidad, obteniendo una versión aceptable del producto. El propósito principal de la implementación es desarrollar la arquitectura y el sistema como un todo, es decir[41]:

- La identificación de componentes significativos arquitectónicamente, tales como componentes ejecutables.
- La asignación de componentes a los nodos en las configuraciones de redes.

4.1.1. Diagrama de despliegue

El diagrama de despliegue se utiliza para modelar los aspectos físicos sobre los que se ejecutarán los componentes del sistema de *software* en términos de nodos, donde los nodos sirven para modelar la topología del *hardware* sobre el que se ejecuta un sistema y representan dispositivos sobre los cuales se pueden desplegar los componentes[42].

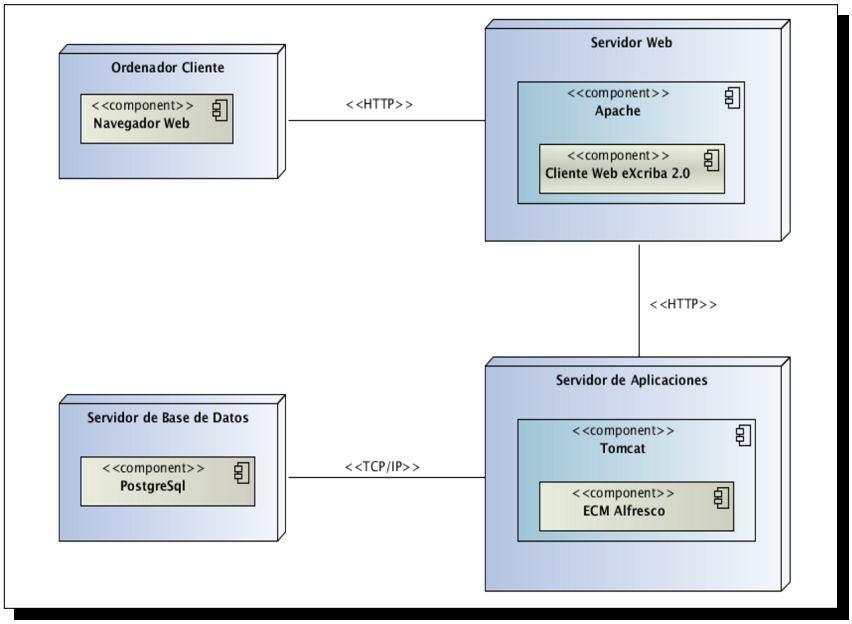


Figura 4.1: Diagrama de despliegue

4.1.1.1. Descripción del diagrama de despliegue

1. Nodo **Ordenador Cliente:** este nodo hará referencia a los ordenadores desde los cuales los usuarios acceden al cliente web de la aplicación a través de un navegador web (Opera, Mozilla Firefox, Internet Explorer, entre otros).

2. Nodo **Servidor Web:** es donde se despliega el cliente web en un servidor Apache 2.0.

3. Nodo **Servidor de Aplicaciones:** aquí se despliega el ECM Alfresco, el cual es el repositorio de contenidos y servicios del GDA eXcriba.

4. Nodo **Servidor de Base de Datos:** es donde se encuentra la Base de Datos (Postgree SQL)[1] que utiliza el ECM Alfresco para almacenar la información estructurada (metadatos asociados a los documentos, datos de los usuarios y de los procesos de negocio, entre otros).

[1] PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente[43].

Según lo mostrado en el diagrama de despliegue y lo explicado anteriormente el sistema eXcriba en conjunto con todos sus módulos se ejecuta sobre un nodo cliente y tres nodos servidores. En primer lugar, el usuario interactúa con el cliente web del GDA eXcriba a través de la comunicación que se establece mediante el protocolo HTTP entre su ordenador y el Servidor Web, luego este último establece una conexión con el Servidor de Aplicaciones mediante el protocolo HTTP para obtener los datos solicitados anteriormente por el usuario en su interacción con la aplicación, para ello el Servidor de Aplicaciones mediante el protocolo TCP/IP accede a la información almacenada en la Base de Datos dando respuesta a la petición realizada por este servidor.

4.1.2. Diagrama de componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño, algunos estereotipos de componentes son los siguientes[44]:

- *«executable»*: es un programa que puede ser ejecutado en un nodo.
- *«file»*: es un fichero que contiene código fuente o datos.
- *«library»*: librería estática o dinámica.
- *«table»*: es una tabla de base de datos.
- *«document»*: es un documento.

El diagrama de componentes muestra la vista física del *software* en términos de componentes ejecutables y librerías de clases con sus relaciones y sus dependencias[42]. Se muestra a continuación el diagrama de componentes del módulo para la gestión de metadatos en el GDA eXcriba.

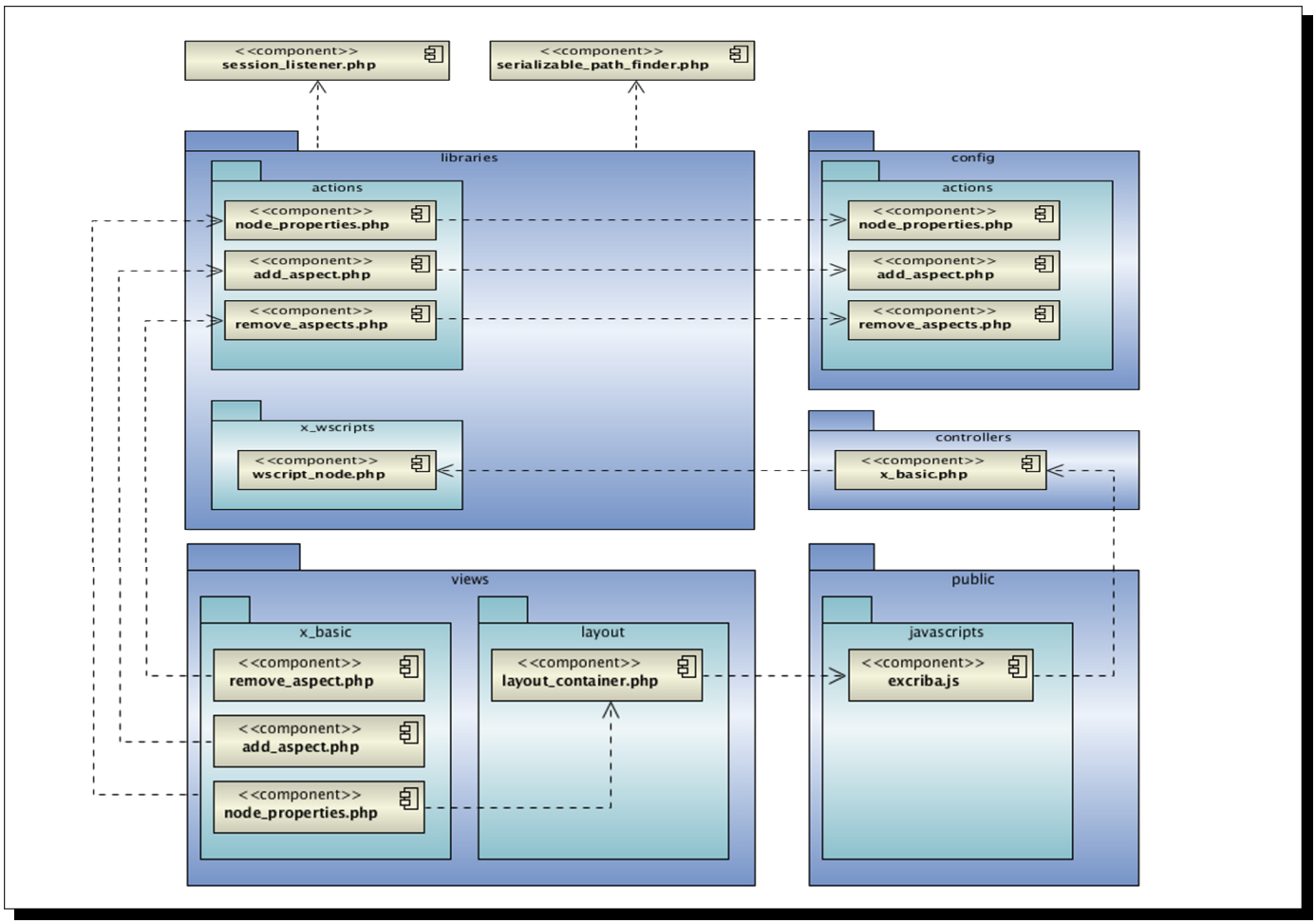


Figura 4.2: Diagrama de componentes

4.2. Prueba

Asegurar la calidad es un factor clave en el proceso de desarrollo de *software* y el instrumento adecuado para garantizar dicha calidad son las pruebas de software. La prueba de software es una actividad dirigida a la evaluación de un atributo o capacidad de un programa o sistema y la determinación de que cumple con sus resultados requeridos[45]. Se presenta a continuación la validación de la solución propuesta.

4.2.1. Criterios de validación de requisitos

La validación del software se logra mediante una serie de pruebas que demuestran que se cumple con los requisitos[37]. Para validar los requisitos obtenidos en el módulo para la gestión de metadatos en el eXcriba se utilizan criterios propuestos por el proceso de mejora basado en el segundo nivel de CMMI que se lleva a cabo en la UCI para la metodología RUP, dichos criterios contienen varias interrogantes que permiten validar si el requisito puede o no aprobarse, estas interrogantes se exponen a continuación. Para ver en detalladamente estos artefactos, consultar el expediente del proyecto eXcriba[2].

- ¿El proveedor del requisito es un proveedor válido?
- ¿El requisito está identificado como único?
- ¿El requisito es modificable?
- ¿El requisito no es ambiguo?
- ¿El requisito está completo?
- ¿El requisito es congruente con otros requisito relacionados?
- ¿El requisito puede ser implementado?
- ¿El requisito puede ser probado?
- ¿El resultado de la evaluación de impacto es positivo?
- ¿El requisito está correcto?
- ¿El requisito es traceable?

Al aplicarse estos criterios sobre los requisitos del módulo se obtuvo como resultado el 100 % de requisitos aprobados.

[2]Puede acceder al expediente a través de la dirección: http://pruebascenia.uci.cu/angola/gestdoc_angola/excriba_web/

4.2.2. Técnica de prueba de caja negra

La prueba es una actividad realizada para evaluar la calidad del producto y para mejorarla, mediante la identificación de defectos y problemas. Las pruebas de *software* consisten en la verificación dinámica del comportamiento de un programa en un conjunto finito de casos de prueba[46], estas no tienen el objetivo de prevenir errores sino de detectarlos. Se efectúan sobre el trabajo realizado y se deben encarar con la intención de descubrir la mayor cantidad de errores posibles. La técnica de caja negra es la utilizada para validar la propuesta de solución.

Las pruebas de caja negra, también denominadas pruebas de comportamiento se centran en los requisitos funcionales del *software*[47] y se llevan a cabo sobre la interfaz del mismo, obviando el comportamiento interno y la estructura del programa. El objetivo de esta prueba es verificar que el sistema cumpla con los requisitos funcionales establecidos por el cliente. En la prueba de caja negra los casos de prueba pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepta de forma adecuada, que se produce una salida correcta y que la integridad de la información externa se mantiene.

Las pruebas de caja negra tratan de encontrar errores de las siguientes categorías[47]:

- Funciones incorrectas o faltantes.
- Errores de interfaz.
- Errores en estructuras de datos a bases de datos externas.
- Errores de comportamiento o desempeño.
- Errores de inicialización.

Con el objetivo de demostrar que el módulo cumple con los requisitos que han sido definidos previamente, se utiliza la técnica de prueba de caja negra, la cual ayuda a mejorar la calidad de dicho módulo al descubrir sobre su interfaz la mayor cantidad posible de errores y en base a dichos errores proceder a la corrección de los mismos. Uno de los métodos que se utiliza para la confección de los casos de pruebas de caja negra es el de partición de equivalencia, método usado en el marco de esta trabajo de diploma para la elaboración de los respectivos casos de pruebas que se emplean para validar la interfaz del módulo.

4.2.2.1. Método de partición equivalente

Los métodos de pruebas no son más que formas de realizar pruebas a un sistema informático para comprobar el cumplimiento de los requerimientos y de las funcionalidades.

Entre los métodos que basan su filosofía en la de caja negra se encuentra el de partición equivalente, técnica que será empleada en el desarrollo de los casos de prueba que contendrá el módulo, pues la misma permite examinar los valores válidos e inválidos de las entradas existentes en el *software*.

La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Por lo general, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica. Las clases de equivalencia se pueden definir de acuerdo con las siguientes directrices[37]:

1. Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos no válidas.

2. Si una condición de entrada requiere un valor específico, se define una clase de equivalencia válida y dos no válidas.

3. Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una no válida.

4. Si una condición de entrada es lógica, se define una clase de equivalencia válida y una no válida.

Luego de aplicar estas directrices se procede a ejecutar los de casos de prueba que se identifiquen.

Para la realización de las pruebas del módulo se generaron los artefactos “Diseño de casos de pruebas basado en casos de uso”. Por cada caso de uso del sistema se generó un documento en donde se recogen todos los datos necesarios para probar la interfaz. A continuación se muestra como ejemplo el caso de prueba para el caso de uso gestionar metadatos de un contenido, especificando para ello las condiciones

que deben cumplirse para su ejecución, la respuesta emitida por el sistema para cada escenario expuesto y los pasos fundamentales a seguir para ejecutar la acción.

Caso de prueba 1. CU_Gestionar metadatos de un contenido.

Descripción de la funcionalidad: el caso de uso inicia cuando el usuario procede a obtener o modificar los metadatos que le son asignados a un documento o carpeta específico del sistema y finaliza cuando se ejecuta la operación deseada.

Condiciones de ejecución:

- El usuario ha sido autenticado en el sistema.
- El usario tiene los permisos requeridos para realizar la acción.

Sección: mostrar propiedades de un contenido

Escenario	Respuesta del sistema	Flujo central
EC 1.1 Mostrar propiedades asociadas a un contenido correctamente.	Se muestran las propiedades asociadas al elemento especificado según su tipología documental.	• Autenticarse en el sistema. • Seleccionar un documento o carpeta en el sistema. • Seleccionar la opción “Propiedades” en la barra lateral derecha.
EC 1.2 Selección de más de un elemento.	Se notifica al usuario que tiene más de un elemento seleccionado con el mensaje “Hay más de un elemento seleccionado en la lista”.	

Tabla 4.1: CP1-CU Gestionar metadatos de un contenido.

Sección mostrar

Sección: modificar propiedades de un contenido

Escenario	Respuesta del sistema	Flujo central
EC 1.1 Modificar propiedades a un contenido correctamente.	Se guardan los cambios actualizando el valor del metadato modificado.	• Autenticarse en el sistema. • Seleccionar un documento o carpeta en el sistema. • Seleccionar la opción "Propiedades" en la barra lateral derecha. • Modificar el valor asociado a una propiedad/metadato.
EC 1.2 Modificar el valor de una propiedad introduciendo un campo vacío.	No se guardan los cambios realizados y se muestra el mensaje de error "Verifique los campos obligatorios".	

Tabla 4.2: CP1-CU Gestionar metadatos de un contenido.

Sección modificar

Caso de Prueba 2: CU_Gestionar aspectos de un contenido.

Descripción de la funcionalidad: el caso de uso inicia cuando el usuario desee añadir o eliminar un conjunto de nuevos metadatos a un documento o carpeta seleccionada en el sistema y finaliza cuando se añaden o se sustraen respectivamente estas propiedades del elemento especificado.

Condiciones de ejecución:

- El usuario ha sido autenticado en el sistema.
- El usario tiene los permisos requeridos para realizar la acción.

Sección: adicionar aspecto a un contenido

Escenario	Respuesta del sistema	Flujo central
EC 1.1 Adicionar aspecto a un contenido correctamente.	Se adicionan al elemento los metadatos contenidos dentro del aspecto especificado.	• Autenticarse en el sistema. • Seleccionar un documento o carpeta en el sistema. • Seleccionar la opción "Acciones" en la barra lateral derecha. • Seleccionar el icono de "Adicionar Aspecto". • Seleccionar el aspecto a adicionar. • Introducir los valores de las propiedades asociadas al aspecto seleccionado. • Presionar el botón Aceptar
EC 1.2 Los valores de las propiedades asociadas al aspecto contienen campos vacíos.	No se adicionan al elemento seleccionado los metadatos contenidos dentro del aspecto especificado y se muestra el mensaje de error "Verifique los campos obligatorios".	
EC 1.3 No existen aspectos para adicionarle a un contenido.	Se notifica al usuario que no existen aspectos en el repositorio con el mensaje "No hay aspectos disponibles".	
EC 1.4 Cancelar operación.	No se adiciona el aspecto y se muestra la interfaz principal del sistema	

Tabla 4.3: CP2-CU Gestionar aspectos de un contenido.

Sección adicionar

Sección: eliminar aspectos de un contenido

Escenario	Respuesta del sistema	Flujo central
EC 1.1 Eliminar aspectos de un contenido correctamente.	Se eliminan del elemento las propiedades contendidas dentro de los aspectos seleccionados.	• Autenticarse en el sistema. • Seleccionar un documento o carpeta en el sistema. • Seleccionar la opción "Acciones" en la barra lateral derecha. • Seleccionar el icono de "Eliminar Aspectos". • Seleccionar el o los aspecto a eliminar. • Presionar el botón Aceptar
EC 1.2 No hay aspectos asociados al elemento especificado.	Se notifica al usuario que el elemento seleccionado no tiene ningún aspecto asociado con el mensaje "No hay aspectos para eliminar".	
EC 1.3 Cancelar operación.	No se elimina el o los aspectos seleccionados y se muestra la interfaz principal del sistema.	

Tabla 4.4: CP2-CU Gestionar aspectos de un contenido.

Sección eliminar

Para ver la versión extendida de estos casos de prueba consultar el **Expediente del Proyecto eXcriba**, donde se encuentran todos los artefactos en su totalidad.

4.2.3. Resultado de las pruebas realizadas

La realización de estas pruebas posibilitaron detectar algunas no conformidades, las cuales fueron erradicadas a medida que se concluía cada una de las iteraciones que se definieron. Dichas pruebas, permitieron corroborar que el módulo cumple con las especificaciones que se trazaron y que se le dio cumplimiento a cada uno de los requisitos definidos. A continuación se muestra la relación de no conformidades (detectadas y resueltas) por iteración:

Iteración	NC Detectadas	Asociadas a	NC Resueltas
I	12	erores de interfaz y validación	7
II	9	errores de interfaz y errores ortográficos	7
III	4	funciones incorrectas o ausentes y errores de validación	4

Tabla 4.5: Resultado de las pruebas

Como se puede observar en la tabla anterior se realizaron tres iteraciones de pruebas. Con dos casos de uso a probar se encontraron en la primera iteración, 12 no conformidades asociadas a errores de interfaz y validación, pudiendo resolverse solo 7, en la segunda iteración fueron detectadas 4 nuevas no conformidades asociadas a errores de interfaz y ortografía, además de las 5 pendientes de la iteración anterior, en esta oportunidad se le dio respuesta a 7 no conformidades permaneciendo irresueltas 2 no conformidades, dándole solución en una tercera iteración junto con 2 nuevas conformidades encontradas, asociadas a errores de validación y funciones incorrectas o ausentes, quedando el sistema listo para su explotación.

Se concluye este capítulo con las pruebas realizadas anteriormente, donde se plasmaron los resultados obtenidos durante el desarrollo de los distintos casos de prueba que se ejecutaron, se define y describe también en este acápite el diagrama de despligue, garantizando mediante la comunicación entre los nodos que lo componen el correcto funcionamiento del sistema en general, además se muestra el diagrama de componentes, el cual genera un vista con las librerías y ficheros que integran el módulo.

Conclusiones

Como resultado de la investigación realizada se desarrolló un módulo para la gestión de metadatos desde el GDA eXcriba, concluyendo:

- La valoración de los mecanismos empleados por diferentes sistemas actuales para la gestión de metadatos posibilitó la identificación de componentes esenciales para su gestión en el GDA eXcriba.
- El uso de técnicas para realizar la captura de requisitos permitió identificar las funcionalidades que debe tener el sistema, logrando definir una especificación completa, correcta y consistente de los requisitos a implementar.
- El desarrollo del módulo facilita la incorporación, asignación y uso de los metadatos asociados a las tipologías documentales en el GDA eXcriba garantizando que permanezcan accesibles y disponibles a lo largo del tiempo.
- Las pruebas aplicadas a la propuesta de solución permitieron validar las funcionalidades implementadas para la gestión de metadatos asociados a las tipologías documentales en el GDA eXcriba.

Recomendaciones

Se recomienda:

1. Realizar pruebas de caja blanca a cada una de las funcionalidades implementadas en el módulo.
2. Hacer un estudio de la gestión de metadatos asociados a las los documentos en formato duro (papel).
3. Desarrollar una herramienta para el diseño de formularios que permitan la visualización y gestión de los metadatos.
4. Incorporar un herramienta para la extracción de metadatos.

Referencias bibliográficas

[1] Dr. Julio Cerdá Díaz. Gestión electrónica de documentos y acceso a la información. *Universidad de Alcalá*, pages 1–39, 2000.

[2] MSc. Mayra Mena Mugica. Utilidad de las soluciones archivísticas para la gestión de la información en los sistemas electrónicos del sector de la salud. *Revista ACIMED (Revista cubana de los profesionales de la información y de la comunicación en Salud)*, 15(3):1–15, 2007. ISSN 1409-4746.

[3] Soluciones Open Source. *Libro Blaco: Gestión Documental Open Source*. Smile open source solutions, 2008. 92 pp.

[4] Proyecto UNE-ISO. Norma iso 15489. Información y documentación. Gestión de documentos. 1:1–30, 2000.

[5] Teresa Allepuz Ros and Carmen Gutiérrez La Rubia. Los sistemas de gestión de la documentación en las organizaciones. *Gabinete de Asesores Documentalistas*, pages 1–7, 1995.

[6] Anne J Gilliland, Tony Gill, Mary Woodley, and Maureen Whalen. *Introduction to Metadata*. Getty Research Institute, second edition, 2008. ISBN 978-0-89236-896-9. 81 pp. [**Consultado**: Octubre 2011].

[7] Carlota Bustelo Ruesta. Los grandes temas relacionados con la gestión de documentos: desafíos y oportunidades. *El profesional de la información*, 20(2):1–6, 2011. [**Consultado**: Octubre 2011].

[8] Eva Méndez Rodríguez. La descripción de documentos electrónicos a través de metadatos: una visión para la archivística desde la nueva e-administración. *Revista d'Arxius*, pages 1–28, 2003.

[9] NISO (National Information Standards Organization). *Understanding Metadata*. NISO Press, 2004. ISBN 1-880124-62-9. 20 pp.

[10] Proyecto UNE-ISO. Norma iso 23081. Información y documentación. Procesos de gestión de documentos.Metadatos para la gestión de documentos. 1:1–29, 2008. ISSN 0210-0614.

[11] Kathleen Burnett, Kwong Bor Ng, and Soyeon Park. A comparison of the two traditions of metadata development. *Journal of the American Society for Information Science*, 50(13):1–9, 1999. ISSN 00028231.

[12] Iberia. Estructuración de los metadatos. retos principales de la gd. http://www.smile-iberia.com/es/libros_blancos/gestion_documental_open_source/retos_principales_de_la_gd/estructuracion_de_los_metadatos/, 2011. [**Consultado**: Octubre 2011].

[13] EAD. Ead (encoded archival description). http://www.loc.gov/ead/, 2010. [**Consultado**: Noviembre 2011].

[14] Moreq. Moreq (model requirements for the management of electronic records). http://www.dlmforum.eu/, 2010. [**Consultado**: Noviembre 2011].

[15] Alfresco ecm. http://www.alfresco.com/es/, 2011. [**Consultado**: Diciembre 2011].

[16] David Caruana, John Newton, Michael Farman, Michael G. Uzquiano, and Kevin Roast. *Alfresco Professional. Practical Solutions for Enterprise Content Management*. Wiley Publishing, Inc, 2011. ISBN 978-0-470-57104-0. 575 pp.

[17] Snig Bhaumik. *Alfresco 3 Cookbook*. Packt Publishing Ltd., 2011. ISBN 978-1-849511-08-7. 380 pp.

[18] Nuxeo. Nuxeo documentation center home. http://doc.nuxeo.com/display/MAIN/Nuxeo+Documentation+Center+Home/, 2011. [**Consultado**: Diciembre 2011].

[19] Jahia Solutions Group SA. Jahia 6.5 integrator's guide. pages 1–128, 2011.

[20] Jahia. Jahia documentation. www.jahia.com/cms/home/community/documentation/technical-overview/overview/everything-is-content-whatever-t.html/, 2011. [**Consultado**: Diciembre 2011].

[21] KnowledgeTree. Document metadata - KnowledgeTree community. http://www.knowledgetree.org/Document_metadata/, 2009. [**Consultado**: Diciembre 2011].

[22] Creating fieldsets with custom metadata. http://help.knowledgetree.com/entries/20310112-creating-fieldsets-with-custom-metadata/, 2009. [**Consultado**: Diciembre 2011].

[23] PHP. Introducción - manual. http://www.php.net/, 2011. [**Consultado**: Diciembre 2011].

[24] Mozilla Rhino. Rhino - javascript for java. http://www.mozilla.org/rhino/, 2005. [**Consultado**: Diciembre 2011].

[25] Martin Bergljung. *Alfresco 3 Business Solutions*. Packt Publishing Ltd., 2011. ISBN 978-1-849513-34-0. 608 pp.

[26] Jonathan Chaffer and Karl Swedberg. *Learning JQuery*. Pack Publishing, 2007. ISBN 978-1-847192-50-9. 376 pp.

[27] jQuery. jquery: The write less, do more. http://jquery.com/, 2011. [**Consultado**: Diciembre 2011].

[28] Ugo Cei and Piergiorgio Lucidi. *Alfresco 3 Web Services*. Packt Publishing Ltd., 2010. ISBN 978-1-849511-52-0. 436 pp.

[29] Dr. Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. Addison Wesley, sexta edition, 2000. ISBN 0-599-87118-0. 180 pp.

[30] CodeIgniter - open source PHP web application framework. http://www.codeigniter.com/, 2008. [**Consultado**: Diciembre 2011].

[31] Ecured. Visual paradigm for uml. http://www.ecured.cu/index.php/Visual_Paradigm/, 2010. [**Consultado**: Diciembre 2011].

[32] Ivar Jacobson, Grady Booch, and James Rumbaugh. *El Proceso Unificado de Desarrollo de Software*. Addison Wesley, 2000. ISBN 84-7829-036-2. 464 pp.

[33] Suzanne Robertson and James Robertson. *Mastering the requirements process*. Second edition, 2006. ISBN 0321419499. 592 pp.

[34] Adriana Milena Rangel Carrillo. *Análisis comparativo de técnicas para obtención de requerimientos para el módulo de facturación del aplicativo GESTASOFT hospitalario para IMSALUD*. PhD thesis, Facultad de Ingenierías y Arquitecturas, Pamplona Colombia, 2007.

[35] Michael Arias Chaves. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. *Revista InterSedes*, VI(10):1–13, 2007. ISSN 1409-4746.

[36] Ian Sommerville. *Ingeniería del Software*. Addison Wesley, séptima edition, 2005. ISBN 84-7829-074-5. 712 pp.

[37] Roger S. Presman. *Ingeniería del Software. Un enfoque práctico*. Pearson Education, sexta edition, 2005. ISBN 9701054733. 900 pp.

[38] IEEE Computer Society. Ieee std 1471-2000, recommended practice for architectural description of software intensive systems. 2000.

[39] David Garlan and Mary Shaw. *An Introduction to Software Architecture*. Carnegie Mellon University, 1994. ISBN 9810215940. 40 pp.

[40] Craig Larman. *UML y Patrones. Introducción al análisis y diseño orientado a objetos*. Prentice Hall, primera edition, 1999. ISBN 970-17-0261-1. 507 pp.

[41] Santiago Mellano, Cecilia Molina, and Germán Pratelli. *Sistema para la Administración de Programas y Proyectos de Investigación coordinados por la Secretaría de Ciencia y Técnica*. PhD thesis, Universidad Nacional de Río Cuarto, 2010. [**Consultado**: Abril 2011].

[42] Jeysson Poquioma. Ingeniería de software de gestión. implementación. pages 1–24, 2010.

[43] PostgreSQL. Sobre postgresql. http://www.postgresql.org.es/sobre_postgresql/, 2012. [**Consultado**: Abril 2012].

[44] Gustavo Marcelo Torossi. Visión general de rup. pages 1–54, 2007.

[45] William C. Hetzel. *The Complete Guide to Software Testing*. John Wiley and Sons Canada, Ltd, second edition, 1992. ISBN 0471565679. 296 pp.

[46] IEEE Computer Society. *SWEBOK, Software Engineering Body of Knowledge*. Group Managing Editor, 2004. ISBN 0-7695-2330-7. 202 pp.

[47] Roger S. Presman. *Ingeniería del Software. Un enfoque práctico*. McGraw-Hill, quinta edition, 2002. ISBN 8448132149. 640 pp.

Bibliografía

[1] Alfresco, "*What is Alfresco ECM*", 2009. Disponible en la direccion web: "http://www.alfresco.com/es/'. [Consultado Noviembre 2011]

[2] Allepuz Ros, Teresa and Gutiérrez La Rubia, Carmen "*Los sistemas de gestión de la documentación en las organizaciones*". Gabinete de Asesores Documentalistas, 1995

[3] Arias Chaves, Michael, "*La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software*". Revista InterSedes, 2007

[4] Bergljung, Martin, "*Alfresco 3 Business Solutions*". Practical implementation techniques and guidance for delivering business solutions with Alfresco. Packt Publishing Ltd, 2011

[5] Bhaumik, Snig, "*Alfresco 3 Cookbook*". Packt Publishing Ltd, 2011.

[6] Burnett,Kathleen and Bor Ng, Kwong and Park, Soyeon, "*A Comparison of the Two Traditions of Metadata Development*". Journal of the American Society for Information Science, 1999

[7] Bustelo Ruesta, Carlota, "*Los grandes temas relacionados con la gestión de documentos: desafíos y oportunidades*". El profesional de la información, 2011

[8] Canales Mora, Roberto, "*Patrones de GRASP*". Autentia, Real Bussinnes Solutions. Madrid, 2003

[9] Caruana, David and Newton, John and Farman, Michael and Uzquiano, Michael and Roast, Kevin, "*Alfreso Professional. Practical Solutions for Enterprise Content Management*". Wiley Publishing, Inc, 2011

[10] Cei, Ugo and Lucidi, Piergiorgio, "*Alfresco 3 Web Services*". Packt Publishing Ltd, 2010

[11] Cerdá Díaz, Julio, "*Gestión electrónica de documentos y acceso a la información*". Universidad de Alcalá, 2000

[12] Chaffer, Jonathan and Swedberg, Karl, "*Learning JQuery*". Pack Publishing, 2007

[13] CodeIgniter, "*CodeIgniter - Open source PHP web application framework*", 2008. Disponible en la direccion web: "http://www.codeigniter.com/". [Consultado Diciembre 2011]

[14] David Suárez, Michel, "*Propuesta arquitectónica para el desarrollo del Gestor de Documentos Administrativos eXcriba 2.0*". Gestión Documental. UCI, 2010

[15] Delgado Dapena, Martha, "*Definición del modelo del negocio y del dominio utilizando Razonamiento Basado en Casos*", 2008

[16] EAD, "*EAD (Encoded Archival Description)*", 2011. Disponible en la direccion web: "http://www.loc.gov/ead/". [Consultado Marzo 2011]

[17] ECM, "*What is Enterprise Content Management (ECM)*", 2011. Disponible en la direccion web: "http://www.aiim.org/What-is-ECM-Enterprise-Content-Management/". [Consultado Marzo 2011]

[18] Elegir PHP, "*¿Por qué elegir PHP?*". [Consultado Diciembre 2011]. Disponible en la direccion web: "http://www.webtaller.com/maletin/articulos/por-que-elegir-php.php/"

[19] Garlan, David and Shaw, Mary, "*An Introduction to Software Architecture*". ISBN: 9810215940. Carnegie Mellon University. Pittsburgh, 1994

[20] González Faúndez, Juan José, "*Documento de Arquitectura de Software*". Chile, 2008

[21] Gilliland, Anne and Gill, Tony and S Woodley, Mary and Whalen, Maureen, "*Introduction to Metadata*". Getty Research Institute, 2008. Disponible en la direccion web: "http://www.getty.edu/research/publications/electronic_publications/index.html/". [Consultado Octubre 2011]

[22] Hernández León, Rolando Alfredo and Coello González, Sayda, "*El Proceso de Investigación Científica*". Editorial Universitaria del Ministerio de Educación Superior. Cuba, 2011

[23] Hetzel, William, "*he Complete Guide to Software Testing*". John Wiley and Sons Canada, Ltd, 1992

[24] Iberia, "*Estructuración de los metadatos. Retos principales de la GD*". Disponible en la direccion web: "http://www.smile-iberia.com/es/libros_blancos/gestion_documental_open_source/retos_principales_de_la_gd/estructuracion_de_los_metadatos/". [Consultado Octubre 2011]

[25] IEEE Computer Society, "*SWEBOK, Software Engineering Body of Knowledge*". Group Managing Editor, 2004

[26] IEEE Computer Society, "*IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software Intensive Systems*", 2000

[27] ISO 15489, "*Norma ISO 15489. Información y Documentación. Gestión de Documentos*". Proyecto UNE-ISO, 2000

[28] ISO 23081, "*Norma ISO 23081. Información y Documentación. Procesos de Gestión de Documentos.Metadatos para la gestión de documentos*". ISSN: 0210-0614. Proyecto UNE-ISO, 2008

[29] Jacobson, Ivar and Booch, Grady and Rumbaugh,James , "*El Proceso Unificado de Desarrollo de Software*". ISBN: 84-7829-036-2. Addison Wesley, 2000

[30] Jahia, "*Jahia Documentation*", 2011. Disponible en la direccion web: "http://www.jahia.com/cms/home/community/documentation/technical-overview/overview/everything-is-content-whatever-t.html/". [Consultado Diciembre 2011]

[31] Jahia 6.5 Integrator's guide, "*Jahia Solutions Group SA*", 2000

[32] Jiban Pal,"*Metadata initiatives and emerging technologies to improve resource discovery*"

[33] jQuery, "*jQuery: The Write Less, Do More*", 2011. Disponible en la direccion web: "http://jquery.com/". [Consultado Diciembre 2011]

[34] KnowledgeTree, "Jahia Document metadata - KnowledgeTree Community", 2011. Disponible en la direccion web: "http://help.knowledgetree.com/entries/20310112-creating-fieldsets-with-custom-metadata/". [Consultado Diciembre 2011]

[35] KnowledgeTree, "*Document Metadata*", 2011. Disponible en la direccion web: "http://www.knowledgetree.org/Document_metadata/". [Consultado Diciembre 2011]

[36] KT, "*Introdución a KnowledgeTree – Sistema de Gestión Documental*". Disponible en la direccion web: "http://radar.com.co/blog/tag/knowledge-tree/". [Consultado Diciembre 2011]

[37] Lacombe Rocha, Claudia, "*Metadatos: Concepto, Funciones y Tipos*". Archivo Nacional. Brasil, 2007

[38] Larman, Craig, "*UML y Patrones. Introducción al análisis y diseño orientado a objetos*". Prentice Hall, 1999

[39] Libro Blanco, "*Gestión Documental Open Source*". Smile open source solutions. Barcelona, 2008.

[40] Marcelo Torossi, Gustavo, "*Visión General de RUP*", 2007

[41] Mena Mugica, Mayra, "*Utilidad de las soluciones archivísticas para la gestión de la información en los sistemas electrónicos del sector de la salud*". ISSN: 1409-4746, 2007.

[42] Méndez Rodríguez, Eva, "*La descripción de documentos electrónicos a través de metadatos: una visión para la Archivística desde la nueva e-Administración*". Revista d'Arxius, 2003.

[43] Mendoza Navarro, Aída Luz, "*La realidad latinoamericana en gestión de documentos electrónicos*". España, 2010

[44] Mellano, Santiago and Molina, Cecilia and Pratelli,Germán, "*Sistema para la Administración de Programas y Proyectos de Investigación coordinados por la Secretaría de Ciencia y Técnica*". Universidad Nacional de Río Cuarto, 2010. Disponible en la direccion web: "http://www.scribd.com/doc/54155511/Tesis-APPI-Java/". [Consultado Abril 2011]

[45] Moreq, "*Moreq (Encoded Archival Description)*", 2010. Disponible en la direccion web: "http://www.dlmforum.eu/". [Consultado Noviembre 2011]

[46] Mozilla Rhino, "*Rhino - JavaScript for Java*", 2005. Disponible en la direccion web: "http://www.mozilla.org/rhino/". [Consultado Diciembre 2011]

[47] NISO (National Information Standards Organization), "*Understanding Metadata*". ISBN: 1-880124-62-9. NISO Press, 2004.

[48] Nuxeo, "*Nuxeo Documentation Center Home*", 2011. Disponible en la direccion web: "http://doc.nuxeo.com/display/MAIN/Nuxeo+Documentation+Center+Home/". [Consultado Diciembre 2011]

[49] Nuxeo Digital Asset Management, "*Metadata Customization*". Disponible en la direccion web: "http://www.nuxeo.com/en/products/digital-asset-management/". [Consultado Diciembre 2011]

[50] Pereira, J, "*Introducción a la Gestión Documental*", 2010

[51] PHP, "*Introducción - Manual*". Disponible en la direccion web: "http://www.php.net/". [Consultado Diciembre 2011]

[52] Presman, Roger, "*Ingeniería del Software. Un enfoque práctico*", 6ta edition. Pearson Education, 2005

[53] Poquioma, Jeysson, "*Ingeniería de Software de Gestión. Implementación*", 2010

[54] Prieto, Félix, "*Patrones de diseño*". Universidad de Valladolid, 2008

[55] Protocolos, "*Protocolos de Comunicación*". Disponible en la direccion web: "http://www.masadelante.com/". [Consultado Marzo 2011]

[56] RAE, "*Diccionario de la Real Academia Española*". Disponible en la direccion web: "http://www.rae.es/rae.html/". [Consultado Octubre 2011]

[57] Rangel Carrillo, Adriana Milena, "*Análisis comparativo de técnicas para obtención de requerimientos para el módulo de facturación del aplicativo GESTASOFT hospitalario para IMSALUD*". Facultad de Ingenierías y Arquitecturas. Colombia, 2007

[58] Real Decreto 4/2010, "*Esquema Nacional de Interoperabilidad en el ámbito de la Administración Electrónica*". España, 2010

[59] Reynoso, Carlos and Kicillof, Nicolás, "*Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*"

[60] Robertson, Suzanne and Robertson, James, "*Mastering the requirements process*". ISBN: 0321419499. Boston, 2006

[61] Slater, Jenny, "*A Guide to Metadata*". CETIS Metadata SIG, 2002

[62] Sommerville, Ian, "*Ingeniería del Software*". ISBN: 84-7829-074-5. Addison Wesley, 2005

[63] PostgreSQL, "*Sobre PostgreSQL*". Disponible en la direccion web: "http://www.postgresql.org.es/sobre_postgresqlt/". [Consultado Abril 2011].

[64] Suárez, Pablo and Fontela, Carlos, "*Documentación y pruebas. Antes del paradigma de objetos*", 2003

[65] Thomas Fielding, Roy, "*Architectural Styles and the Design of Network-based Software Architectures*". ISBN: 0-599-87118-0. Addison Wesley, 2000

[66] Vásquez Paulus, Cristian, "*Metadatos: Introducción e historia*". Chile, 2008

[67] Visual Paradigm for UML, "*Visual Paradigm - EcuRed*", 2010. Disponible en la direccion web: "http://http://www.ecured.cu/index.php/Visual_Paradigm/". [Consultado Diciembre 2011].

Glosario de términos

A

agente En el contexto de la Gestión Documental se define como individuo, grupo de trabajo u organización responsable o involucrado en la creación de documentos, en su incorporación al sistema así como en otros procesos de gestión de documentos.

API Application Programming Interface: Es una llave de acceso a funciones que nos permiten hacer uso de un servicio web provisto por un tercero, dentro de una aplicación web propia, de manera segura.

C

CASE Computer Aided Software Engineering (Ingeniería de Software Asistida por Ordenador), aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

E

esquema Plan lógico que muestra las relaciones entre los distintos elementos del conjunto de metadatos, normalmente mediante el establecimiento de reglas para su uso y gestión y específicamente respecto a la semántica, la sintaxis y la obligatoriedad de los valores.

F

fiabiliad Probabilidad de buen funcionamiento de algo.

framework Plataformas o herramientas del mundo de la informática que le proveen a los programadores un grupo de facilidades en el ámbito para la cual han sido creadas.

G

GDA Gestor de Documentos Administrativos: Grupo de trabajo perteneciente al departamento de Gestión Documental.

H

hardware Dispositivos que están conectados físicamente al ordenador.

HTTP HyperText Transfer Protocol: Es el protocolo de transferencia de hipertexto, es el método más común de intercambio de información en la World Wide Web, el método mediante el cual se transfieren las páginas web a un ordenador.

I

IDE Integrate Development Enviroment: Entorno de desarrollo integrado. Herramienta que se usa para facilitar el desarrollo de software.

interoperabilidad Capacidad de los sistemas de información y de los procedimientos a los que éstos dan soporte, de compartir datos y posibilitar el intercambio de información y conocimiento entre ellos.

J

J2EE Java Platform Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4), es una plataforma de programación — parte de la Plataforma Java — para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de N capas.

Jam Warehouse Es una de las mejores compañías desarrolladoras de software a medida en Sudáfrica.

S

software Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

T

TCP/IP Transmission Control Protocol/Internet Protocol: son las siglas de Protocolo de Control de Transmisión/Protocolo de Internet, protocolo que hace posible la transferencia de datos entre redes de ordenadores.

testeo Someter algo a un control o prueba.

The Federal Records Act Organización que establece el marco de trabajo para los programas de gestión de expedientes en las Agencias Federales. Este marco de trabajo se realiza mediante evaluación de los documentos (para determinar el valor de registro y disposición final de los expedientes temporales o permanentes), la regulación y aprobación de la disposición de los registros federales, que operan Centros de Registros Federales y la preservación de los registros permanentes.

U

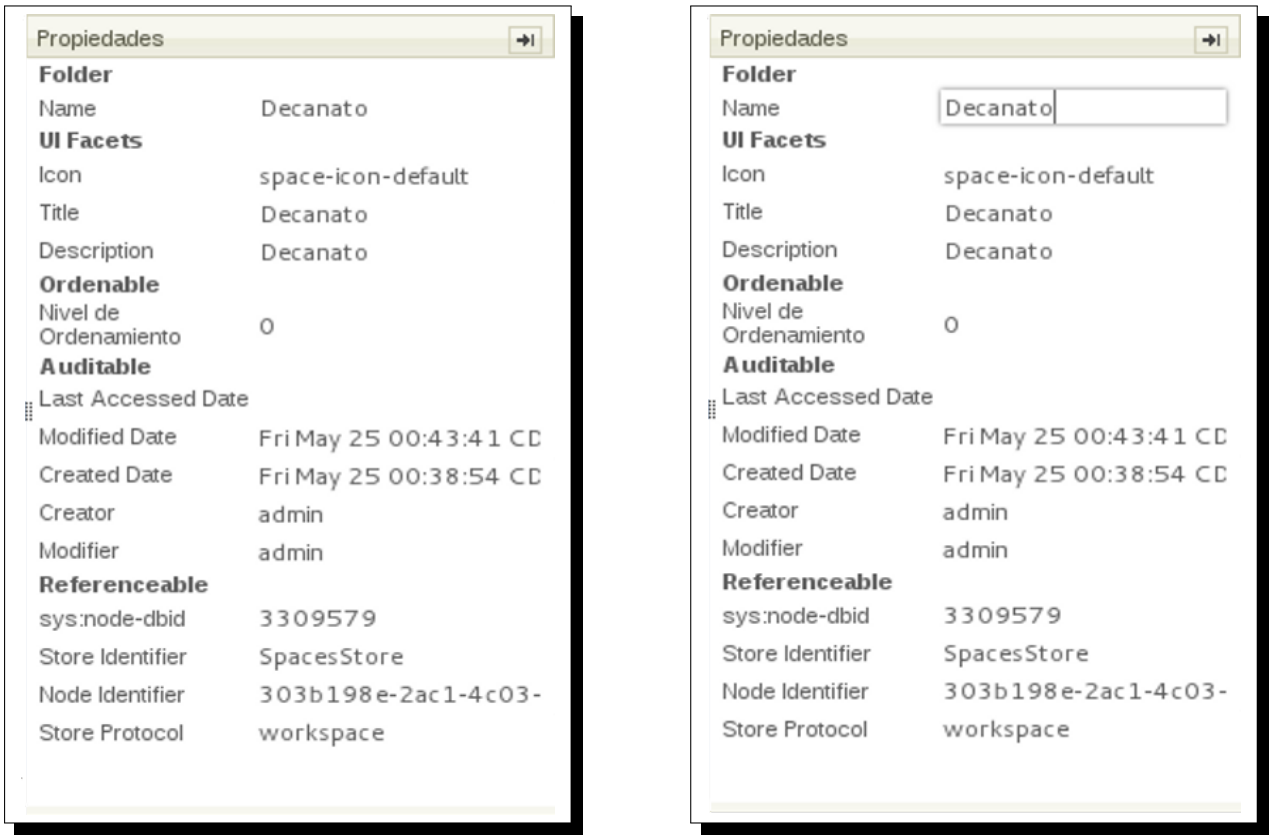
UCI Universidad de las Ciencias Informáticas.

UML Unified Modeling Language: Lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

URN Uniform Resource Name: Nombre Uniforme de Recursos, son cadenas de texto que se usan para nombrar recursos con el objetivo de lograr su identificación, sirven para identificar pero no para localizar (URI = URL + URN).

Anexo A

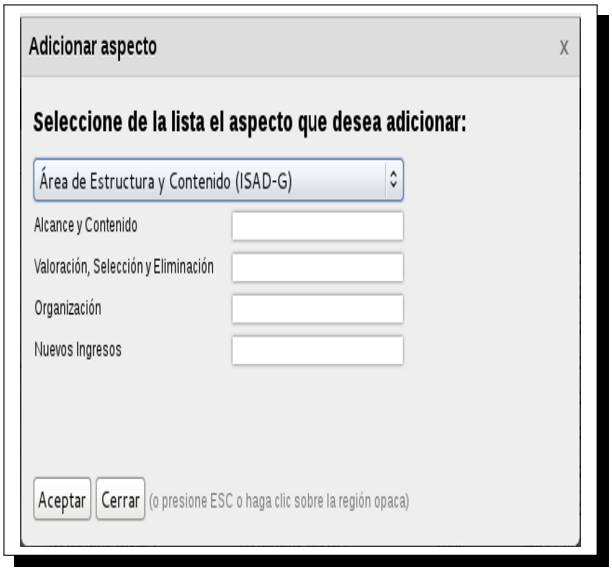
Prototipos de los requisitos funcionales



(a) Prototipo del RF.1

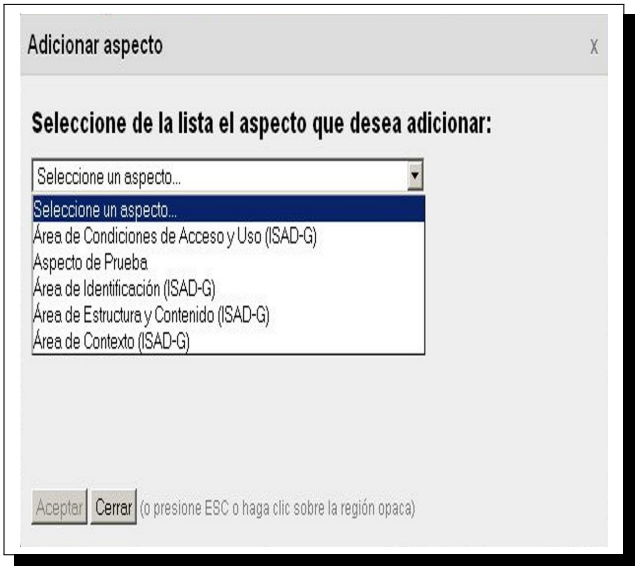
(b) Prototipo del RF.2

Figura A.1: Prototipos del caso de uso gestionar metadatos de un contenido



(a) Prototipo del RF.3

(b) Prototipo del RF.4



(c) Prototipo del RF.5

Figura A.2: Prototipos del caso de uso gestionar aspectos de un contenido

Anexo B

Diseño del Caso de Uso gestionar aspectos de un contenido. Sección: adicionar

B.1. Diagrama de clases del diseño

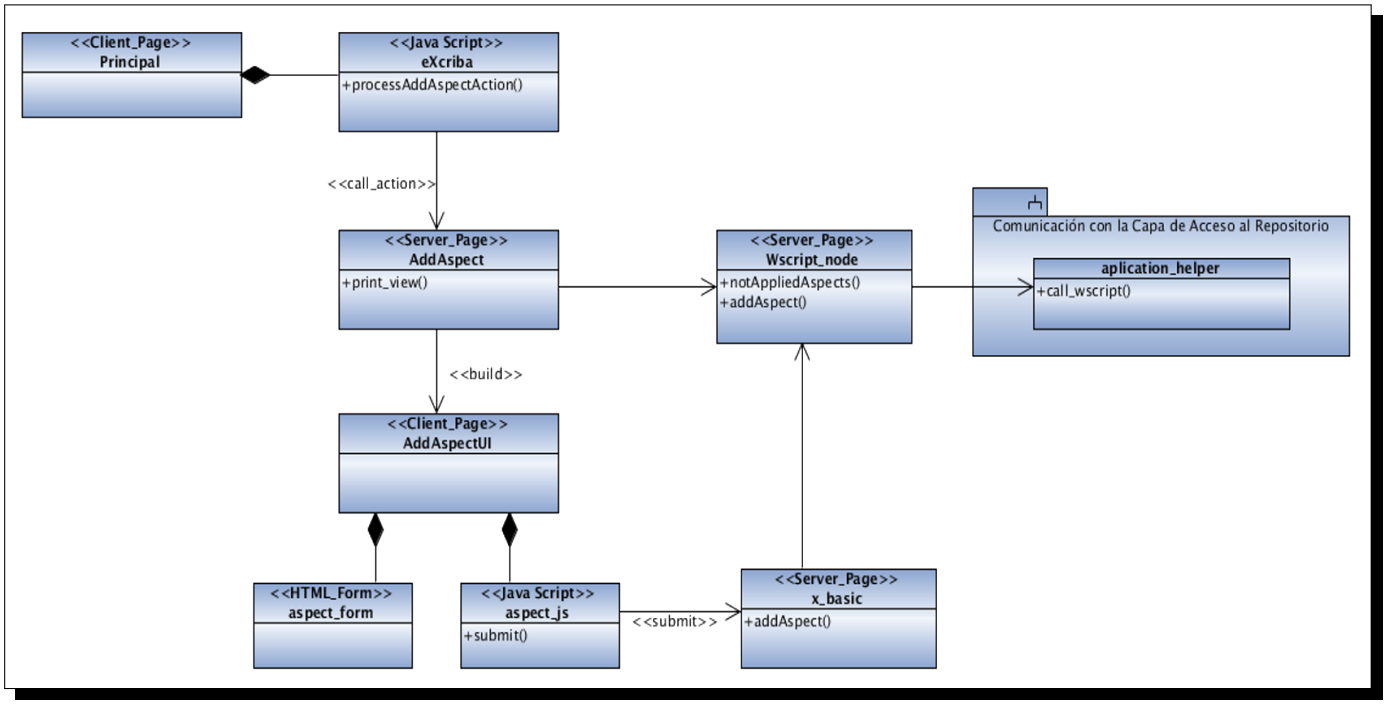


Figura B.1: Diagrama de clases del diseño del CU Gestionar aspectos. Sección: adicionar

B.2. Descripción de las clases del diseño

A continuación se exponen en forma de tablas las principales clases que intervienen en el caso de uso *Gestionar aspectos de un contenido*, de la sección adicionar, así como las funcionalidades identificadas como necesarias exclusivamente para la realización del mismo.

Nombre: AddAspect	
Tipo de clase: Controladora	
Descripción: Coordina las operaciones que deben llevarse a cabo para generar el formulario donde el usuario podrá adicionar nuevos metadatos a un documento o carpeta a través de la definición de aspectos.	
Responsabilidades:	
Nombre	**Descripción**
print_view()	Genera la vista donde el usuario seleccionará el aspecto a adicionar a un documento o carpeta específico.

Tabla B.1: Descripción de la clase AddAspect

Nombre: Wscript_node	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para hacer la llamada a los diferentes servicios disponibles.	
Responsabilidades:	
Nombre	**Descripción**
notAppliedAspects()	Prepara los datos necesarios para hacer la llamada al servicio web notAppliedAspects que obtiene una lista de los aspectos que aún no han sido aplicados a algún nodo.

addAspect()	Prepara los datos necesarios para hacer la llamada al servicio web addAspect el cual adiciona nuevas propiedades a un documento o carpeta previamente seleccionado, para ello recibe como parámetros el identificador del documento o carpeta, el nombre del aspecto a añadir, la lista de propiedades que se adicionarán al elemento y los valores de dichas propiedades.

Tabla B.2: Descripción de la clase Wscript_node

Nombre: x_basic	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para interactuar con la capa de Acceso al Repositorio. Su función es actuar como la interfaz a través de la cual se comunica la capa de aplicación/negocio (específicamente las funcionalidades del módulo x_basic_operations) con la de Acceso al Repositorio.	
Responsabilidades:	
Nombre	**Descripción**
addAspect()	Coordina las operaciones necesarias que deben llevarse a cabo para adicionar un aspecto, entre ellas interactuar con la Capa de Acceso al Repositorio.

Tabla B.3: Descripción de la clase x_basic

B.3. Descripción de los servicios

Para poder realizar la funcionalidad adicionar aspectos se hace imprescindible un servicio que posibilite obtener el listado de todos los aspectos que puedan adicionarse a un nodo específico, es decir, aquellos aspectos que aún no han sido aplicados al nodo. El objetivo de este servicio es impedir que el usuario pueda asociarle un aspecto a un documento o carpeta que ya lo contiene, por lo que se propone la implementación del servicio **NotAppliedAspects**, descrito en la Tabla A.4, además se hace necesario el uso del servicio ya existente **AddAspect**, el cual posibilitará una vez obtenido el resultado del servicio NotAppliedAspects incorporarle nuevos metadatos a un nodo, dígase documento o carpeta.

Servicio: NotAppliedAspects	
Descripción:	Devuelve una lista de los aspectos que no han sido asociados a un nodo.
URL:	http://<host>:<puerto>/alfresco/service/cu/uci/excriba/node/storeType/storeId/nodeUUID/aspects/not-applied?subtypes-of=parent-aspects
Requerimiento de Autenticación:	Usuario
Requerimiento de Transacción:	Requerida
Formato de Plantilla de Respuesta:	JSON
Paquete:	cu.uci.excriba.node
Método HTTP:	GET
Lenguaje de implementación del controlador:	Java
Clase:	cu.uci.excriba.wscripts.node.NotAppliedAspects
Parámetros o Argumentos de entrada	
Nombre y Descripción:	**Formato/Ejemplo**
storeType: identificador del tipo de almacén.	workspace

storeId: identificador del almacén (store).	SpacesStore
nodeUUID: identificador del nodo.	ab3eff9c-f868-44f0-b351-bdb657f6d57b
subtypes-of: listado de aspectos de los cuales deben heredar los aspectos devueltos.	aspecto1,aspecto2,aspecto3,....,aspectoN
Parámetros o Argumentos de salida	
Nombre y Descripción:	**Formato/Ejemplo**

aspects: listado de aspectos aplicados al nodo	[
	"nombre_del_aspecto" : {
	"name" : "nombre_del_aspecto",
	"isAspect" : true,
	"title" : "título_del_aspecto",
	"description" : "descripción_del_aspecto",
	"parent" : {
	"name" : "nombre_del_padre",
	"title" : "título_del_padre",
	"url" : "url_del_padre"
	},
	"defaultValues" : {
	"valores_por_defecto"
	},
	"defaultAspects" : {
	"aspectos_por_defecto"
	},
	"properties" : {
	"nombre de la propiedad": {
	"name" : "nombre de la propiedad",
	"title" : "título de la propiedad",
	"defaultValues" : "valores por defecto",
	"dataType" : "tipo de dato de la propiedad",
	"multiValued" : multivaluado(true/false),
	"mandatory" : obligatorio(true/false),
	"protected" : protegido(true/false),
	"url" : "url de la propiedad"
	},
	"url" : "url del aspecto"
	}
	],

errors: listado de errores	[
	"errors" : {
	"nombre del aspecto": "descripción del error "
	}
	]

Tabla B.4: Descripción del servicio NotAppliedAspects

AddAspect se implementó con el objetivo de adicionar un conjunto de nuevos metadatos a los nodos, para ello recibe como paramétros el identificador del nodo, el nombre del aspecto y la lista de propiedades contenidas dentro del aspecto a adicionar con sus respectivos valores.

B.4. Diagrama de interacción del diseño

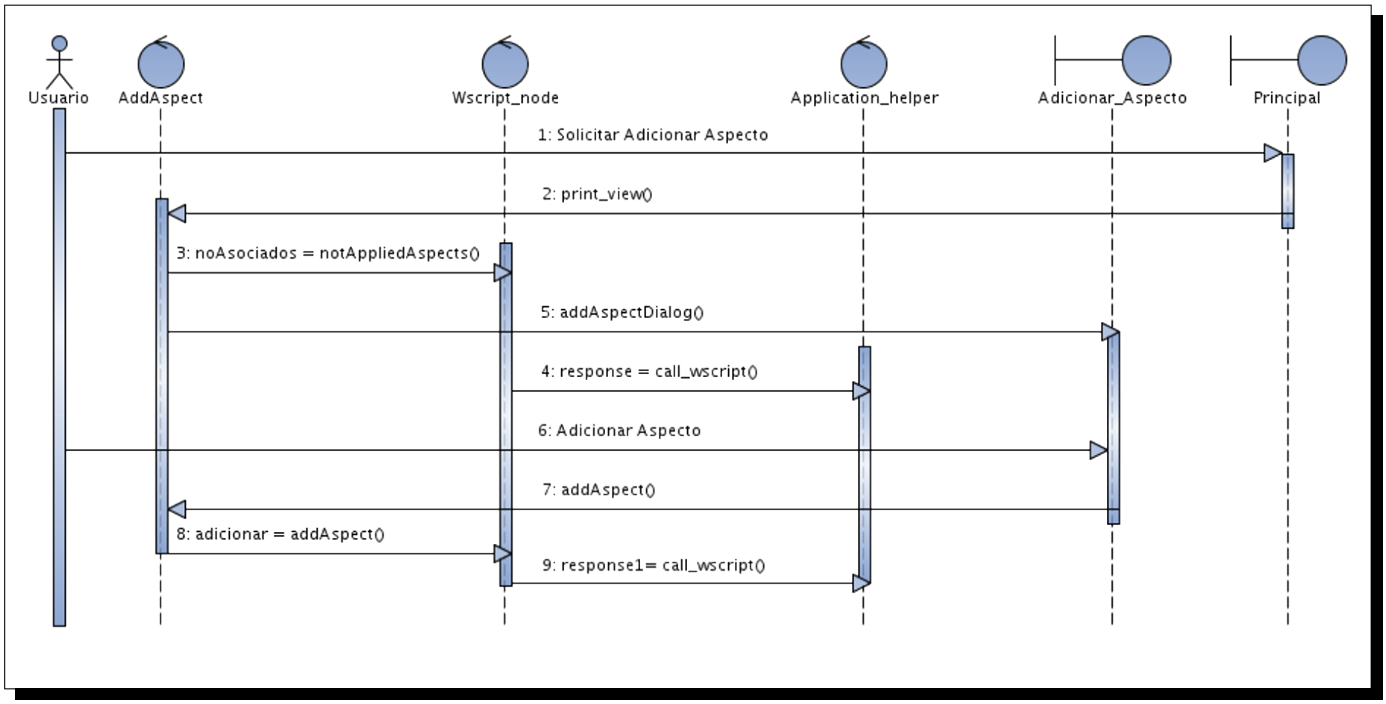


Figura B.2: Diagrama de interacción del diseño del CU Gestionar aspectos. Sección: adicionar

Anexo C

Diseño del Caso de Uso gestionar aspectos de un contenido. Sección: eliminar

C.1. Diagrama de clases del diseño

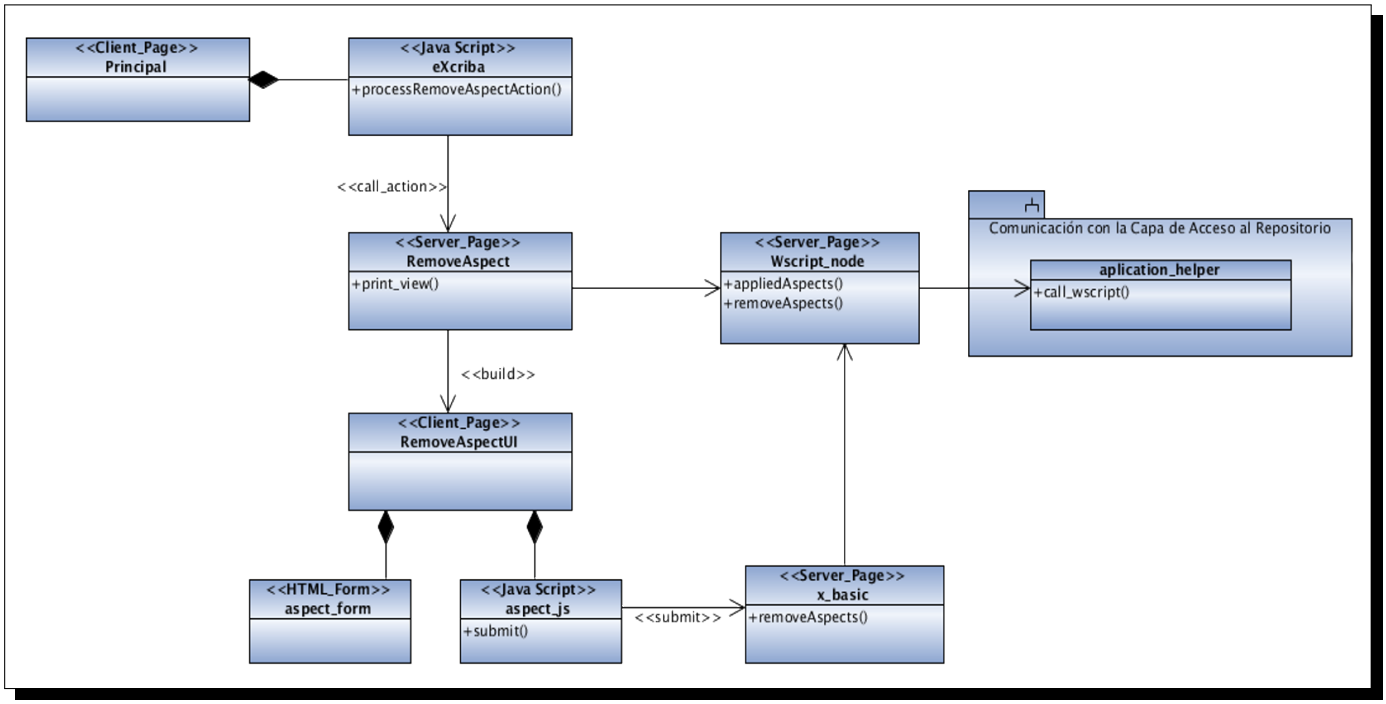


Figura C.1: Diagrama de clases del diseño del CU Gestionar aspectos. Sección: eliminar

C.2. Descripción de las clases del diseño

A continuación se exponen en forma de tablas las principales clases que intervienen en el caso de uso *Gestionar aspectos de un contenido* de la sección eliminar, así como las funcionalidades identificadas como necesarias exclusivamente para la realización del mismo.

Nombre: RemoveAspect	
Tipo de clase: Controladora	
Descripción: Coordina las operaciones que deben llevarse a cabo para generar el formulario donde el usuario podrá eliminar uno o varios aspectos a un documento o carpeta.	
Responsabilidades:	
Nombre	**Descripción**
print_view()	Genera la vista donde el usuario seleccionará el aspecto a adicionar a un documento o carpeta específico.

Tabla C.1: Descripción de la clase RemoveAspect

Nombre: Wscript_node	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para hacer la llamada a los diferentes servicios disponibles.	
Responsabilidades:	
Nombre	**Descripción**
appliedAspects()	Prepara los datos necesarios para hacer la llamada al servicio web appliedAspects que obtiene una lista de los aspectos aplicados a un documento o carpeta seleccionado anteriormente.

removeAspects()	Prepara los datos necesarios para hacer la llamada al servicio web removeAspect el cual elimina un conjunto de propiedades a un documento o carpeta previamente seleccionado, para ello recibe como parámetros el identificador del documento o carpeta y el listado de aspectos a sustraer.

Tabla C.2: Descripción de la clase Wscript_node

Nombre: x_basic	
Tipo de clase: Controladora	
Descripción: Contiene los métodos necesarios para interactuar con la capa de Acceso al Repositorio. Su función es actuar como la interfaz a través de la cual se comunica la capa de aplicación/negocio (específicamente las funcionalidades del módulo x_basic_operations) con la de Acceso al Repositorio.	
Responsabilidades:	
Nombre	**Descripción**
removeAspects()	Coordina las operaciones necesarias que deben llevarse a cabo para eliminar uno o varios aspectos, entre estas operaciones se encuentra interactuar con la Capa de Acceso al Repositorio.

Tabla C.3: Descripción de la clase x_basic

C.3. Descripción de los servicios

Para poder eliminar uno o varios aspectos a un nodo es necesario el desarrollo de dos servicios que permitan ejecutar esta acción en el sistema, uno que obtenga los aspectos asociados un nodo específico y otro servicio que elimine cualquier aspecto de los que se listen con el primer servicio. Se implementarán en función de los antes mencionado los servicios **AppliedAspects** y **RemoveAspects**, dichos servicios son explicados detalladamente a continuación.

Servicio: AppliedAspects	
Descripción:	Devuelve una lista de aspectos aplicados o asociados a un nodo.
URL:	http://<host>:<puerto>/alfresco/service/cu/uci/excriba/node/storeType/storeId/nodeUUID/aspects/list/applied?subtypes-of=parent-aspects
Requerimiento de Autenticación:	Usuario
Requerimiento de Transacción:	Requerida
Formato de Plantilla de Respuesta:	JSON
Paquete:	cu.uci.excriba.node
Método HTTP:	GET
Lenguaje de implementación del controlador:	Java
Clase:	cu.uci.excriba.wscripts.node.AppliedAspects
Parámetros o Argumentos de entrada	
Nombre y Descripción:	**Formato/Ejemplo**
storeType: identificador del tipo de almacén.	workspace
storeId: identificador del almacén (store).	SpacesStore
nodeUUID: identificador del nodo.	ab3eff9c-f868-44f0-b351-bdb657f6d57b

subtypes-of: listado de aspectos de los cuales deben heredar los aspectos devueltos	aspecto1,aspecto2,aspecto3,....,aspectoN
Parámetros o Argumentos de salida	
Nombre y Descripción:	**Formato/Ejemplo**

aspects: listado de aspectos aplicados al nodo.	[
	"nombre_del_aspecto" : {
	"name" : "nombre_del_aspecto",
	"isAspect" : true,
	"title" : "título_del_aspecto",
	"description" : "descripción_del_aspecto",
	"parent" : {
	"name" : "nombre_del_padre",
	"title" : "título_del_padre",
	"url" : "url_del_padre"
	},
	"defaultValues" : {
	"valores_por_defecto"
	},
	"defaultAspects" : {
	"aspectos_por_defecto"
	},
	"properties" : {
	"nombre de la propiedad": {
	"name" : "nombre de la propiedad",
	"title" : "título de la propiedad",
	"defaultValues" : "valores por defecto",
	"dataType" : "tipo de dato de la propiedad",
	"multiValued" : "multivaluado(true/false)",
	"mandatory" : "obligatorio(true/false)",
	"protected" : "protegido(true/false)",
	"url" : "url de la propiedad"
	},
	"url" : "url del aspecto"
	}
	]

errors: listado de errores.	[
	"errors" : {
	"nombre del aspecto": "descripción del error"
	}
	]

Tabla C.4: Descripción del servicio AppliedAspects

Servicio: RemoveAspects	
Descripción:	Elimina un conjunto de aspectos de un nodo.
URL:	http://<host>:<puerto>/alfresco/service/cu/uci/excriba/node/storeType/storeId/nodeUUID/aspects/remove?selected=aspects
Requerimiento de Autenticación:	Usuario
Requerimiento de Transacción:	Requerida
Formato de Plantilla de Respuesta:	JSON
Paquete:	cu.uci.excriba.node
Método HTTP:	GET
Lenguaje de implementación del controlador:	Java
Clase:	cu.uci.excriba.wscripts.node.RemoveAspects
Parámetros o Argumentos de entrada	
Nombre y Descripción:	**Formato/Ejemplo**
storeType: identificador del tipo de almacén.	workspace
storeId: identificador del almacén (store).	SpacesStore
nodeUUID: identificador del nodo.	ab3eff9c-f868-44f0-b351-bdb657f6d57b
selected: listado de aspectos que se desean eliminar.	aspecto1,aspecto2,aspecto3,....,aspectoN
Parámetros o Argumentos de salida	
Nombre y Descripción:	**Formato/Ejemplo**

removed: listado de aspectos que fueron eliminados	`"removed" : {`
	`  aspecto_eliminado`
	`},`

errors: listado de errores	"errors" : {
	"nombre del aspecto": "descripción del error"
	}

Tabla C.5: Descripción del servicio RemoveAspects

C.4. Diagrama de interacción del diseño

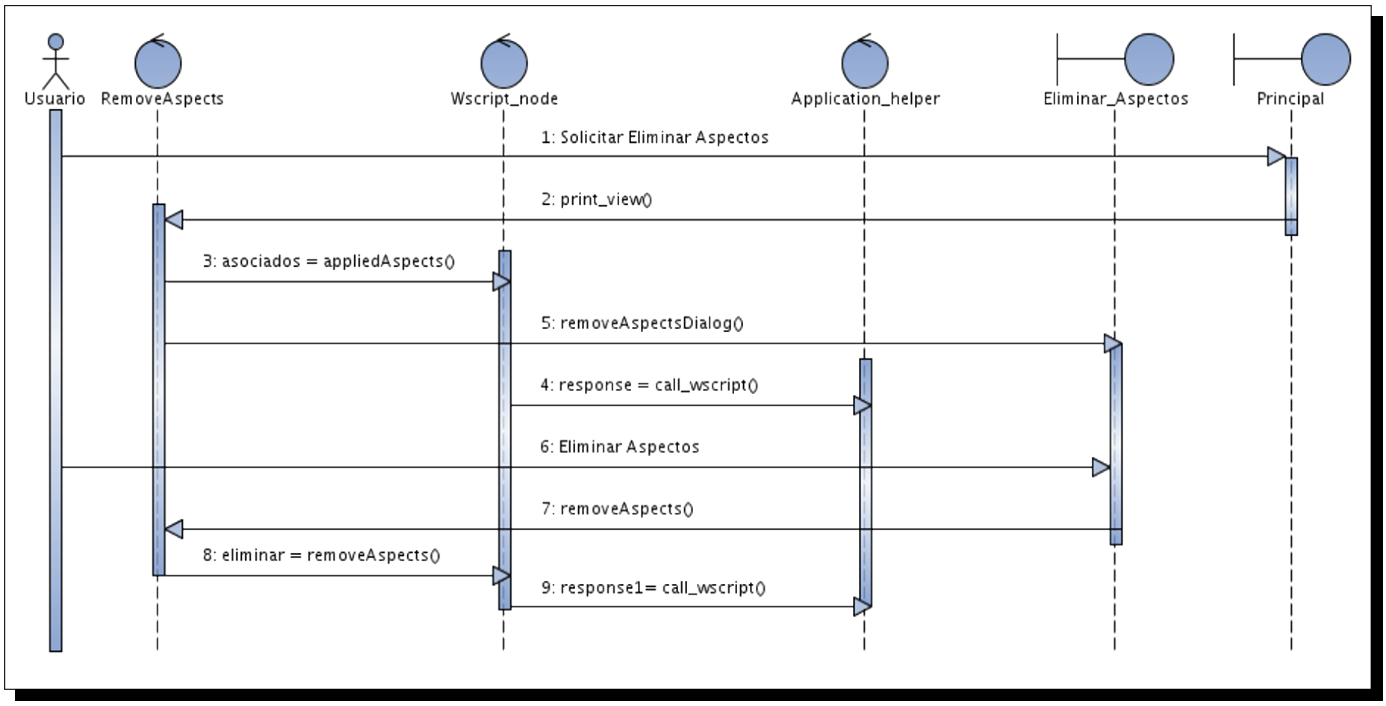


Figura C.2: Diagrama de interacción del diseño del CU Gestionar aspectos. Sección: eliminar