Universidad de las Ciencias Informáticas

Facultad 6

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

Título

Desarrollo de un componente para la gestión de tipologías de archivos multimedia en aplicaciones web.

Autor

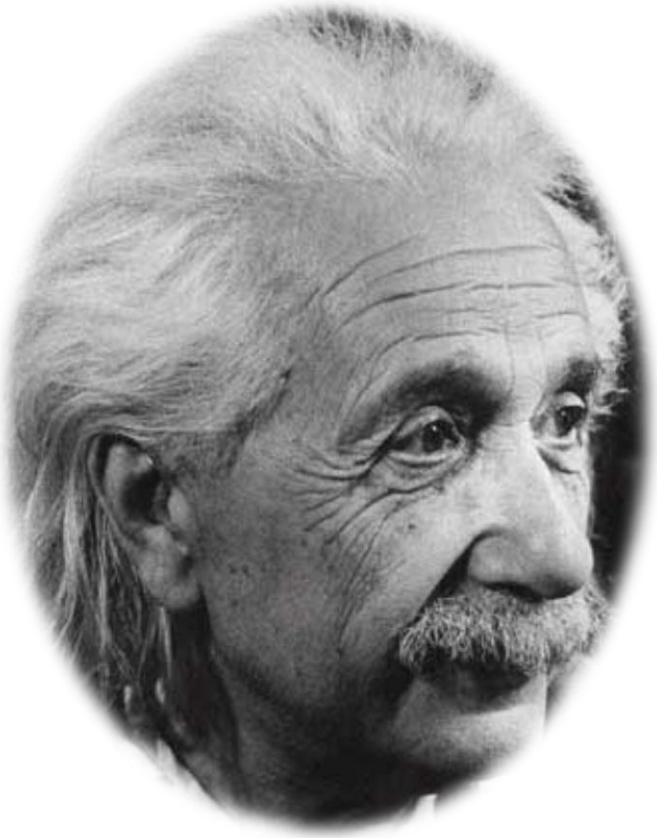
Iván Betancourt Rodríguez

Tutor

Frank Torres Rodríguez

Ciudad de La Habana, junio de 2011

‘‘Año 53 de la Revolución’’



La mayoría de las ideas fundamentales de la ciencia son esencialmente sencillas y, por regla general pueden ser expresadas en un lenguaje comprensible para todos.

Albert Einstein

Declaración de Autoría

Declaramos ser los autores de la presente tesis y autorizamos a la Universidad de las Ciencias Informáticas para que haga uso de la misma, como estime pertinente, cediéndole de esta forma los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____ Firma del Autor
Iván Betancourt Rodríguez

_____ Firma del Tutor
Frank Torres Rodríguez

Agradecimientos

A la Universidad de las Ciencias Informáticas por forjarme como profesional.

A toda mi familia, por siempre estar ahí junto a mí.

A mis amistades y a todas las personas que he tenido el placer de conocer en esta maravillosa etapa de mi vida.

A mi tutor Frank por estar pendiente siempre de mi trabajo y ayudarme en todo lo que me hizo falta.

A todas aquellas personas que de una forma u otra me han ayudado a alcanzar mis sueños, los llevaré siempre muy presente, pues el agradecimiento es algo que se lleva todos los días en la mente y el corazón.

Gracias a todos porque parte de lo que soy se los debo a ustedes.

Iván

Dedicatoria

A mis padres Santo y Deisy por ayudarme siempre, por haberme brindado apoyo, amor y cariño. Gracias a su esfuerzo se hizo realidad este sueño.

A mi hermano por haber estado siempre en los momentos más duros.

A mi novia por ayudarme, brindarme su amor y comprensión en los momentos más difíciles, gracias por todo Tita.

A todas las personas que de una forma u otra han contribuido a convertirme en lo que soy hoy.

Iván

Resumen

La gestión de tipologías de archivos multimedia es un proceso de suma importancia para cualquier aplicación en la que se manejen un gran número de multimedia con diferentes fines, pues de esta gestión dependen los metadatos que se almacenen de cada una de ellas, los cuales facilitan los proceso de búsqueda, procesamiento y posterior utilización de dichas multimedia. El presente trabajo de diploma contiene la investigación y desarrollo de un componente para la gestión de tipologías de archivos multimedia.

Para el desarrollo de este componente se utilizó RUP como metodología de desarrollo y Visual Paradigm v6.4 como herramienta de modelado. Como Entorno de Desarrollo Integrado (IDE) se utilizó NetBeans v6.9, el Sistema Gestor de Bases de Datos fue PostgreSQL v8.4, el lenguaje de programación en que se desarrolló fue PHP v5.3.2 y como servidor web su utilizó Apache v2.2. Todas herramientas y tecnologías libres.

Con el desarrollo de este componente y posterior puesta en práctica se espera facilitar la gestión de tipologías en aplicaciones web, la cual es una necesidad del Departamento de Señales Digitales y podrá ser utilizado en aplicaciones web, tales como el portal Inter-Nos de la Universidad de las Ciencias Informáticas (UCI).

Palabras Claves: archivos multimedia, tipologías, metadatos, catalogación, componente.

Índice de Contenido

Índice de Tablas y Figuras

Introducción

La informática como ciencia surge a finales del siglo XIX y en pocos años experimentó un auge vertiginoso, lo que posibilitó que su uso fuera más allá de realizar simples operaciones de cómputo. Las computadoras fueron ganando en propiedades y funcionalidades y en la década de los 90 del pasado siglo se comenzaron a utilizar para la creación, manipulación y reproducción de archivos multimedia. Además, el aumento de la capacidad en discos duro de las computadoras permitió que cada vez se pudieran almacenar mayor cantidad de videos, de los cuales se hacía necesario guardar información en dependencia de su tipo.

El surgimiento del Internet y el aumento de las velocidades de las redes de computadoras sentaron las bases para que los videos se pudieran difundir por la red y a su vez permitieron que surgieran sitios especializados en la presentación de materiales audiovisuales de diferentes tipos. En el ámbito internacional se pueden mencionar algunos como www.youtube.com, www.tv.com, www.videowebperu.com, www.NBC.com y ww.video.aol.com. En Cuba el uso de esta tecnología no está tan difundido, los aportes más significativos los ha realizado la Universidad de las Ciencias Informáticas con el desarrollo del portal Inter-Nos para la publicación de conferencias, canales de televisión en vivo y otros materiales audiovisuales. (Abreu, 2009)

Con el aumento de los videos en la red surge la necesidad de estandarizar la información que se almacena de estos; así surgen los metadatos que son datos que se encargan de mantener un registro sobre el significado, contexto o propósito de un objeto informativo, de tal forma que permita descubrir, entender, extraer y administrar dicho objeto. (Paulus, 2008)

Los metadatos tienen sus orígenes en los catálogos, que son listas ordenadas alfabéticamente sin criterios de clasificación sofisticados, que complican considerablemente la recuperación de información. Un avance importante en cuanto a esquemas de clasificación se desarrolla alrededor del año 1900 cuando los catálogos de libros son reemplazados completamente por tarjetas. En la década del sesenta del pasado siglo con el auge de los métodos de producción en masa se hacen necesario disponer de múltiples copias de los catálogos existentes; surgen masivas colecciones distribuidas de libros y los catálogos de tarjetas no logran satisfacer los nuevos requerimientos. Entonces se hizo necesario desarrollar estándares de codificación, llamados hoy en día metadatos. (Serrano, 2008)

Los primeros metadatos (digitales) y sus bases se desarrollaron a finales del siglo XX, que es cuando emergen múltiples estándares de codificación, lenguajes y protocolos que se utilizan en la generación y uso de catálogos. Entre ellos se encuentran (Serrano, 2008):

1. Machine Readable Cataloguing (MARC): El MARC fue un gran avance porque permitió el intercambio de información, el acceso a catálogos colectivos y la catalogación compartida.
2. ISO Z39.50: es un protocolo para la generación de consultas a lo largo de múltiples catálogos online. De origen estadounidense, data de 1988, momento en que fue aprobado por la NISO (National Information Standards Organization) y permite a un usuario de un sistema buscar y recuperar la información sin saber la sintaxis utilizada por los otros sistemas.
3. Standard Generalized Markup Language (SGML): Es un estándar internacional que consta de un conjunto de reglas para describir la estructura de un documento de tal forma que puedan ser intercambiados a través de las plataformas computacionales. SGML es extremadamente flexible y es la base de los lenguajes de marcado más utilizados hoy en día.
4. Document Type Definition (DTD): Son aplicaciones de SGML utilizadas para definir las estructuras de un tipo de documento en especial.

En el mundo del video y sonido digital uno de los primero pasos que se da en la creación de metadatos es el MPEG-7, que se establece como estándar en marzo del 2000 en la 51ª Convención de MPEG desarrollada en Noordwijkerhout. (Ballesteros, 2006)

Un término muy asociado a los metadatos son las tipologías de archivos multimedia, que es una clasificación de los tipos de multimedia basados en un estudio hecho previamente en el ámbito donde se difunden. Dependiendo de la tipología y el estándar de metadato será la información que se almacenará del video, la cual servirá posteriormente para su búsqueda y utilización.

Siguiendo estas tendencias surge en la Universidad de las Ciencias Informáticas la Plataforma VideoWeb, una plataforma para la transmisión de contenidos audiovisuales a través de la red de datos. Esta plataforma pertenece al Departamento de Señales Digitales el cual se dedica desarrollar productos, servicios y soluciones informáticas en el campo del procesamiento de Señales Digitales. Este departamento necesita gestionar las tipologías de sus archivos multimedia de forma homogénea y estandarizar la información que se almacena de estos archivos, pues el departamento está compuesto por varios proyectos, entre ellos la Plataforma VideoWeb, que están desarrollando aplicaciones de forma independiente y en ocasiones son para un mismo cliente, por lo que se hace necesario que la información

de los archivos multimedia que se almacene y la forma en que se manipule sea la misma en cada una de las aplicaciones en desarrollo. Además este proceso se debe realizar siguiendo alguno de los estándares internacionales en materia de metadatos para multimedia que permita la compatibilidad en un futuro con otras aplicaciones y obtener esta información de los archivos multimedia de forma automática. También es necesario realizar este proceso de forma dinámica, de tal manera que se pueda personalizar la información que se va a almacenar de los archivos multimedia, aunque siguiendo alguno de los estándares de metadatos que permiten la personalización de los campos a almacenar.

Situación que permite plantear el siguiente **problema a resolver:** ¿Cómo facilitar la gestión dinámica de la información asociada a los archivos multimedia en las aplicaciones web del Departamento de Señales Digitales?

Para dar respuesta a esta interrogante se define como **objeto de estudio** el proceso de gestión de información asociada a los archivos multimedia; delimitando como **campo de acción:** un componente para la gestión de las tipologías de archivos multimedia.

Se plantea como **idea a defender:** con el desarrollo de un componente para la gestión de tipologías de archivos multimedia sobre tecnologías de desarrollo web se garantiza la correcta manipulación de información asociada a los archivos multimedia en las aplicaciones web.

Se define como **objetivo general**: desarrollar un componente de software que permita gestionar tipologías de forma dinámica para manipular la información asociada a los archivos multimedia en aplicaciones web del Departamento de Señales Digitales.

Para alcanzar dichos objetivos se plantea desarrollar las siguientes **tareas**:

1. Analizar los principales conceptos asociados a la catalogación de archivos multimedia.
2. Seleccionar y argumentar la Metodología de Desarrollo de Software a usar en el proceso.
3. Definir las herramientas y tecnologías a utilizar en la construcción de la solución.
4. Identificar las funcionalidades que debe brindar el componente.
5. Realizar el modelado de los artefactos necesarios para la implementación del componente.
6. Implementar las funcionalidades que permitan definir la estructura de las tipologías.
7. Implementar las funcionalidades para gestionar los datos de las tipologías.

8. Realizar las pruebas al componente.
9. Realizar manual de instalación y uso del componente.

Con el correcto cumplimiento de las tareas se espera obtener los siguientes resultados o aportes.

Posibles resultados:

- Documentación que recoja el resultado de la investigación realizada.
- Documentación UML de los artefactos resultantes del análisis y diseño.
- Componente para la gestión de tipologías de archivos multimedia en aplicaciones web.
- Manual de instalación y uso del componente.

Para el desarrollo completo del trabajo y su total entendimiento se hizo necesario emplear los métodos de investigación teóricos los cuales se mencionan a continuación:

Métodos Teóricos:

Analítico-Sintético: Mediante esté método se analizaron los principales conceptos y estándares relacionados con el proceso de catalogación de multimedia, con el objetivo de sentar las bases teóricas de la investigación y comprender su funcionamiento, para usarlo posteriormente en el desarrollo del componente.

Modelación: Ofrece la posibilidad de crear abstracciones para explicar la realidad. Se hace visible en el trabajo al modelar la representación del funcionamiento del patrón arquitectónico Modelo Vista Controlador.

Histórico-Lógico: Se utilizó en el análisis de la base teórica, el estudio de las tecnologías a emplear y la descripción de los procesos relacionados a la gestión de archivos multimedia.

El trabajo de diploma está compuesto por cuatro capítulos:

Capítulo 1: Fundamentación Teórica: Se describen las tendencias actuales en el ámbito de la catalogación de archivos multimedia a nivel internacional. Se realiza un estudio de la metodología, herramientas y técnicas a utilizar y se justifica la selección de cada una de ellas.

Capítulo 2: Características del Sistema: En este capítulo se definen los requerimientos a implementar. Además, se realiza el diagrama de casos de uso del sistema y la descripción de estos.

Capítulo 3: Análisis y Diseño del Sistema: Se realiza el análisis del sistema a desarrollar con el objetivo de cumplir con todos los requerimientos definidos en el capítulo anterior. Se desarrolla el modelo de

diseño que contiene los diagramas de clases del diseño y los diagrama de interacción. Se describe el diagrama de clases. Se genera el diagrama de entidad relación de la Base de Datos y el Diagrama de Despliegue.

Capítulo 4: Implementación y Pruebas: Trata los aspectos relacionados con la construcción de la solución propuesta donde se genera el diagrama de componentes. Además se realizan pruebas de caja negra con el objetivo de comprobar el correcto cumplimiento de los requisitos funcionales propuestos para el componente.

Capítulo 1: Fundamentación Teórica

Introducción

En el presente capítulo se analizan los diferentes elementos teóricos sobre el proceso de catalogación de archivos multimedia. Se define la metodología de desarrollo de software a utilizar, definiendo las principales características que se tuvieron en cuenta para su selección. También se seleccionan las herramientas y otras tecnologías que se utilizarán a lo largo del proceso de desarrollo de software.

1.1 Catalogación de Archivos multimedia

La catalogación es el proceso de clasificar elementos pertenecientes a un mismo conjunto y se desarrolla pues la gran cantidad de integrantes existentes de dicho conjunto dificultan los procesos de búsqueda y consulta de estos elementos. (Wordreference, 2010)

Para el proceso de catalogación de multimedia a nivel mundial se utilizan los metadatos, que son datos asociados a los archivos multimedia que tienen como función almacenar información de estos archivos y su objetivo es brindar información sobre los datos. Se pueden definir además como “datos sobre datos” (INFLANET, 1998).

Los **metadatos** tienen tres funciones básicas:

- Proporcionar una descripción de un objeto o entidad de información junto con otra información necesaria para su manejo y preservación.
- Suministrar los puntos de acceso a esa descripción por medio de los cuales se generará un índice.
- Codificar la descripción para facilitar su manejo por medios automatizados.

Los metadatos en el caso de los archivos multimedia se pueden presentar de dos formas. La primera es en forma de etiquetas incluidas en el contenedor del archivo y la segunda es almacenados en un archivo independiente del archivo multimedia.

Los metadatos en forma de etiqueta, una de las facilidades más importante que proporcionan es que permite realizar búsquedas rápidas y sencillas de aquellos archivos que se desean. Las búsquedas se realizan generalmente utilizando varios criterios como pueden ser autor, palabras claves o copyright. Estas búsquedas se pueden realizar cuando se tienen grandes colecciones de archivos multimedia de forma local. Algunos de los estándares de este tipo de metadatos son: ID3, Exif, XMP y TIFF.

ID3: Es el más popular de los estándares de metadatos en forma de etiquetas. Es utilizado principalmente en archivos de audio como son MP3, OGG y WMA. Permite almacenar etiquetas de uso común por ejemplo autor, título, álbum y palabras claves. También permite almacenar información gráfica como puede ser la portada del álbum (ONeill, 2010).

Extensible Metadata Platform (XMP): es el estándar de metadatos que utilizan las aplicaciones de Adobe. Permite crear metadatos para formatos contenedores de multimedia comunes como son: FLV, QuickTime (MOV), Vídeo para Windows (AVI), Windows Media (ASF, WAV) y MPEG (MP3, MPEG-2, MPEG-4). Es escrito utilizando el lenguaje de marcas extensible XML. Permite añadir metadatos con propiedades como la ubicación, nombre del autor y copyright. El estándar es, como su nombre lo indica, diseñado para ser extendido, permitiendo a los usuarios incorporar sus propios tipos de metadatos personalizados dentro de los datos de XMP (Adobe, 2010).

Exchangeable image file format (Exif): es un estándar para el intercambio de información en el almacenamiento de archivos de imagen, especialmente los que utilizan las compresiones JPEG y TIFF. La mayoría de las cámaras digitales utilizan el formato Exif. Las principales etiquetas definidas por el formato son: fecha y hora, localización, descripción e información sobre copyright (Exif, 2008).

En general este tipo de metadatos facilita el trabajo cuando se tienen los archivos multimedia en un disco local, aunque su uso para el trabajo con servidores se hace costoso al acceder a la información contenida dentro de un archivo multimedia. Además otras de las deficiencias que presentan es que son muy diferentes los campos que se almacenan en dependencia del tipo de archivo multimedia, lo cual puede complejizar procesos de búsqueda y gestión en una aplicación que se dedique a la manipulación de una amplia gama de multimedia, por ejemplo de imágenes, video y sonido.

El otro tipo de metadatos es el que se almacena independiente del archivo multimedia, el cual es el que ha alcanzado mayor relevancia en los últimos años, pues permiten un acceso más amplio y rápido a la información almacenada de las medias. Además no es necesario tener el archivo multimedia para realizar los procesos de presentación de la información y posibilita que siguiendo un estándar universal se almacenen los metadatos de archivos multimedia de forma uniforme, sin importar el tipo que sea. Actualmente existen varios estándares de metadatos de este tipo en el ámbito del video y sonido digital, entre estos se destacan por el uso y facilidades que brindan el **MPG-7** y el **Doblin Core.**

1.1.1 MPEG-7

MPEG-7: también llamado "Interfaz de descripción de contenidos multimedia", se creó en el año 2001 para estandarizar la extracción de características basadas en el contenido de los diferentes tipos de información multimedia. Proporciona una serie de herramientas para describir contenido multimedia. Estas herramientas de descripción (metadatos, descriptores, esquemas de descriptores) sirven para crear descripciones que serán la base para aplicaciones que permitan el acceso a contenido de este tipo.
El estándar MPEG-7 está formado por los siguientes componentes (Ballesteros, 2006):

1. **MPEG-7 Systems:** Describe las herramientas necesarias para desarrollar descripciones MPEG-7 para un eficiente transporte, almacenamiento y arquitectura del terminal.
2. **MPEG-7 Description, Definition, Language:** Lenguaje que especifica los esquemas de descripción permitiendo la extensión y modificación de los existentes. Es un lenguaje basado en XML.
3. **MPEG-7 Visual:** Conjunto de herramientas de descripción para descripciones visuales.
4. **MPEG-7 Audio:** Conjunto de herramientas de descripción para descripciones de audio.
5. **MPEG-7 Multimedia, Description, Schemes:** Conjunto de herramientas de descripción con características genéricas y descripciones multimedia.
6. **MPEG-7 Reference Software:** Implementación de las partes relevantes de software del estándar MPEG-7 en estado normativo.
7. **MPEG-7 Conformance Testing:** Normas y procedimientos para la aprobación de las implementaciones de MPEG-7.
8. **MPEG-7 Extraction and use of descriptions:** Material informativo (en forma de informe técnico) sobre la extracción y el uso de algunas herramientas de descripción.
9. **MPEG-7 Profiles and levels:** Conjunto de normas y perfiles.
10. **MPEG-7 Schema Definition:** Especificación del esquema utilizado en la DDL (Description Definition Language).

MPEG-7 define una serie de **descriptores** que permiten analizar y caracterizar el contenido de fuentes audiovisuales para su posterior indexación, búsqueda o comparación.

Estos descriptores son la parte más importante del estándar pues son lo que define lo que se debe almacenar de cada característica de un video. A continuación se presenta una figura que recoge las principales características de cada uno de los descriptores que propone dicho estándar:

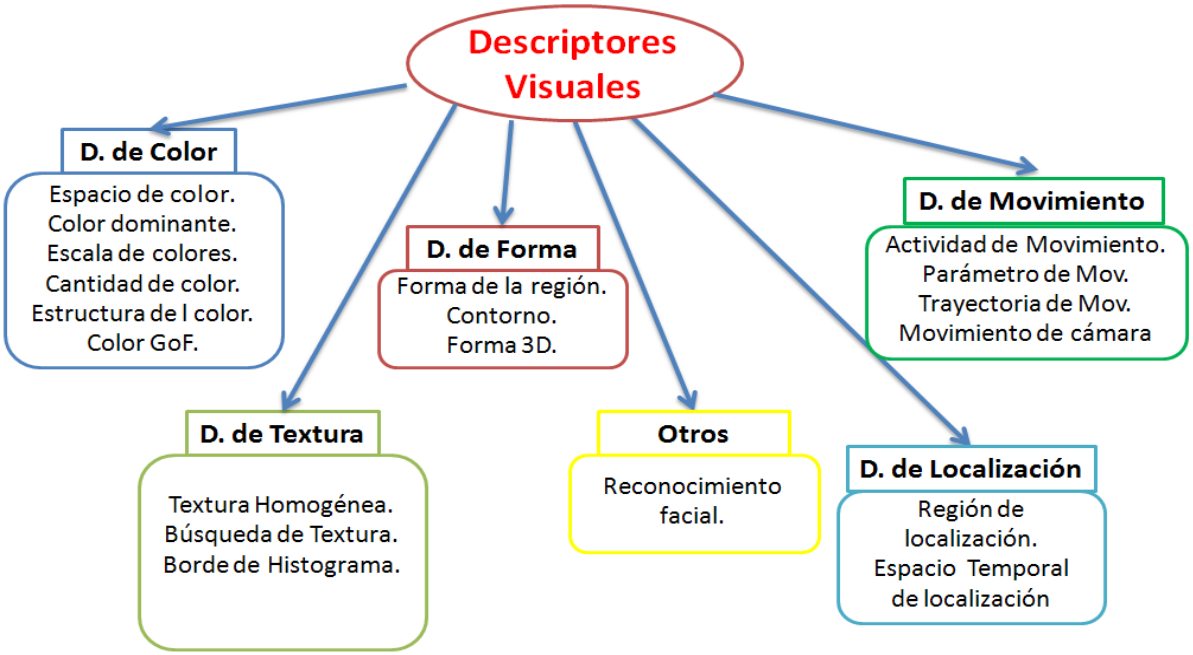


Figura 1 Descriptores visuales del estándar MPG-7

1.1.2 Dublin Core:

Dublin Core: es un modelo de metadatos elaborado y auspiciado por la DCMI (Dublin Core Metadata Initiative), una organización dedicada a fomentar la adopción extensa de los estándares interoperables de los metadatos y a promover el desarrollo de los vocabularios especializados de metadatos para permitir sistemas más inteligentes en el descubrimiento de recursos.

Es un sistema de 15 definiciones semánticas descriptivas, que pretenden transmitir un significado semántico a las mismas. Los implementadores puedes seleccionar según las necesidades que tenga en las aplicaciones que se están desarrollado cuales de estas definiciones utilizar. Cada elemento es opcional y puede repetirse. Además, los elementos pueden aparecer en cualquier orden. Otra característica importante es que las implementaciones del estándar se pueden hacer XML.

Las 15 definiciones propuestas por Dublin Core se clasifican en 3 grupos según las características que se almacenan (Masa, 2008):

1. **Contenido:** se definen características generales del archivo multimedia (título, materia,

descripción, etc.).

2. **Propiedad Intelectual:** se definen características relativas a la propiedad intelectual (creador, editor, derechos de autor).

3. **Características Técnicas:** se definen propiedades específicas del material audiovisual como son: formato, lenguaje y fecha de producción.

Dentro de cada clasificación se encuentran los siguientes elementos (Masa, 2008):

1. **Título**

Etiqueta: Title. El nombre dado a un recurso, usualmente por el autor.

2. **Autor o Creador**

Etiqueta: Creator. La persona u organización responsable de la creación del contenido intelectual del recurso. Por ejemplo, los autores en el caso de documentos escritos, artistas, fotógrafos e ilustradores en el caso de recursos visuales.

3. **Palabras Claves**

Etiqueta: Subject. Los tópicos del recurso. Típicamente, Subject expresará las palabras claves o frases que describen el título o el contenido del recurso. Se fomentará el uso de vocabularios controlados y de sistemas de clasificación formales.

4. **Descripción**

Etiqueta: Description. Una descripción textual del recurso, tal como un resumen en el caso de un documento o una descripción del contenido en el caso de un documento visual.

5. **Editor**

Etiqueta: Publisher. La entidad responsable de hacer que el recurso se encuentre disponible en la red en el formato actual, por ejemplo la empresa editora, un departamento universitario u otro tipo de organización.

6. **Colaboradores**

Etiqueta: Contributor. Una persona u organización que haya tenido una contribución intelectual significativa en la creación del recurso pero cuyas contribuciones son secundarias en comparación a las de las personas u organizaciones que crearon el material (por ejemplo, editor, ilustrador y traductor).

7. **Fecha**

Etiqueta: Date. Fecha en la que el recurso se puso a disposición del usuario en su forma actual. Se recomienda la utilización de uno de los formatos basado en la norma ISO 8601 que incluye, entre otras,

fechas en el formato AAAA-MM-DD. De esta forma la fecha 1994-11-05 correspondería al 5 de noviembre de 1994.

8. **Tipo del Recurso**

Etiqueta: Type. La categoría del recurso, por ejemplo conferencia, película, video musical, serie. Para asegurar la interoperabilidad, Type debería ser seleccionado de una lista de valores definida previamente por el grupo de trabajo.

9. **Formato**

Etiqueta: Format. El formato de datos de un recurso, usado para identificar el software y posiblemente, el hardware que se necesitaría para mostrar el recurso. Para asegurar la interoperabilidad, los valores de Format deberían ser seleccionados de entre una lista de valores definida previamente por el grupo de trabajo.

10. **Identificador del Recurso**

Etiqueta: Identifier. Secuencia de caracteres usados para identificar unívocamente un recurso. Ejemplos para recursos en línea pueden ser URLs. Para otros recursos pueden ser usados otros formatos de identificadores, como por ejemplo ISBN.

11. **Fuente**

Etiqueta: Source. Secuencia de caracteres utilizado para identificar unívocamente un trabajo a partir del cual proviene el recurso actual.

12. **Lengua**

Etiqueta: Language. Lenguajes del contenido intelectual del recurso. Prácticamente el contenido de este campo debería coincidir con los de la RFC 1766, por ejemplo: en, es, de, fi, ja y zh.

13. **Relación**

Etiqueta: Relation. Un identificador de un segundo recurso y su relación con el recurso actual. Este elemento permite enlazar los recursos relacionados y las descripciones de los recursos.

14. **Cobertura**

Etiqueta: Coverage. La característica de cobertura espacial y/o temporal del contenido intelectual del recurso.

15. **Derechos**

Etiqueta: Rights. Una referencia (URL, por ejemplo) para una nota sobre derechos de autor, para un servicio de gestión de derechos o para un servicio que dará información sobre términos y condiciones de

acceso a un recurso.

1.1.3 Herramientas para la gestión de metadatos de archivos multimedia.

Para la gestión de metadatos de archivos multimedia existen varias herramientas implementadas a nivel internacional, las cuales generalmente están basadas en los estándares de metadatos para archivos multimedia estudiados anteriormente. Estas herramientas muchas veces son páginas web en las cuales es necesario insertar los datos del archivo multimedia y esta genera el metadato en formato XHTML o XML. Algunas de las aplicaciones web que brindan esta funcionalidad son:

Stanford Online Accessibility Program: es una aplicación desarrollada por la Universidad de Stanford con el objetivo de ayudar a los creadores y diseñadores de contenidos en línea en la producción de material que sea accesible al mayor público posible. Implementa el estándar de metadatos Dublin Core de forma estática, sin dar la posibilidad de añadirle nuevos campos en dependencia de las necesidades de los clientes. Se puede acceder a través de la dirección **http://soap.stanford.edu/plugins/dublincore/.**

WebArchiv Dublin Core Metadata Creator: es una aplicación web desarrollada por la Biblioteca Nacional de la República Checa con el objetivo de hacer accesible la gran cantidad de recursos electrónicos que solo se encuentran publicados en Internet. Se encarga de hacer accesible la información de forma estándar y de forma que pueda ser útil para futuras generaciones. Implementa el estándar Dublin Core para la generación de metadatos de forma configurable pues se pueden adaptar agregándole o eliminado los campos que no se necesiten. Permite exportar en formatos XHTML y RDF y se puede acceder a través de la dirección**: http://www.webarchiv.cz/generator/dc_generator.php.**

Estas aplicaciones web facilitan los procesos de generación del XML del estándar Dublin Core pero no son útiles en procesos de gestión y almacenamiento de los metadatos, además presentan como desventajas para el Departamento de Señales Digitales que es necesario estar conectados a Internet para generar los metadatos.

También existen software para la creación de metadatos, a continuación se presentan algunos de estos:

Proceedings Paper: es una herramienta semiautomática que permite la gestión del estándar de metadato MPEG-7. Permite hacer uso de los metadatos extraídos como puede ser realizar búsquedas por color o palabras claves. Además implementa algunas funcionalidades de forma automática como son la detección de textos en el video o la detección de la región. Presenta como desventaja que es privativo y hay que pagar por la licencia.

MPEG-7 XM: software que permite el desarrollo de aplicaciones basadas en MPEG-7. Implementado en C++ como lenguaje de programación. Permite la extracción automática de características de los videos. Fue implementado para sistemas operativos basados GNU-Linux y no es soportado por Windows. Utiliza un servidor de base de datos específico para el almacenamiento de los metadatos: XMServer.

Ambas aplicaciones son de escritorio e implementan el estándar MPEG-7 pero no se ajustan a las necesidades del departamento pues lo que el departamento necesita es implementar una aplicación web que gestione la información de los archivos multimedia.

En el ámbito del Departamento de Señales Digitales existe una implementación de un módulo para el Sistema Gestor de Contenidos Drupal que realiza la gestión de tipologías de archivos multimedia pero tiene como desventajas que no se basa en ninguno de los estándares de metadatos mencionados anteriormente y está implementado para Drupal, lo que no permite hacer la gestión de forma genérica en cualquier tipo de aplicaciones web.

1.1.4 Estándares de metadatos, tipologías de archivos multimedia y su gestión en aplicaciones web.

Los estándares de metadatos son modelos que se siguen para almacenar información de los archivos multimedia, pero estos en ocasiones se alejan o no se ajustan de las necesidades de un cliente determinado; debido a esto muchos estándares de metadatos como el Dublin Core permiten cierta flexibilidad a la hora de seleccionar cuales campos de los que él define utilizar y en muchos caso incluso dan la posibilidad de incluir otros campos que no están definidos por el estándar. Esto se debe a que algunas veces se hace necesario guardar información de los videos que no es relevante para el cliente que la está gestionado y en otras se queda por debajo de las necesidades de este. Lo anteriormente planteado está muy relacionado con la tipología del archivo multimedia, pues no se necesita guardar la misma información (metadatos) de un video musical que de una conferencia científica, los objetivos de ambos archivos multimedia no son los mismos, por lo que la información que se debe almacenar debe estar en correspondencia con su futura aplicación y con los intereses que se quiera alcanzar con el uso de dicho archivo multimedia.

Lo anteriormente planteado unido al uso extendido de las aplicaciones web para la gestión y publicación de archivos multimedia en la red, crean la necesidad de desarrollar un componente para la gestión de tipologías en aplicaciones web, de forma tal que se permita definir los metadatos que se van a almacenar

en dependencia de las necesidades del cliente y basados en los estándares de metadatos mencionados anteriormente.

Para lograr su correcto desarrollo y alcanzar los resultados esperados utilizando las mejores prácticas a nivel internacional en el desarrollo de software, una de las tareas fundamentales es definir una metodología de desarrollo que guíe durante el desarrollo del componente.

1.2 Metodología de Desarrollo de Software.

Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software.

Es como un libro de recetas de cocina, en el que se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. (UMurcia, 2006)

No existe una metodología de software universal. Las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigen que el proceso sea configurable.

Dentro de la ingeniería de software existen algunas metodologías que se encargan de elaborar estrategias de desarrollo de software que promueven nuevas prácticas para un buen desarrollo del mismo, entre las más conocidas se encuentran: Rational Unified Process (RUP), eXtreme Programming (XP) y Microsoft Solution Framework (MSF). En el desarrollo del componente se utilizará RUP pues es la metodología que está definida por el Departamento de Señales Digitales por las ventajas que proporciona para el desarrollo de software de forma segura.

1.2.1 RUP.

RUP (Proceso Unificado de Desarrollo): es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. La versión que se ha estandarizado vio la luz en 1998 y se conoció en sus inicios como Proceso Unificado de Rational 5.0; de ahí las siglas con las que se identifica a este proceso de desarrollo (Jacaboson, 2000).

Un proceso de desarrollo de software define quién hace qué, cómo y cuándo. RUP define cuatro elementos: trabajadores (roles) que responden a la pregunta ¿Quién?, las actividades que responden a la pregunta ¿Cómo?, los artefactos (productos), que responden a la pregunta ¿Qué? y los flujos de trabajo

de las disciplinas que responde a la pregunta ¿Cuándo?

Trabajadores (quién): Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.

Actividades (cómo): Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.

Artefactos (qué): Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

Flujo de actividades (cuándo): Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de soporte. En la Figura 2 se representa el proceso en el que se grafican los flujos de trabajo y las fases y se muestra la dinámica expresada en iteraciones y puntos de control.

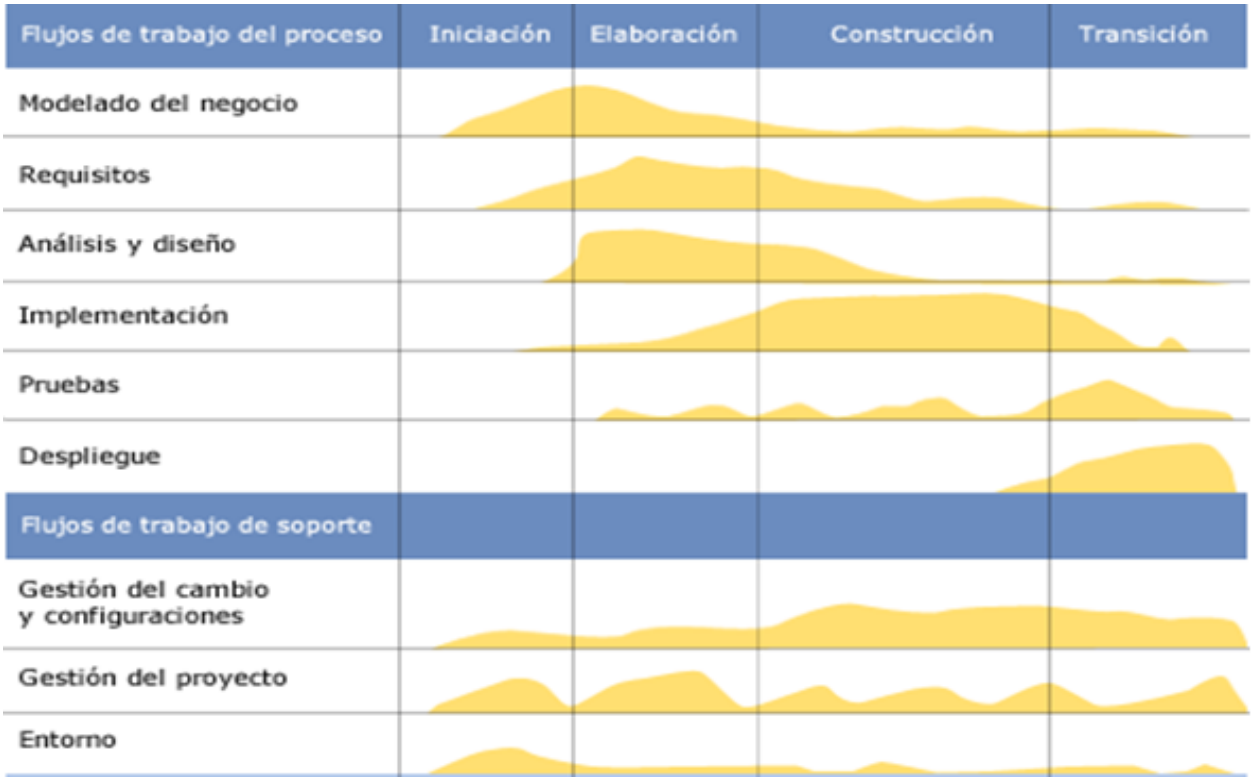


Figura 2 Gráfico Bidimensional de RUP.

Flujos de trabajo de RUP (Jacaboson, 2000)**:**

1. **Modelado de negocio:** Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
2. **Requerimientos o Requisitos:** Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
3. **Análisis y diseño:** Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
4. **Implementación:** Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
5. **Prueba (Testeo):** Busca los defectos a lo largo del ciclo de vida del software.
6. **Despliegue:** Produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
7. **Gestión del cambio y configuración:** Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
8. **Gestión del proyecto:** Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
9. **Entorno:** Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Fases de RUP (Jacaboson, 2000)**:**

Conceptualización (Concepción o Inicio): Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.

Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.

Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene 1 o varios release del producto que han pasado las pruebas. Se ponen estos release a consideración de un subconjunto de usuarios.

Transición: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Principales características de RUP (Jacaboson, 2000)**:**

1. **Dirigido por casos de uso:** los casos de uso guían los procesos de diseño, implementación y prueba. Estos constituyen un elemento integrador y una guía del trabajo. Además proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.
2. **Centrado en la arquitectura:** La arquitectura de un sistema es la organización o estructura de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.
3. **Iterativo e incremental:** RUP divide el trabajo en partes más pequeñas o mini proyectos. Permitiendo que el equilibrio entre casos de uso y arquitectura se vaya logrando durante cada mini proyecto, así durante todo el proceso de desarrollo. Cada mini proyecto se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto.

En el desarrollo del proyecto con RUP para poder realizar y diseñar los principales artefactos generados durante el proceso de desarrollo de software se utilizará como lenguaje de representación visual UML.

1.2.2 UML.

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos que se crean durante el proceso de desarrollo. UML no es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso. UML es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos. (Jacaboson, 2000)

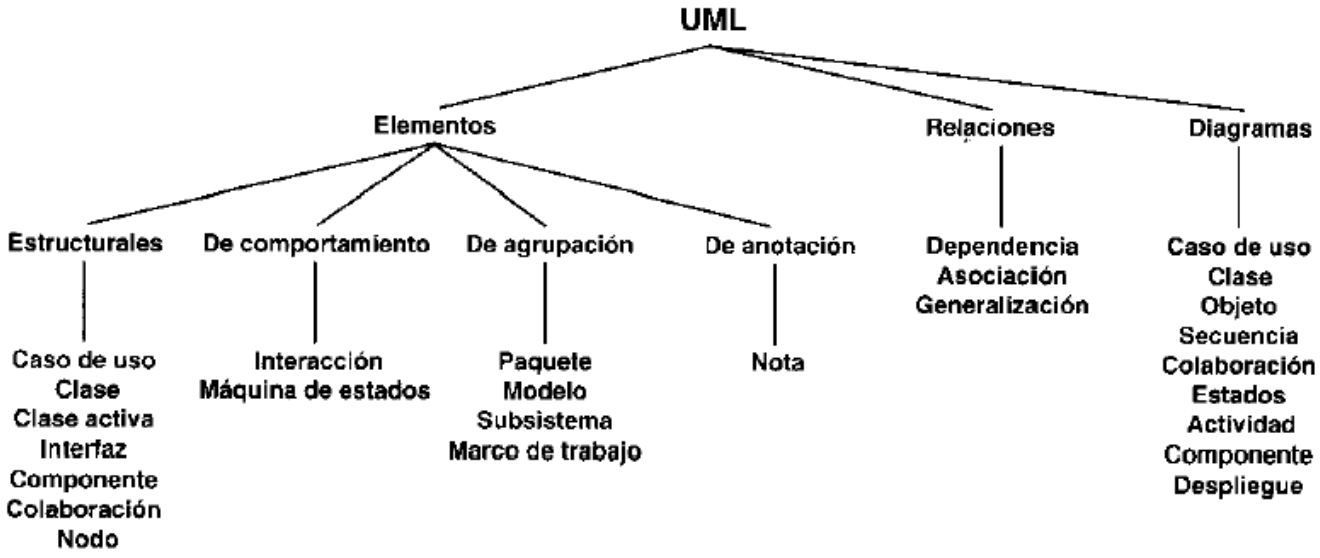


Figura 3 Vocabulario UML.

1.2.3 Visual Paradigm 6.4.

Se seleccionó Visual Paradigm como herramienta para el modelado UML pues esta herramienta desarrollada para diseñar software con programación orientada a objetos y busca reducir la duración del ciclo de desarrollo brindando ayuda tanto a arquitectos, analistas, diseñadores y desarrolladores.

Es una herramienta colaborativa que soporta a varios usuarios trabajando en un mismo proyecto. Genera la documentación del proyecto automáticamente en varios formatos como son HTML o PDF, y permite control de versiones. Brinda la posibilidad de generar código a partir de los diagramas, para plataformas como .Net, Java y PHP, así como obtener diagramas a partir del código.

Presenta licencia gratuita y comercial. Es fácil de instalar y actualizar, y compatible entre ediciones. Por lo que se puede concluir que Visual Paradigm se caracteriza por su robustez, usabilidad y portabilidad, por lo que se decidió adoptar como herramienta CASE para realizar el modelado de todo el componente.

1.3 Lenguajes de programación.

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que éste pueda comunicarse con los dispositivos hardware y software existentes. (Definición, 2008).

1.3.1 PHP.

PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje de programación interpretado ampliamente usado, diseñado especialmente para desarrollo web y que puede ser incrustado dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. A esto se añaden valores como el hecho de ser un proyecto de código abierto, gratuito y multiplataforma. (Skilar, 2005).

Es un lenguaje de programación rápido y soportado por la mayoría de servidores Web incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, entre otros.

Ventajas:

1. Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
2. Posee una amplia documentación en su página oficial.
3. Permite las técnicas de Programación Orientada a Objetos (POO).
4. Contiene bibliotecas nativas de funciones sumamente amplias.
5. No requiere definición de tipos de variables, aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
6. Tiene manejo de excepciones (desde PHP5).

1.3.2 HTML

HTML: Hyper Text Markup Language (Lenguaje de Marcado de Hipertexto), es un sistema de etiquetas utilizado normalmente en la www (World Wide Web). Lo que se observa al visualizar una página en internet no es más que la interpretación que hace el navegador del código HTML. Este se genera por el servidor web al interpretar un código generado en PHP.

Ventajas: Sencillo de comprender, presenta un texto estructurado de forma agradable, no se necesita grandes conocimientos cuando se cuenta de un editor de páginas web, contiene archivos pequeños, despliegue rápido y lo admiten todos los navegadores.

Desventajas: Es un lenguaje estático, la interpretación de cada navegador puede ser diferente, guarda

muchas etiquetas que pueden convertirse en basura que dificulta la corrección, el diseño es más lento y las etiquetas son muy limitadas.

1.3.3 XHTML1.0

XHTML significa Lenguaje extensible de marcación de hipertexto (Extensible Hypertext Markup Language). Este lenguaje ofrece la norma clásica para crear páginas web, el Lenguaje de Marcación de Hipertexto (HTML), y la nueva norma para datos descriptivos, el Lenguaje de Marcación Extensible (XML, Extensible Markup Language) (Valentine Chelsea, 2003).

XHTML sigue el conjunto de reglas impuestas por XML; por lo tanto, desde el punto de vista sintáctico, se necesita ser más cuidadoso y más correcto cuando se construyan páginas web con XHTML que cuando se hagan con HTML.

1.3.4 XML

XML Extensible Markup Language (Lenguaje de Marcas Extensible): es un metalenguaje compuesto por un grupo de reglas y convenciones sintácticas que se pueden utilizar para construir nuestros propios grupos de elementos de marcación.

XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Este está derivado de SGML (ISO 8879). XML también desempeña un papel cada vez más importante en el intercambio de una amplia variedad de datos en la Web como son los RSS o los metadatos de un archivo (W3C, 2010).

1.3.5 Javascript

JavaScript es un lenguaje script usado para incluir código en el cliente que será interpretado por el navegador. Este tipo de lenguaje es interpretado, no compilado. Es el lenguaje de programación Web del lado del cliente más extendido. Otras características importantes es que manejado por eventos e independiente de la plataforma.

Ventajas: Es un lenguaje de scripting seguro y fiable, los script tienen capacidad limitada por razones de seguridad y el código se ejecuta en el cliente.

Desventaja: Presenta un código visible para cualquier usuario y el código debe descargarse completamente.

1.4 Entorno de Desarrollo Integrado (IDE)

IDE (Entorno Integrado de Desarrollo en inglés Integrated Development Enviroment): Son programas que brindan una serie de facilidades a los programadores en el proceso de desarrollo de software. La mayoría de los IDE permiten editar, compilar y depurar código, así como crear interfaces gráficas. Algunos IDE soportan múltiples lenguajes de programación, tales como Eclipse y Netbeans. Estos pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes.

1.4.1 Netbeans 6.9

Netbeans es un IDE de código abierto y libre utilizado para desarrollar aplicaciones de escritorio, web y para móviles. Permite el desarrollo de aplicaciones en varios lenguajes como son Java, PHP, Ruby y C++. Es un producto gratuito sin restricciones de uso, presenta un excelente completamiento de código. La programación mediante NetBeans se realiza a través de componentes de software modulares, los cuales están a disposición del usuario, en su página oficial, para conseguir mejoras en las aplicaciones.

1.5 Sistema Gestor de Bases de Datos

Un Sistema Gestor de Bases de Datos (SGBD) o DBMA (DataBase Management System) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos **(Avila, 2008)**.

Algunos ejemplos de SGBD son Oracle, DB2, PostgreSQL, MySQL, MS SQL Server, etc.

Un SGBD debe permitir:

• Definir una base de datos: especificar tipos, estructuras y restricciones de datos.

• Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.

• Manipular la base de datos: realizar consultas, actualizarla, generar informes.

1.5.1 PostgreSQL 8.4

Por políticas de desarrollo del Departamento de Señales Digitales el Sistema Gestor de Bases de Datos que se utiliza es PostgreSQL, por eso se caracterizará para mostrar las principales características y

ventajas que trae para el desarrollo del componente.

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (Berkeley Software Distribution) y con su código fuente disponible libremente. Permite una fácil gestión de los usuarios y de las bases de datos que contenga el sistema. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (Martínez, 2010).

Contiene APIs para programar en C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, PHP, Lisp, Scheme, Qt y muchos otros. Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.

El desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores con la que se puede contar para obtener soporte ante cualquier problema presentado con el uso del sistema gestor de base de datos.

Algunos datos interesantes sobre los límites que tiene PostgreSQL como gestor de base datos son los siguientes:

Límite	Valor
Máximo tamaño base de dato	Ilimitado (Depende del sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo de la versión)
Máximo número de índices por tabla	Ilimitado

Figura 4 Tabla Límites de PostgreSQL.

1.6 Servidor Web Apache 2.2

El Servidor Web Apache permite interpretar PHP, el lenguaje de programación en el que se va a implementar el componente y responder al cliente que hizo la solicitud en lenguaje HTML. Este intercambio de información entre el navegador y el servidor se realiza mediante el protocolo HTTP. El servidor web recibe las peticiones de los clientes y pone en funcionamiento una serie de herramientas para darle respuesta a la solicitud.

Apache es un servidor flexible, rápido y eficiente. Está desarrollado bajo licencia BSD lo cual permite hacer modificaciones en el código siempre que se reconozca el trabajo de la comunidad. Es el servidor web más utilizado en Internet.

Algunas de las características por las que este servidor logra la popularidad son (Cira Orta, 2006):

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Es una tecnología gratuita de código fuente abierta.
- Permite la creación de sitios web dinámicos en varios lenguajes como son: Perl, PHP, Python y otros lenguajes.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor dando la posibilidad de ejecutar un determinado script cuando ocurra un error en concreto.

1.7 Conclusiones

Después de realizar un análisis de los principales conceptos relacionados al proceso de catalogación de media se puede concluir que uno de los conceptos más importantes relacionados con este proceso son los metadatos. Se realizó un estudio de los diferentes tipos de metadatos existentes, así como los estándares más utilizados a nivel internacional y se determinó que dada las características el que más se ajusta a las necesidades del Departamento de Señales Digitales es el Dublin Core.

Como metodología para el desarrollo se propone RUP, pues esta se centra en obtener al final del ciclo de vida del proyecto un producto documentado e implementado con calidad, y para el modelado UML de los artefactos y elementos generados durante el proceso de desarrollo se utilizará Visual Paradigm 6.4.

Se utilizará PHP como lenguaje de programación pues en este es un potente lenguaje de programación orientado a objeto para el desarrollo de aplicaciones web.

El IDE de desarrollo será Netbeans en su versión 6.9. Es multiplataforma, gratuito y tiene un buen completamiento de código. Como servidor web se empleará Apache 2.0 el más utilizado a nivel mundial y Sistema Gestor de Bases de Datos a utilizar será PostgreSQL pues es el uno de las más potentes, robustos y seguros y el que se utiliza en los proyectos desarrollados en el Departamento de Señales Digitales.

Capítulo 2: Características del Sistema

2.1 Introducción

En el presente capítulo se desarrolla una investigación del entorno donde se implementará el componente con el objetivo de definir las condiciones o capacidades que debe brindar el mismo. Se presenta el Modelo de Dominio para comprender el contexto donde se implementará el sistema. Además se definen los requisitos funcionales y no funcionales, el diagrama de casos de uso del sistema y se describen los casos de uso del sistema.

2.2 Modelo de Dominio

Un modelo de dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema, teniendo como ventaja el permitir ayudar a los usuarios, clientes y desarrolladores a utilizar un vocabulario común para poder entender el contexto en que se emplaza el sistema.

Para comprender el dominio donde se va a implementar el sistema se hizo un estudio de varios proyectos del Departamento de Señales Digitales donde fueron identificados los siguientes elementos pertenecientes al dominio de los proceso de gestión de tipologías de archivos multimedia:

Aplicación: Cualquier aplicación web del Departamento de Señales Digitales que necesite administrar tipologías y la información relativa a los archivos multimedia.

Archivos Multimedia: Fichero multimedia de audio o video que será publicado en la plataforma y posteriormente reproducido por los usuarios. De los archivos multimedia es de los cuales se almacenan los diferentes metadatos.

Metadatos: Conjunto de datos asociados a los archivos multimedia en dependencia de la tipología de archivo multimedia a la que pertenezcan. Estos datos se almacenan en la base de datos para facilitar procesos como publicaciones y búsquedas de archivos de archivos multimedia.

Tipologías de Archivos Multimedia: Categoría en la que se clasifican los archivos multimedia en dependencia de su tipo y de la información que se desee almacenar de este.

Bases de Datos: Entidad en la cual se almacenan de manera estructurada y con la menor redundancia posible los datos referentes a los archivos multimedia y los usuarios del sistema.

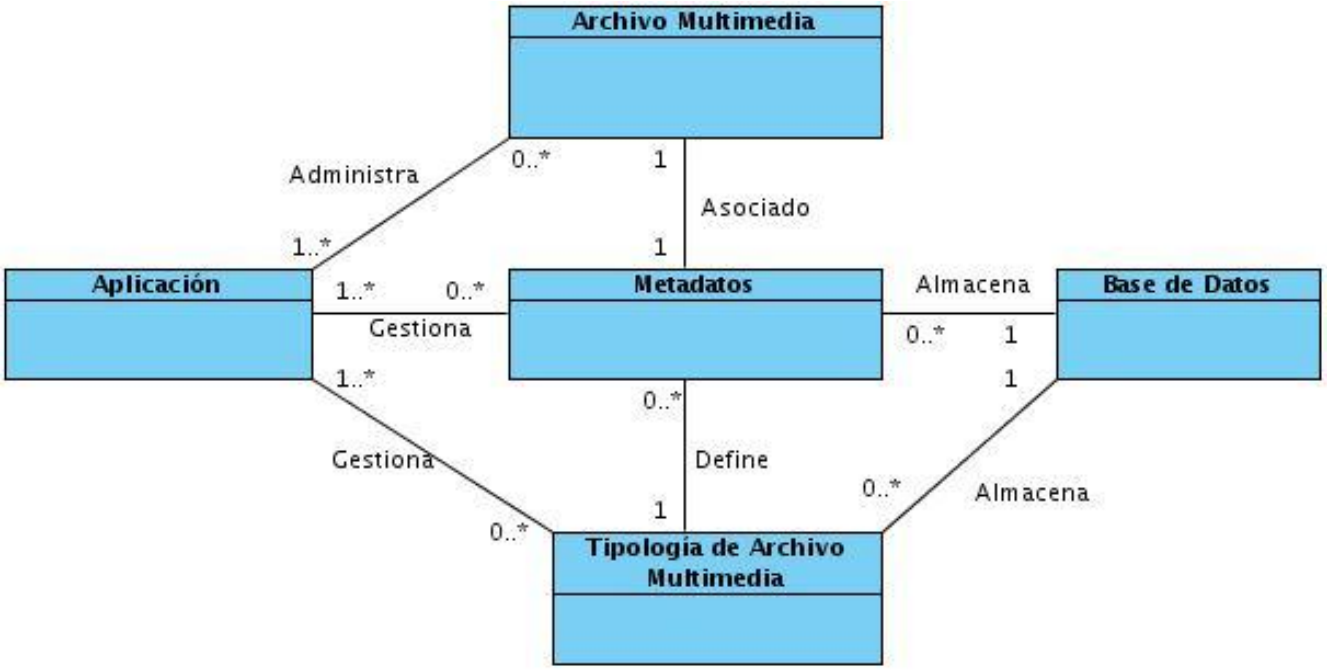


Figura 5 Modelo de Dominio.

2.3 Especificación de los Requisitos del Sistema

2.3.1 Requisitos Funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir.

Para el componente propuesto se definen los siguientes requisitos funcionales (RF):

RF 1: Adicionar tipología de AM.

Descripción: El componente debe permitir adicionar una nueva tipología de archivo multimedia y definir los datos que va a incluir dicha tipología. La tipología adicionada al sistema no se podrá utilizar hasta que no se cree físicamente en la base de datos.

RF 2: Adicionar campos a una tipología de AM existente.

Descripción: El componente debe permitir adicionar campos a las tipologías existentes. Estos campos serán usados para almacenar los datos asociados a los archivos multimedia.

RF 3: Editar campo a una tipología de AM existente.

Descripción: El componente debe permitir editar los valores de los campos adicionados a las tipologías.

RF 4: Eliminar campo a una tipología de AM existente.

Descripción: El componente debe permitir eliminar campos a las tipologías existentes.

RF 5: Adicionar relación entre tipología de AM.

Descripción: El componente debe permitir adicionar relaciones a las tipologías existentes con otras tipologías ya creadas. Estas relaciones permitirán vincular los datos de varias tipologías para aumentar la flexibilidad de los datos.

RF 6: Editar relación entre tipologías de AM.

Descripción: El componente debe permitir editar las relaciones existentes entre las tipologías.

RF 7: Eliminar relación entre tipologías AM.

Descripción: El componente debe permitir eliminar relaciones a las tipologías existentes.

RF 8: Editar tipología de archivo multimedia.

Descripción: El componente debe permitir modificar los datos de una tipología de archivo multimedia.

RF 9: Eliminar tipología de archivo multimedia.

Descripción: El componente debe permitir eliminar una tipología de archivo multimedia.

RF 10: Crear tipología previamente adicionada.

Descripción: El componente debe permitir crear físicamente la tabla de la tipología en la base de datos a partir de los campos y las relaciones adicionados previamente, la tipología queda disponible para ser utilizada a la hora de adicionar datos asociados a los archivos multimedia.

RF 11: Adicionar datos asociados a un archivo multimedia.

Descripción: El componente debe permitir adicionar los datos asociados a un archivo multimedia según la tipología a la que pertenezca dicho archivo multimedia.

RF 12: Editar datos asociados a un archivo multimedia

Descripción: El componente debe permitir modificar los datos asociados a un archivo multimedia.

RF 13: Eliminar datos asociados a un archivo multimedia

Descripción: El componente debe permitir eliminar los datos asociados a un archivo multimedia.

RF 14: Obtener información de tipologías de archivos multimedia

Descripción: El componente debe permitir obtener información de las tipologías de archivos multimedia existentes.

RF 15: Obtener información de relaciones entre tipologías de archivos multimedia

Descripción: El componente debe permitir obtener información relativa a las relaciones existentes entre tipologías de archivos multimedia.

RF 16: Obtener información de los campos de tipologías de archivos multimedia

Descripción: El componente debe permitir obtener información relativa a los campos de tipologías de archivos multimedia existentes.

RF 17: Obtener los datos asociados a las tipologías de archivos multimedia

Descripción: El componente debe permitir obtener los datos asociados a las tipologías de archivos multimedia existentes.

2.3.2 Requisitos No Funcionales

El producto final de entrega debe tener características y cualidades específicas las cuales se identifican como requerimientos no funcionales. En el desarrollo del componente se identificaron los siguientes requisitos no funcionales:

- **Usabilidad**

El componente podrá ser usado por cualquier persona que tenga conocimientos en programación web en lenguaje PHP y en base de datos objeto-relacionales.

- **Rendimiento**

El componente para lograr un buen rendimiento debe tener un rápido procesamiento de los datos, dando una respuesta rápida que nunca debe exceder los 5 segundos.

- **Soporte**

El componente contará con una ayuda donde el usuario podrá suplir las dudas que se le puedan presentar durante la utilización del mismo. El usuario del componente deberá recibir un adiestramiento previo en la utilización del sistema con el fin de que pueda explotar las prestaciones del sistema sin contratiempos ocasionados por la falta de preparación técnica.

- **Software**

La aplicación debe correr en sistemas operativos Windows, Unix y Linux. Deberá disponer de un navegador web, estos pueden ser IE 7, Opera 9, Google Chrome 1 y Firefox 2 o versiones superiores de estos.

- **Hardware**

Máquina Cliente:

1. Procesador Pentium 4 o superior.
2. Mínimo 256 Mb de RAM.

Máquina servidor:

1. Procesador Pentium 4 o superior.
2. Mínimo 1 Giga de RAM.

2.4 Modelo de Casos de Uso del Sistema

Los casos de uso se emplean para capturar el comportamiento deseado del sistema en desarrollo, sin tener que especificar cómo se implementa ese comportamiento. Proporcionan un medio para que lo desarrolladores, los usuarios finales del sistema y los expertos del dominio lleguen a una comprensión común del sistema.

2.4.1 Diagrama de Casos de Uso del Sistema

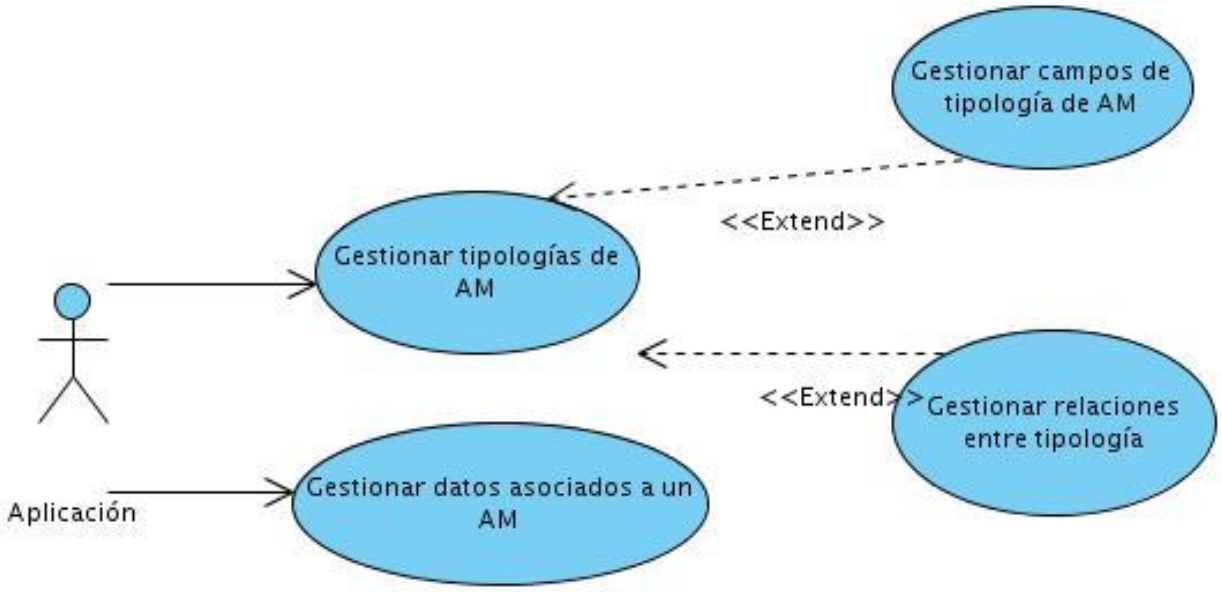


Figura 6 Diagrama de Casos de Uso del Sistema.

2.4.2 Descripción de los Casos de Uso del Sistema:

Caso de Uso: Gestionar Tipología de Archivos Multimedia.

Caso de Uso:	Gestionar Tipología de Archivos Multimedia
Actores:	Aplicación
Resumen:	El caso de uso se inicia cuando desde la aplicación se desea adicionar una nueva tipología de archivos multimedia, modificar los datos de una existente o eliminarla. Termina con la creación, modificación o eliminación de una tipología de archivo multimedia o con un mensaje de proceso fallido.
Precondiciones:	
Referencias:	RF1,RF8, RF9, RF10, RF14
Prioridad:	Crítico

Flujo Normal de Eventos	
Adicionar tipología	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita adicionar una nueva tipología de archivo multimedia llamando a la funcionalidad adicionar tipología de archivo multimedia.	2. El sistema comprueba que los datos enviados por la aplicación son correctos.
	3. El sistema almacena los datos de la tipología de archivo multimedia y retorna un mensaje que fue exitosa la adición.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de adición pues los datos enviados no son correctos.
Editar datos de tipología	
Acción del Actor	**Respuesta del Sistema**

1. La aplicación solicita editar una tipología de archivos multimedia.	2. El sistema busca los datos de la tipología seleccionada y los retorna.
3. La aplicación envía los datos modificados llamando a la funcionalidad editar tipología de archivo multimedia.	4. El sistema comprueba que los datos modificados son correctos.
	5. El sistema actualiza los datos de la tipología de archivo multimedia y retorna un mensaje que fue exitoso el proceso de edición.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 El sistema retorna un mensaje indicando que fue fallido el proceso de edición pues los datos enviados no son correctos.
Crear tipología	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita crear una nueva tipología de archivo multimedia.	2. El sistema verifica que los datos de la tipología seleccionada sean correctos.
	3. El sistema crea físicamente la tabla de la tipología en la base de datos. La tipología queda disponible para ser utilizada a la hora de adicionar nuevos datos asociados a los archivos multimedia.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de creación pues los datos enviados no son correctos.
Eliminar tipología	
Acción del Actor	**Respuesta del Sistema**

1. La aplicación solicita eliminar una tipología de archivos multimedia.	2. El sistema verifica que los datos de la tipología seleccionada sean correctos.
	3. El sistema elimina la tipología, en caso de estar creada borra físicamente la tabla de la base de datos y retorna un mensaje indicando que se eliminó la tipología de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de eliminación pues los datos enviados no son correctos.
Obtener Información de Tipología de Archivos Multimedia	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita obtener información de una tipología de archivos multimedia.	2. El sistema valida los datos enviados por la aplicación.
	3. El sistema busca los datos de la tipología seleccionada y los retorna.
Flujos Alternos	
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de obtener la información pues los datos enviados no son correctos.

Caso de Uso: Gestionar Campos de una tipología de Archivos Multimedia.

Caso de Uso:	Gestionar Campos de una tipología de Archivos Multimedia.
Actores:	Aplicación
Resumen:	El caso de uso se inicia cuando desde la Aplicación se desea adicionar, obtener, editar o eliminar campos de una tipología de archivos multimedia.
Precondiciones:	Que exista al menos una tipología de archivos multimedia.

Referencias:	RF2, RF3, RF4, RF15
Prioridad:	Crítico
Flujo Normal de Eventos	
Adicionar campo	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita adicionar un nuevo campo a una tipología de archivos multimedia.	2. El sistema verifica que no se haya dejado en blanco ningún campo obligatorio, que el nombre del campo en la base de datos sea correcto y que no exista otro campo con ese nombre en la tipología seleccionada.
	3. El sistema adiciona el nuevo campo a la tipología, retorna un mensaje indicando que la operación fue realizada de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de adición pues los datos enviados no son correctos.
Editar campo	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita editar un campo de tipología de archivos multimedia.	2. El sistema busca los datos del campo de tipología seleccionado y los retorna.
3. La aplicación envía los datos modificados llamando a la funcionalidad editar campo de tipología de archivos multimedia.	4. El sistema comprueba que los datos modificados sean correctos.
	5. El sistema actualiza los datos del campo tipología de archivos multimedia y retorna un mensaje indicando que fue exitoso el proceso de edición.

Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 El sistema retorna un mensaje indicando que fue fallido el proceso de edición pues los datos enviados no son correctos.
Eliminar campo	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita eliminar un campo de tipología de archivos multimedia.	2. El sistema verifica que el campo de tipología seleccionado sea correcto.
	3. El sistema elimina el campo de tipología de archivos multimedia y retorna un mensaje indicando que se eliminó el campo de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de eliminación pues los datos enviados no son correctos.
Obtener Información de un campos	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita obtener información de campos de tipología de archivos multimedia.	2. El sistema valida los datos enviados por la aplicación.
	3. El sistema busca los datos de los campos de tipología seleccionada y los retorna.
Flujos Alternos	
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de obtener la información pues los datos enviados no son correctos.

Caso de Uso: Gestionar Relaciones entre Archivos Multimedia.

Caso de Uso:	Gestionar Relaciones entre Archivos Multimedia.
Actores:	Aplicación
Resumen:	El caso de uso se inicia cuando desde la Aplicación se desea adicionar, obtener, editar o eliminar relaciones entre dos archivos multimedia.
Precondiciones:	Que exista al menos dos tipologías de archivos multimedia.
Referencias:	RF5, RF6, RF7, RF16
Prioridad:	Crítico

Flujo Normal de Eventos	
Adicionar relación	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita adicionar una nueva relación entre tipologías de archivos multimedia.	2. El sistema verifica que no se haya dejado en blanco ningún campo obligatorio y que no exista una relación con la misma tabla seleccionada con el mismo campo.
	3. El sistema adiciona la nueva relación entre tipologías de archivos multimedia, retorna un mensaje indicando que la operación fue realizada de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de adición pues los datos enviados no son correctos.
Editar relación	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita editar una relación entre tipologías de archivos multimedia.	2. El sistema busca los datos de la relación entre tipologías seleccionada y los retorna.
3. La aplicación envía los datos modificados	4. El sistema comprueba que los datos

llamando a la funcionalidad editar relación entre tipologías de archivos multimedia.	modificados sean correctos
	5. El sistema actualiza los datos de la relación entre tipologías de archivos multimedia y retorna que fue exitoso el proceso de edición.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 El sistema retorna un mensaje indicando que fue fallido el proceso de edición pues los datos enviados no son correctos.
Eliminar relación	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita eliminar una relación entre tipologías de archivos multimedia.	2. El sistema verifica que la relación entre tipologías de archivos multimedia seleccionada sea correcta.
	3. El sistema elimina la relación entre tipologías de archivos multimedia y retorna un mensaje indicando que se eliminó la relación entre tipologías de archivos multimedia de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de eliminación pues los datos enviados no son correctos.
Obtener Información de las relaciones	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita obtener información de una relación entre tipologías de archivos multimedia.	2. El sistema valida los datos enviados por la aplicación.

	3. El sistema busca los datos de la relación entre tipologías de archivos multimedia seleccionada y los retorna.
Flujos Alternos	
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de obtener la información pues los datos enviados no son correctos.

Caso de Uso: Gestionar Datos Asociados a un Archivo Multimedia.

Caso de Uso:	Gestionar Datos asociados a un archivo multimedia
Actores:	Aplicación
Resumen:	El caso de uso se inicia cuando desde la aplicación se desea adicionar, editar o eliminar los datos asociados a un archivo multimedia.
Precondiciones:	Que exista alguna tipología de archivos multimedia.
Referencias:	RF11, RF12, RF13, RF 17
Prioridad:	Crítico
Flujo Normal de Eventos	
Adicionar Datos Asociados a un Archivo Multimedia.	
Acción del Actor	**Respuesta del Sistema**
1. La aplicación solicita adicionar datos asociados a un archivo multimedia.	2. El sistema verifica los datos insertados sean correctos y que no se halla dejado en blanco ningún campo obligatorio.
	3. El sistema adiciona los datos asociados al archivo multimedia y retorna un mensaje indicando que la operación fue realizada de forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**

	2.1 El sistema muestra un mensaje indicando que los datos no se han completado correctamente.
Editar Datos Asociados a un AM	
1. La aplicación solicita editar los datos asociados a un archivo multimedia.	2. El sistema busca los datos del archivo multimedia seleccionado y los retorna.
3. La aplicación envía los datos modificados llamando a la funcionalidad editar datos asociados a un archivo multimedia.	4. El sistema comprueba que los datos modificados sean correctos.
	5. El sistema actualiza los datos del archivo multimedia y retorna un mensaje que fue exitoso el proceso de edición.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 El sistema retorna un mensaje indicando que fue fallido el proceso de edición pues los datos enviados no son correctos.
Eliminar los Datos Asociados a un AM	
1. La aplicación solicita eliminar los datos asociados a un archivo multimedia.	2. El sistema verifica los datos archivo multimedia seleccionada sea correcta.
	3. El sistema elimina los datos asociados al archivo multimedia y retorna un mensaje indicando que se eliminaron los datos de forma correcta forma correcta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de eliminación pues los datos enviados no son correctos.

Obtener datos asociados a un Archivos Multimedia	
Acción del Actor	**Respuesta del Sistema**
3. La aplicación solicita obtener datos asociados a un archivo multimedia.	4. El sistema valida los datos enviados por la aplicación.
	3. El sistema busca los datos asociados al archivo multimedia y los retorna.
Flujos Alternos	
	2.1 El sistema retorna un mensaje indicando que fue fallido el proceso de obtener la información pues los datos enviados no son correctos.

2.5 Conclusiones

En este capítulo se expuso la propuesta del sistema. Se trabajó en la fase de requerimientos, definiendo el modelo del dominio, así como los requerimientos funcionales y no funcionales que guiarán el proceso de desarrollo del sistema. Se elaboró el diagrama de casos de uso del sistema y se describieron todos los casos de uso.

Capítulo 3: Análisis y Diseño del Sistema

3.1 Introducción

En el presente capítulo se realiza el análisis y diseño de la solución propuesta. Se realiza un estudio de los patrones de diseño para lograr un entendimiento de cómo se deben desarrollar los diagramas de clase del diseño. Se modelan los diagramas de clases del análisis y del diseño de los diferentes casos de uso definidos en el capítulo anterior, y se desarrollan los modelos de datos y de despliegue del componente.

3.2 Patrones Arquitectónicos

Los patrones arquitectónicos expresan un esquema organizativo estructural fundamental para sistemas de software y definen las reglas generales de organización, las restricciones en la forma y la estructura de un grupo numeroso de sistemas de software. La selección de un patrón arquitectónico es una decisión fundamental de diseño en el desarrollo de un sistema de software.

Para el desarrollo de las funcionalidades a implementar en el componente se seleccionó el patrón arquitectónico Modelo Vista Controlador (MVC).

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo, una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación (Fabien Potencier, 2008)**.**

Modelo: Encapsula los datos y las funcionalidades. Es independiente de cualquier representación de salida y comportamiento de entrada.

Vista: Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.

Controlador: Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio para el modelo o la vista.

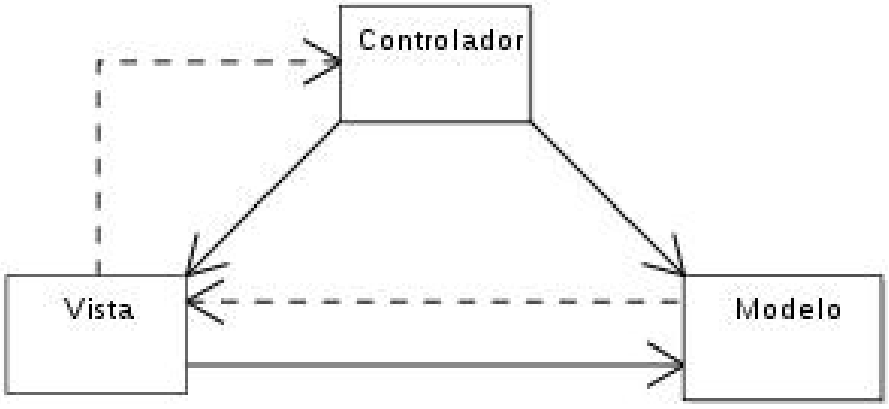


Figura 7 Modelo Vista Controlador.

3.3 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño es una solución a un problema de diseño, es importante destacar que estos facilitan la reusabilidad, extensibilidad y mantenimiento de las aplicaciones.

Para dar solución al problema planteado se hizo uso de estos patrones los cuales se clasifican en patrones de diseño **GRASP** (General Responsibility Assignment Software Patterns) y **GOF** (Gang-of-Four).

Patrón Fachada

El patrón fachada consiste en implementar una interfaz simplificada para un sistema complejo. La idea es implementar una clase con un interfaz sencillo y que encapsule los detalles de la interacción entre todas las clases del sistema. Es importante destacar que se sigue permitiendo el acceso directo a las clases del sistema, a clientes que necesiten "acceso a bajo nivel" pero se simplifica la interacción para los que no necesiten más que operaciones comunes.

Los patrones de asignación de responsabilidades **GRASP**, permiten asignar correctamente las responsabilidades a cada una de las clases que intervienen en el modelo; de este grupo de patrones fueron utilizados en el diseño e implementación del componente los siguientes:

Creador: refiere a asignar responsabilidades a las clases de crear instancias de otras conociendo que las primeras son las que contienen la información para ello.

Controlador: El patrón controlador sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa la funcionalidad, de tal forma que es la que recibe los datos y la que los envía a las distintas clases según el método llamado.

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

3.4 Modelo del Análisis

El modelo de análisis es una aproximación al modelo del diseño. En este modelo hay un refinamiento de los requisitos, sin embargo no se tiene en cuenta el lenguaje de programación que se va a utilizar en la construcción de la aplicación, debido a que el objetivo del análisis es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución.

A continuación se refleja el diagrama de clases de análisis correspondiente a los casos de uso descritos en el capítulo anterior.

3.4.1 Diagramas de Clases del Análisis

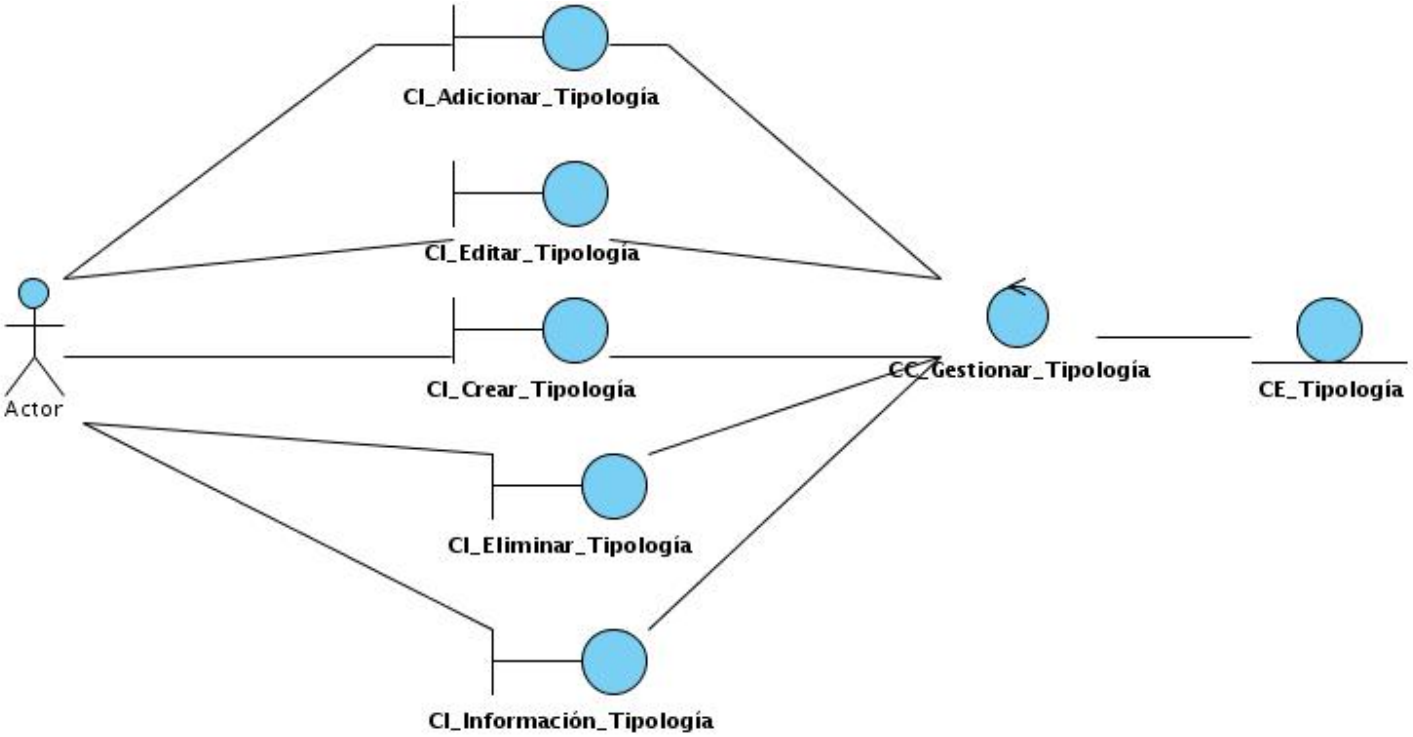


Figura 8 Diagrama de Clase del Análisis CU Gestionar Tipología

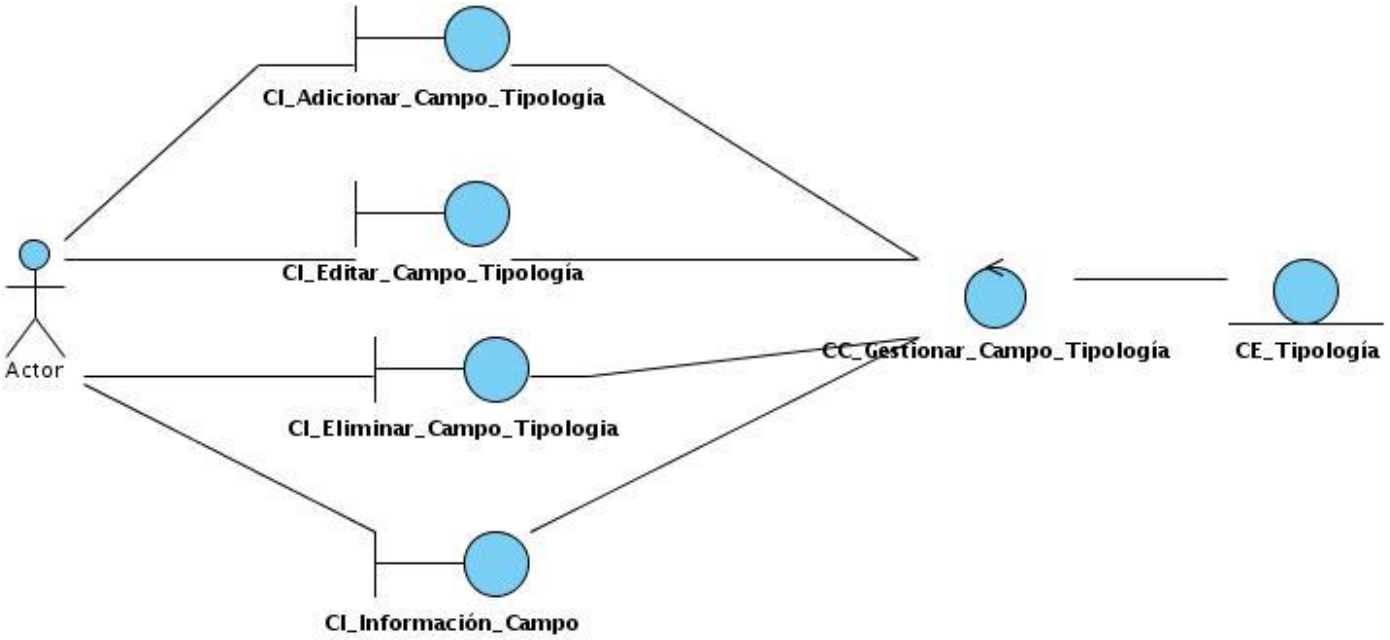


Figura 9 Diagrama de Clase del Análisis CU Gestionar Campos de Tipología

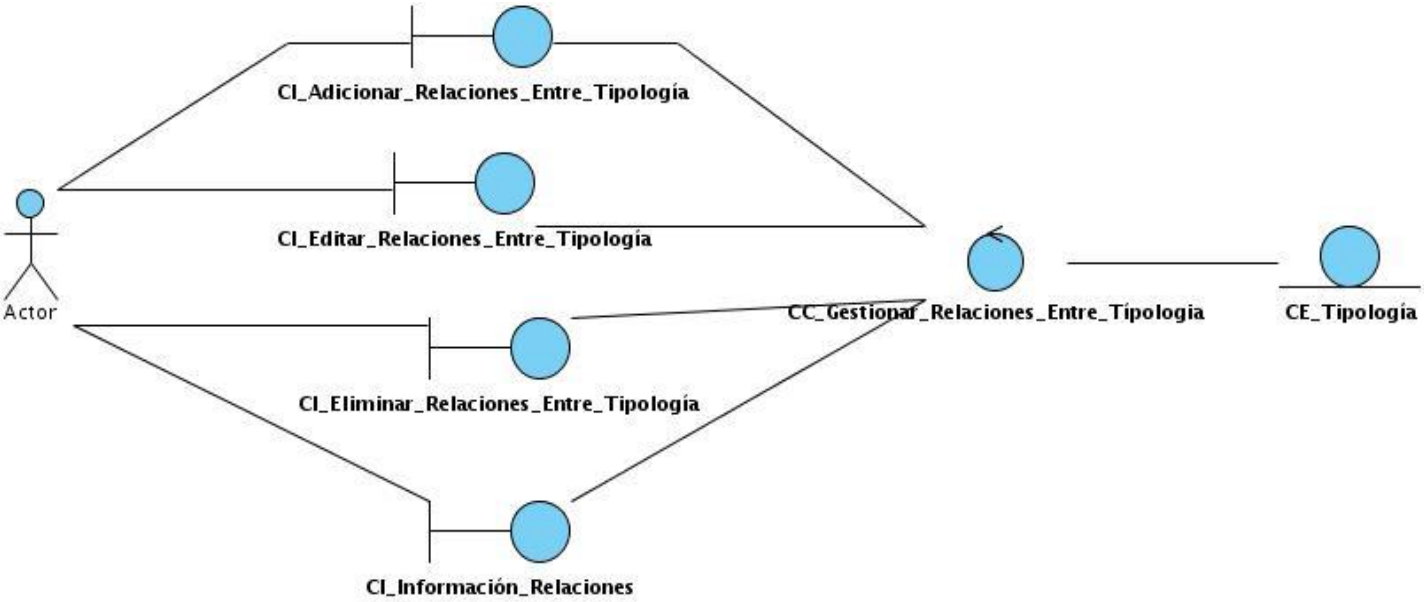


Figura 10 Diagrama de Clase del Análisis CU Gestionar Relaciones entre Tipología

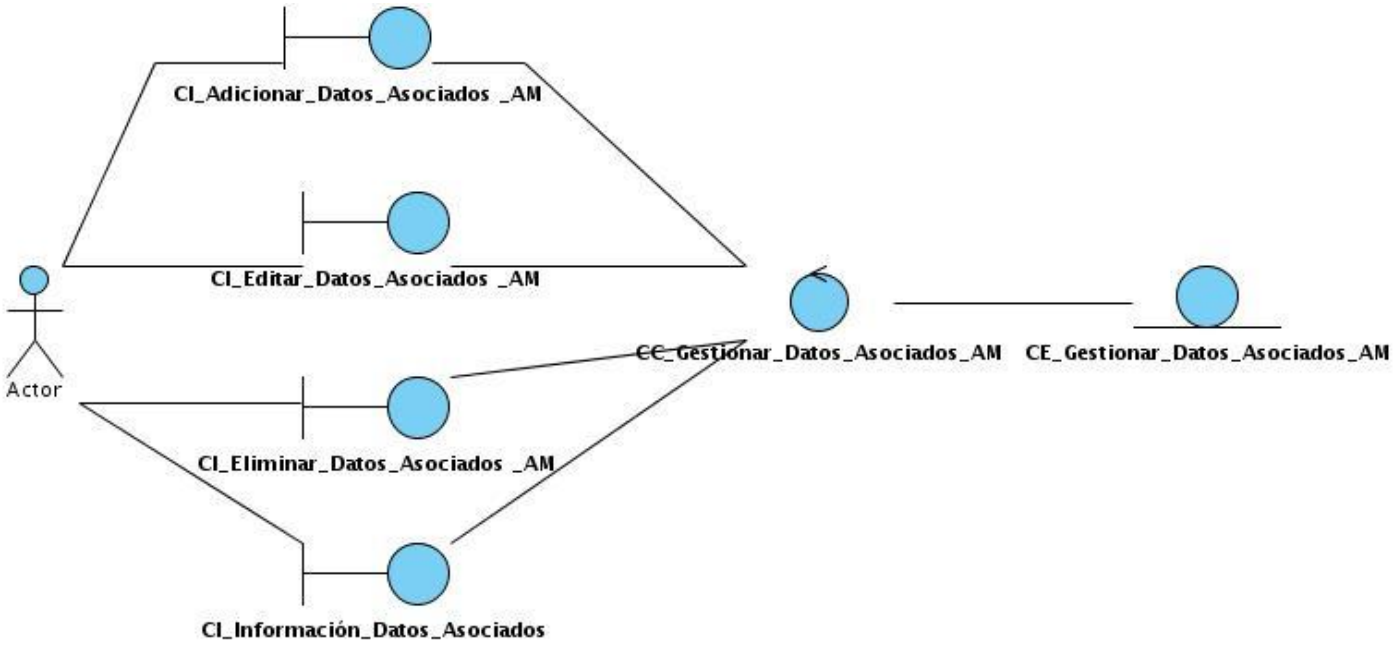


Figura 11 Diagrama de Clase del Análisis CU Gestionar Datos Asociados a un Archivo Multimedia

3.5 Modelo de Diseño

3.5.1 Diagramas de Clases del Diseño

Los diagramas de clases del diseño representan las diferentes clases que componen un sistema y cómo se relacionan unas con otras. Una clase de diseño es aquella clase suficientemente detallada que sirve como base para generar código fuente o lo que es lo mismo, es aquella cuya especificación es completa hasta un nivel que se pueda implementar.

A continuación se muestran el diagrama de clases del diseño, el cual se desarrolló siguiendo el patrón arquitectónico MVC.

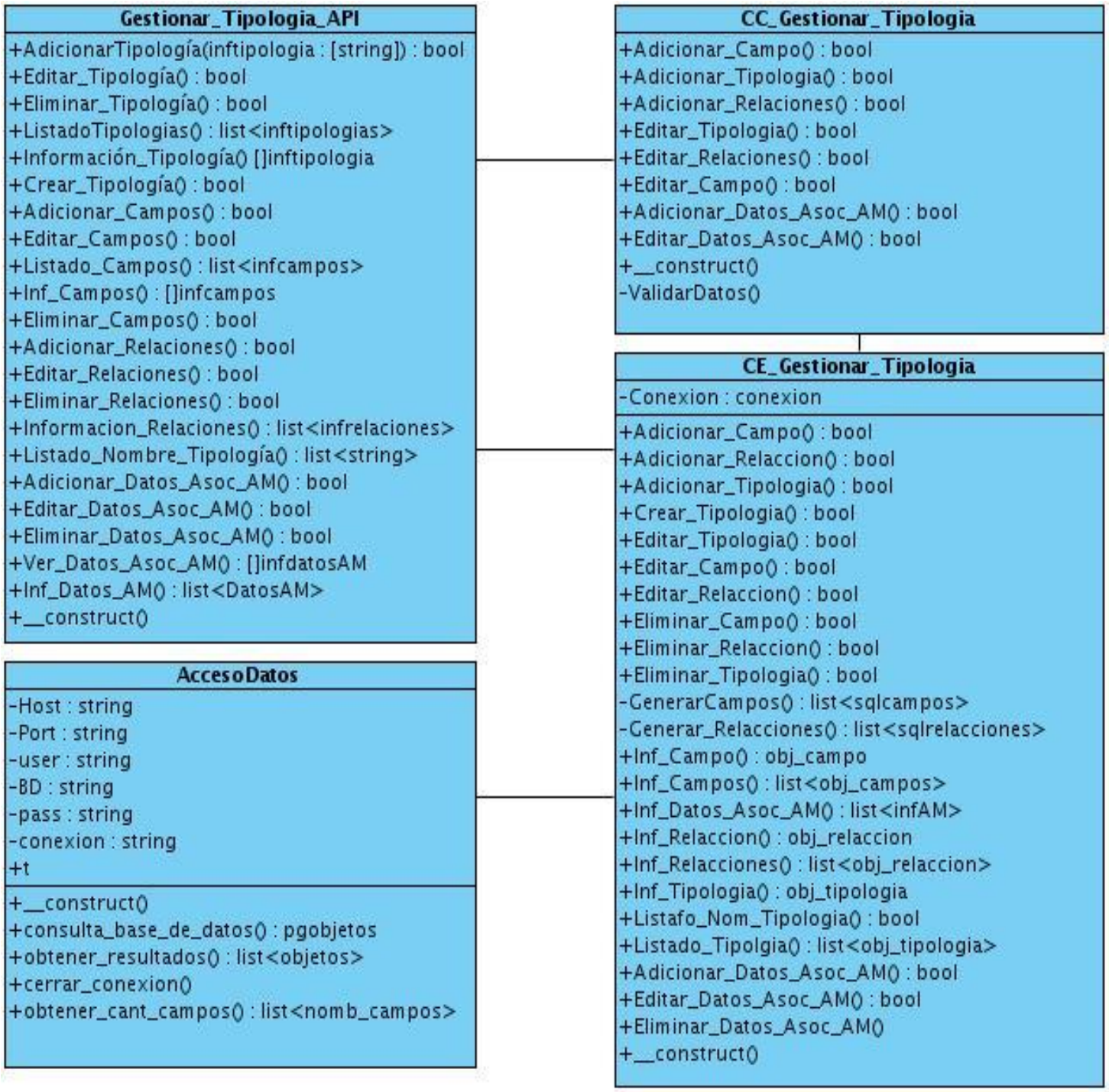


Figura 12 Diagrama de Clases del Diseño.

3.5.2 Diagramas de Interacción

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema, lo que conlleva a modelar instancias concretas o prototípicas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos. Todo en el contexto de un escenario que ilustra un comportamiento. En el contexto de las clases se describe la forma en que grupos de objetos colaboran para proveer un comportamiento.

Mientras que un diagrama de casos de uso presenta una visión externa del sistema, las funcionalidades de dichos casos de uso se recogen como un flujo de eventos y estas se utilizan para lograr representar las interacciones entre objetos.

A continuación se muestran Diagramas de Secuencia del Caso de Uso Gestionar Tipología para las sesiones Adicionar, Editar, Eliminar y Obtener información de las Tipologías de archivos multimedia en el flujo normal de los eventos.

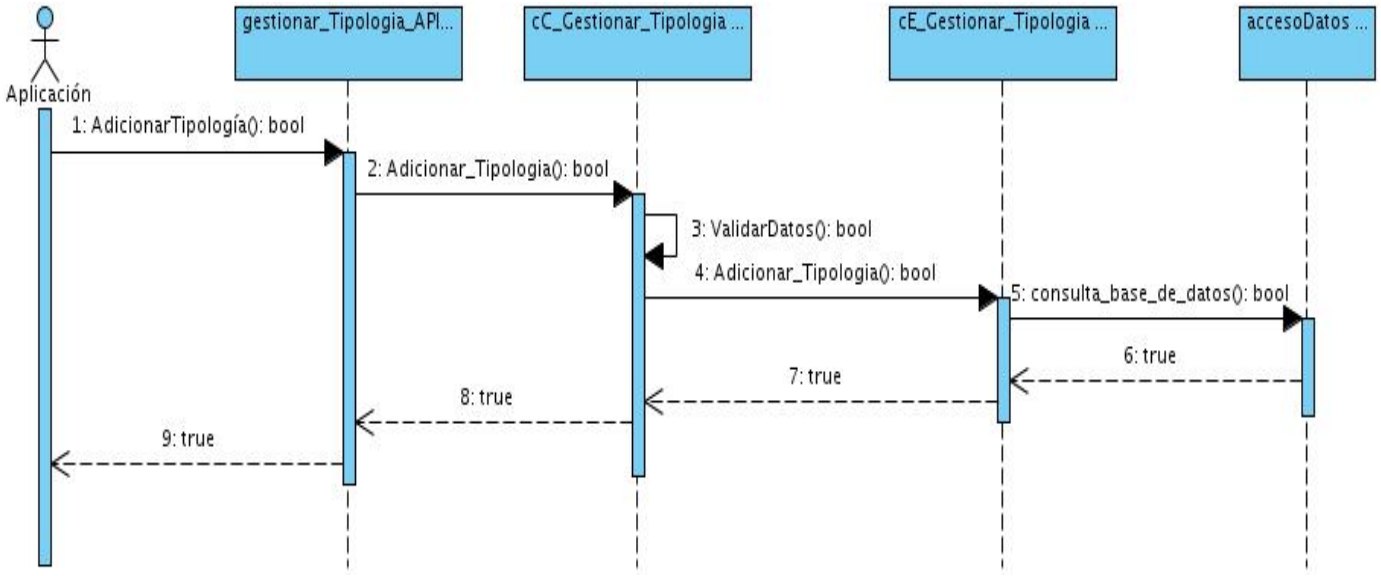


Figura 13 Diagrama de Secuencia: CU Gestionar Tipología Sesión Adicionar Tipología.

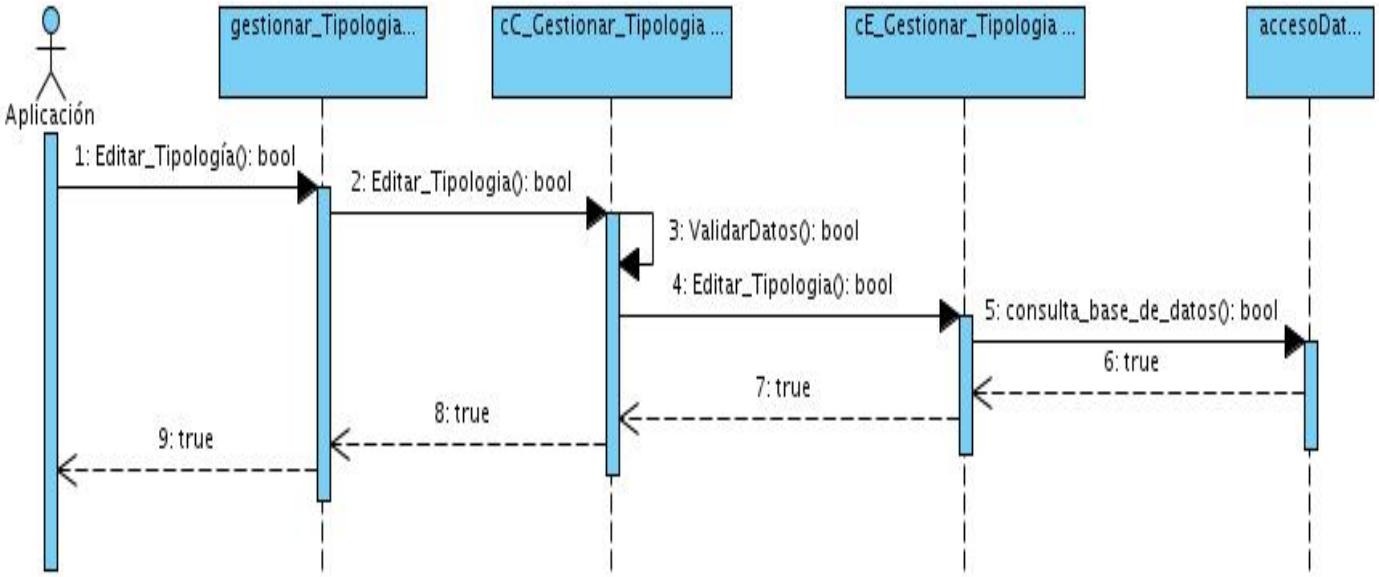


Figura 14 Diagrama de Secuencia: CU Gestionar Tipología Sesión Editar Tipología.

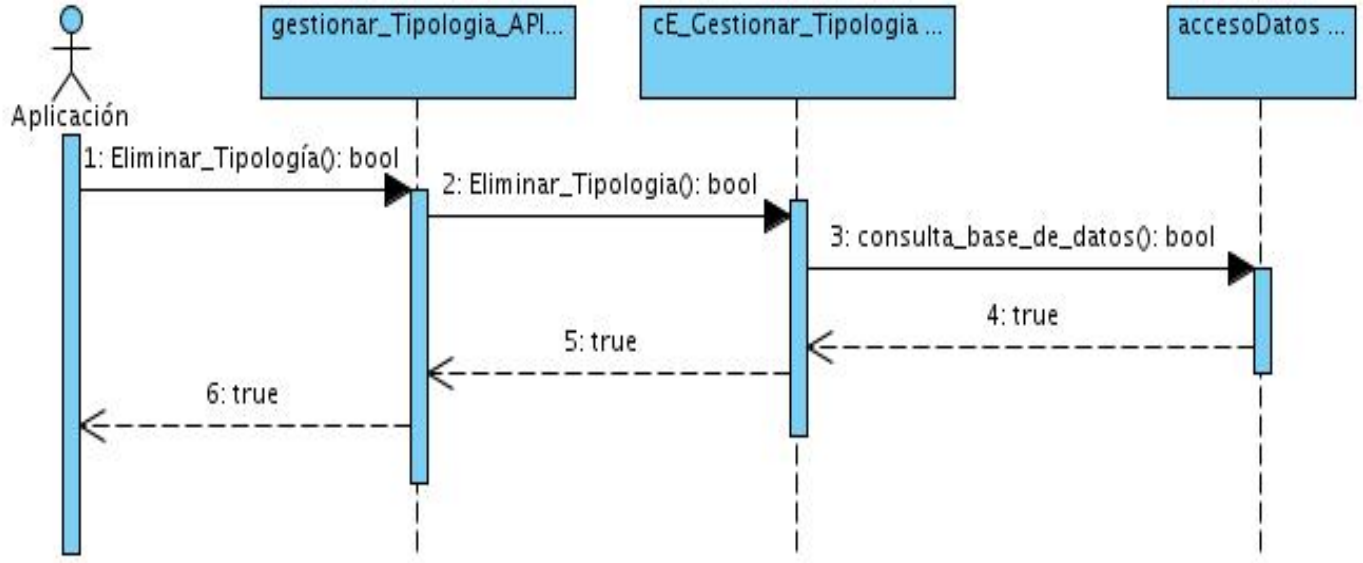


Figura 15 Diagrama de Secuencia: CU Gestionar Tipología Sesión Eliminar Tipología.

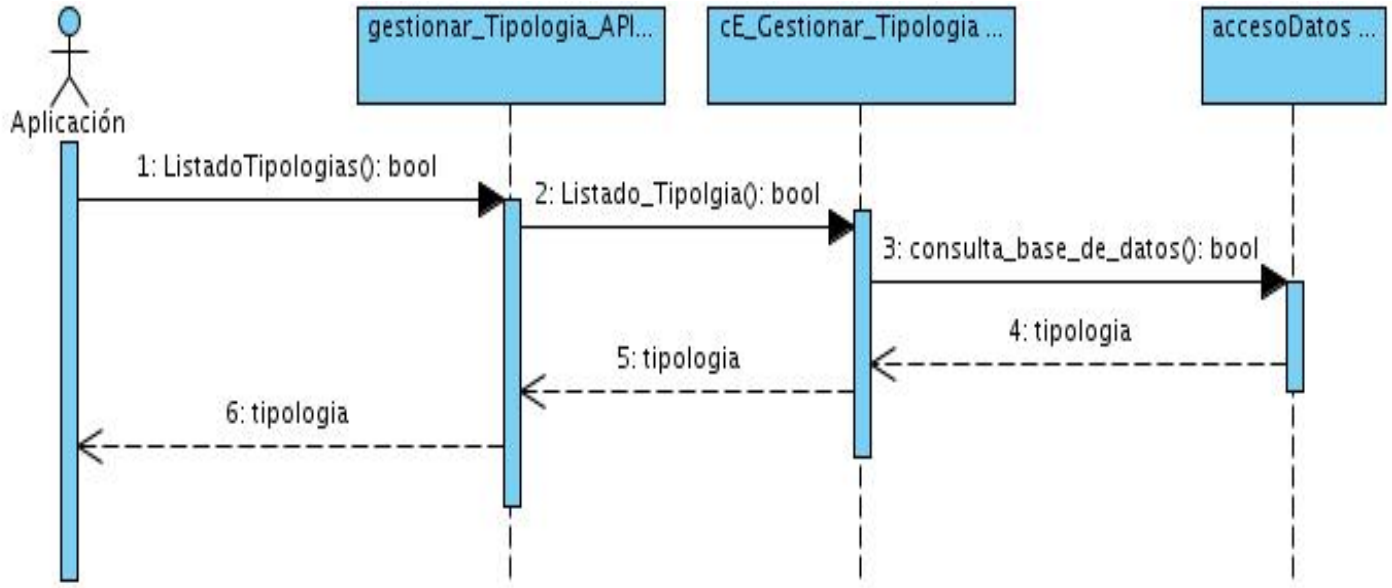


Figura 165 Diagrama de Secuencia: CU Gestionar Tipología Sesión Obtener Información de Tipología de Archivos Multimedia

3.6 Diagrama de Despliegue

El Modelo o Diagrama de Despliegue es utilizado para visualizar la distribución de los componentes de software en los nodos físicos. También se utiliza para capturar los elementos de configuración de procesamiento y las conexiones entre esos elementos.

Estos diagramas están compuesto por:

Nodos: Elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos.

Dispositivos: Nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.

Conectores: Expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.

Figura 17 Diagrama de Despliegue.

Descripción de los Nodos:

PC Cliente: Nodo cliente mediante el cual el actor accederá a la aplicación utilizando un navegador web.

Servidor Web: Nodo en el cual se encontrará instalado el servidor Web Apache el que alojará la aplicación web que contiene el componente.

Servidor de Base de Datos: En el nodo servidor de base de datos, se encontrará la base de datos del componente. A esta se accederá a través del componente acceso a datos que se implementará en la aplicación web perteneciente al nodo Servidor Web.

3.7 Modelo de Datos

En el Modelo de datos se describen las tablas que representan las distintas entidades que pertenecen al dominio del problema y serán almacenadas en la base de datos, en el caso del componente implementado tiene cuatro tablas inicialmente: tabla Definicion_Tipologia, tabla Campos de una tipología, tabla Relación entre tipologías y la tabla Tipologia_General.

De la tabla Tipologia_General heredarán las tipologías que se vayan creando en el componente los cuales tendrán las características definidas en la tabla Definicion_Tipologia.

La tabla **Definicion_Tipologia** contendrá la información necesaria para crear una tipología. Esta tabla contiene campos tales como:

- **Nombre de la Tipología:** Nombre general de la tipología.
- **Nombre de la Tabla:** Nombre de la tabla en que se va a almacenar dicha tipología.
- **Creada:** Atributo que permitirá determinar si dicha tabla está creada o no.

La tabla **Campos** almacena los campos que tendrá la tipología, esta tabla tiene una relación de uno a muchos con la tabla Definicion_Tipologia. Los campos que contiene dicha tabla son:

- **Nombre del Campo:** Almacena el nombre del campo.
- **Tipo Dato:** Almacena el tipo de dato que será el campo
- **Requerido:** Permite definir si el campo será requerido o no.

- **ID de la Tipología:** Identificador de la tipología a la cual pertenece el campo.

La tabla **Relación** almacena las diferentes relaciones que se establecen entre dos o más tablas. Esta tabla contiene los siguientes campos:

- **ID de la Tipología**: Almacena el identificador de la tipología relacionada.
- **Tipo de Relación:** Tipo de relación establecida entre las dos tablas, puede ser de uno a uno, uno a muchos.
- **Tabla que relaciona:** Nombre de la tabla con la que se relaciona la tabla de la cual se almacena el identificador.

La tabla **Tipologia_General** define la información general que se almacenará de cualquier tipología. Con esta tabla tendrán relaciones las demás tablas perteneciente a tipologías que se añadan con el uso del componente.

La siguiente figura muestra el Diagrama Entidad-Relación del componente:

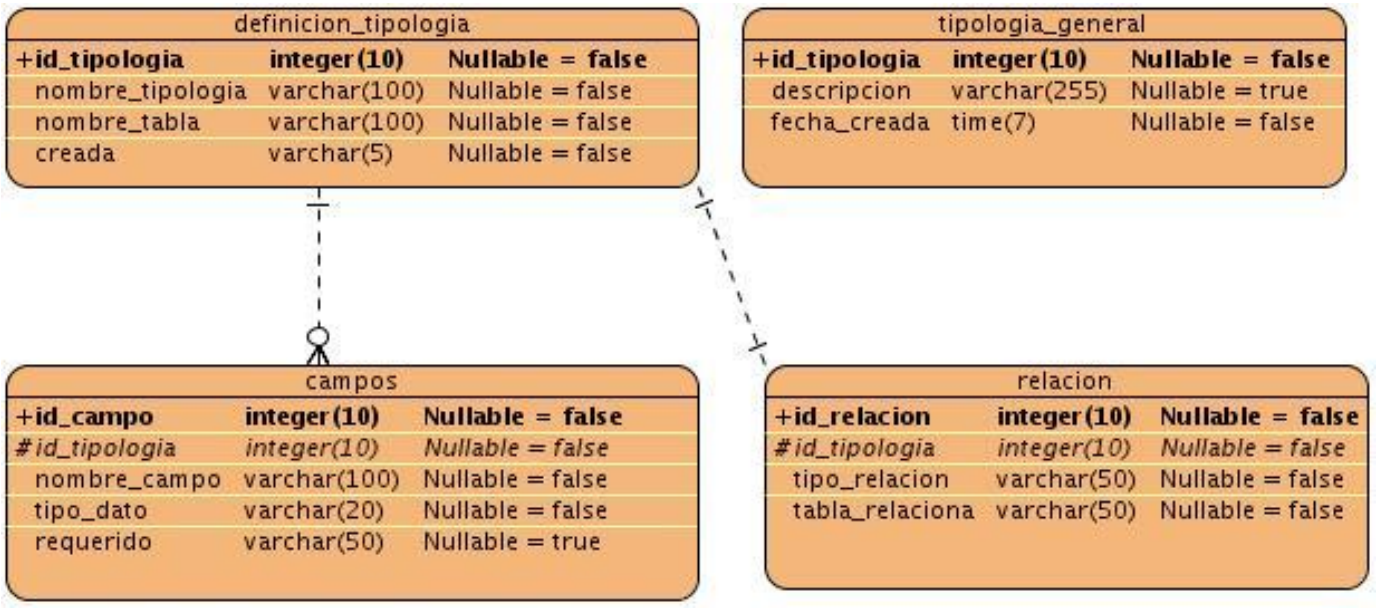


Figura 18 Diagrama Entidad-Relación.

3.8 Conclusiones

En el presente capítulo se definieron los patrones de diseño a utilizar en la implementación del componente. Se modelaron los diagramas de clase del diseño permitiendo definir los principales métodos y atributos a desarrollar durante el proceso de implementación del componente. Además se modelaron los diagramas de secuencia los cuales permiten lograr un mejor entendimiento de los flujos de los casos de uso. Se describió el diagrama de despliegue para lograr una visión de la distribución física del sistema y el diagrama de entidad-relación para usarlo posteriormente en la creación de la base de datos a utilizar en la implementación componente. Una vez completado el flujo de Análisis y Diseño se está en condiciones de comenzar con la implementación del componente.

Capítulo 4: Implementación y Pruebas

4.1 Introducción

La implementación comienza con el resultado del diseño. El presente capítulo trata sobre los flujos de trabajo implementación y prueba en el cual se desarrolla el diagrama de componentes, donde se describen los elementos físicos del sistema y sus relaciones. Además se realizan las pruebas de caja negra las cuales permiten obtener un conjunto de entradas con el objetivo de ejercitar completamente todos los requisitos funcionales del componente.

4.2 Diagrama de Componentes

Un diagrama de componentes representa la separación del sistema de software en componentes físicos (archivos, módulos, paquetes, etc.) y muestra la dependencia entre estos componentes. Se utiliza para modelar la vista estática de un sistema, además de mostrar la organización y las dependencias entre un conjunto de clases.

El diagrama de componente propuesto está compuesto por un archivo llamado Gestionar_Tipologia_API.php el cual será la interfaz del componente para las aplicaciones que lo vayan a utilizar. Además está compuesto por tres paquetes Controlador, Modelo y Acceso a Datos.

El Controlador incluye la clase controladora Gestionar Tipología y una clase para realizar las validaciones de las entradas de datos.

El Modelo está compuesto por la clase entidad Gestionar Tipología y el Paquete Acceso a Datos está compuesto por la Clase Acceso a Datos que se encargará de la conexión a la Base de Datos.

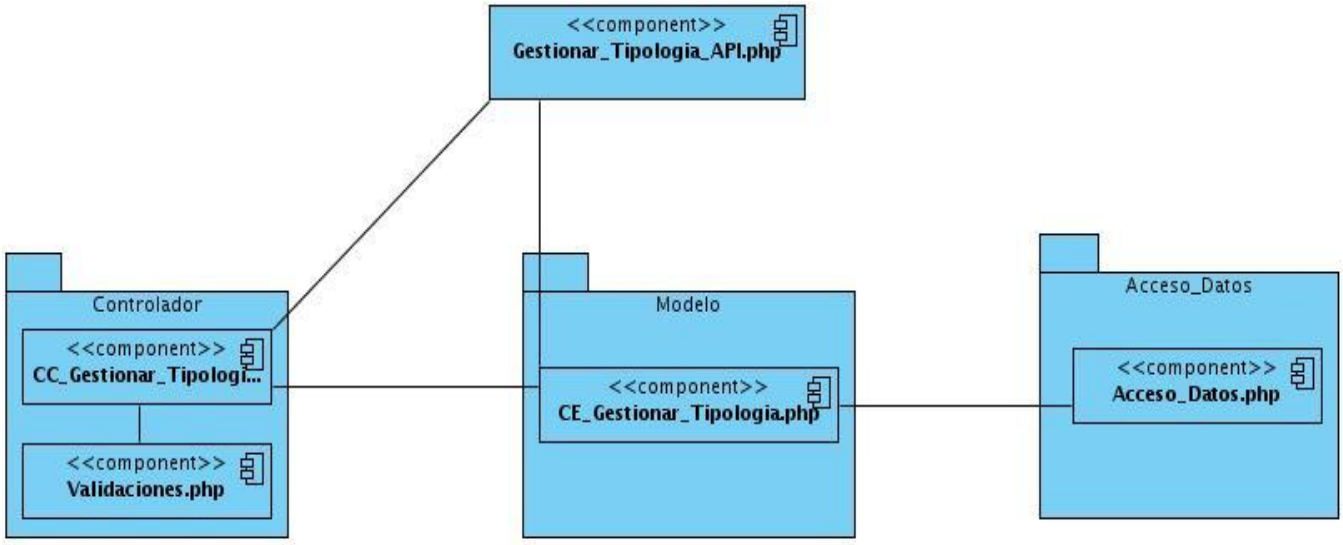


Figura 19 Diagrama de Componentes

4.2 Modelo de Prueba

El flujo de trabajo de Pruebas es uno de los flujos más importantes propuestos por la metodología de desarrollo de software utilizada (RUP) pues este presta servicios a los demás flujos. Además las pruebas se hacen con el objetivo de encontrar errores cometidos en los flujos previos para entregar el sistema al usuario final sin errores.

Las pruebas son actividades en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos especificados permitiendo que los resultados sean observados y registrados. Estas son un elemento crítico para la garantía de la calidad del software.

En el ámbito de la Ingeniería de Software existen dos métodos fundamentales de pruebas: pruebas de caja negra y pruebas de caja blanca.

4.2.1 Pruebas de Caja Negra

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene.

Para validar las principales funcionalidades del componente se realizaron pruebas de Caja Negra para las cuales fue necesario crear cuatro casos de prueba.

Casos de Prueba (CP)

Un caso de prueba es un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular. Los casos de pruebas se pueden derivar de los casos de uso del sistema o de la realización de estos en el modelo de diseño, permitiendo así validar los requerimientos funcionales del sistema.

Para el desarrollo de las pruebas al componente se desarrollaron cuatro casos de pruebas, uno por cada caso de uso del sistema. Los casos de pruebas desarrollados se muestran a continuación:

Caso de Prueba: Gestionar Tipologías de Archivos Multimedia.

Descripción del CU Gestionar Tipologías de Archivos Multimedia: El caso de uso se inicia cuando desde la aplicación se desea adicionar una nueva tipología de archivos multimedia, modificar, eliminar u obtener información de esta. Termina con la creación, modificación o eliminación de la tipología de archivo multimedia o con un mensaje de proceso fallido.

1. Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Adicionar tipología.	EC 1.1: Adicionar tipología satisfactoriamente.	La aplicación muestra un formulario para entrar los datos de la tipología, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.2: Adicionar tipología falla.	La aplicación muestra un formulario para entrar los datos de la tipología a adicionar. Una vez enviados por el usuario el sistema le muestra un mensaje informando que estos no son correctos.
SC 2: Editar datos de tipología.	EC 2.1: Editar datos de tipologías satisfactoriamente.	La aplicación muestra los datos de la tipología seleccionada por el usuario, el usuario edita los datos y los envía. Los datos se modifican correctamente y se muestra un mensaje al usuario indicando que la edición fue satisfactoria.

	EC 2.2: Editar datos de tipologías falla.	La aplicación muestra los datos de la tipología seleccionada por el usuario, el usuario edita los datos y los envía. El sistema muestra un mensaje indicando que los datos no son correctos.
SC 3: Crear tipología.	EC 3.1: Crear tipología satisfactoriamente	El usuario selecciona la tipología a crear, la aplicación crea la tipología de forma satisfactoria.
	EC 3.2: Crear tipología falla	El usuario selecciona la tipología a crear, la aplicación muestra un mensaje indicando que ocurrió un error en el proceso de creación de la tipología.
SC 4: Eliminar tipología.	EC 4.1: Eliminar tipología satisfactoriamente	El usuario selecciona la tipología a eliminar y la aplicación elimina la tipología satisfactoriamente.
	EC 4.2: Eliminar tipología falla.	El usuario selecciona la tipología a eliminar, la aplicación muestra un mensaje indicando que ocurrió un error durante el proceso de eliminación.
SC 5: Obtener información de tipología de archivos multimedia	EC 5.1: Obtener información de tipología de archivos multimedia satisfactoria.	El usuario accede a la página de información de tipologías de archivos multimedia y la aplicación muestra la información de forma satisfactoria.
	EC 5.2 Obtener información de tipología de archivos multimedia falla.	El usuario accede a la página de información de tipologías de archivos multimedia y la aplicación muestra un mensaje indicando que ocurrió un error en el proceso de mostrar la información relativa a la tipología de archivos multimedia.

2. Descripción de variable.

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre_ tipología	Campo de texto	No	Cualquier combinación de letras, números, guiones bajo y espacios.
2	Nombre_ tabla	Campo de texto	Si	Cualquier combinación de letras, números y guiones bajo que no comience con números.
3	Id de la tipología	Campo oculto	No	Número entero que debe estar en correspondencia con el identificador de una tipología existente en la base de datos.

3. Secciones a revisar.

SC 1: Adicionar tipología

Id del escenario	Nombre_ tipología	Nombre_ tabla	Respuesta del Sistema	Resultado de la Prueba
EC1 Adicionar tipología satisfactoriamente.	V	V	Se añade la tipología y se muestra un mensaje informando que se añadió satisfactoriamente.	Satisfactoria
EC2 Adicionar tipología falla.	V	I	Se muestra un mensaje indicando que la tipología no se adicionó pues los datos no son correctos.	Satisfactoria
	I	V		
	I	I		

SC 2: Editar datos de tipología

Id del escenario	Nombre_ tipología	Nombre_ tabla	Id de la tipología	Respuesta del Sistema	Resultado de la Prueba
EC1 Editar datos de tipología satisfactoria mente.	V	V	V	Se edita la tipología correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC2 Editar datos de tipología falla.	V	I	V	Se muestra un mensaje indicando que los datos de la tipología no se editaron pues estos no son correctos.	Satisfactoria
	I	V	V		
	V	V	I		
	I	I	I		

SC 3: Crear tipología

Id del escenario	Id de la tipología	Respuesta del Sistema	Resultado de la Prueba
EC1 Crear tipología satisfactoria mente.	V	Se crea la tipología correctamente y se muestra un mensaje informando que se la operación fue satisfactoria.	Satisfactoria
EC 2 Crear tipología falla.	I	Se muestra un mensaje indicando que la tipología no se pudo crear de forma satisfactoria.	Satisfactoria.

SC 4: Eliminar tipología

Id del escenario	Id de la tipología	Respuesta del Sistema	Resultado de la Prueba
EC1 Eliminar tipología satisfactoriamente.	V	Se elimina la tipología correctamente y se muestra un mensaje informando que se la operación fue satisfactoria.	Satisfactoria
EC2 Eliminar tipología falla.	I	Se muestra un mensaje indicando que la tipología no se pudo eliminar de forma satisfactoria.	Satisfactoria

SC 5: Obtener Información de Tipología de Archivos Multimedia

Id del escenario	Id de la tipología	Respuesta del Sistema	Resultado de la Prueba
EC1 Obtener Información de Tipología de Archivos Multimedia satisfactoriamente.	V	Se obtiene la información de la tipología de archivos multimedia correctamente.	Satisfactoria
EC 2 Obtener Información de Tipología de Archivos Multimedia falla.	I	Se muestra un mensaje indicando que no se puede mostrar la información de la tipología de archivos multimedia.	Satisfactoria

En el **Anexo 2** se encuentran los restantes tres casos de prueba aplicados al componente.

Resultado de las pruebas:

Fueron probados cuatro casos de uso, los que representan el cien por ciento del total de requisitos funcionales. Para cada requisito se tuvo en cuenta los diferentes escenarios que pudieran existir, los cuales implican cada una de las posibles interacciones del usuario o combinaciones de situaciones posibles con el fin de evaluar el comportamiento del sistema ante cada funcionalidad.

El resultado de cada prueba coincide con la especificación de requisito lo que demuestra la correcta implementación de los mismos.

4.3 Conclusiones

Como resultado de este capítulo se obtuvo el diagrama de componentes el cual muestra la estructura física que tiene el componente de software. Además para validar la completitud de respuesta a los requerimientos identificados en el Capítulo 2 se realizaron pruebas de caja negra al componente, quedando elaborado de esta forma el Modelo de Pruebas, conformado por cuatro casos de prueba que se corresponden con los cuatro casos de uso identificados anteriormente.

Conclusiones Generales

La realización de la investigación arrojó los siguientes resultados que dan cumplimiento a los objetivos propuestos:

- Se realizó un estudio de los principales conceptos relacionados con los estándares de metadatos existentes demostrando la necesidad de implementar un componente para aplicaciones web que permita estandarizar la gestión de la información relativa a los archivos multimedia en el Departamento de Señales Digitales.
- Para desarrollar el sistema se caracterizaron las herramientas y tecnologías lo que permitió sentar las bases para el posterior trabajo con las mismas.
- Se definieron los requisitos funcionales, los no funcionales y los casos de uso con sus descripciones permitiendo definir las características del sistema y sirviendo de guía para los flujos y fases posteriores.
- Para comprobar la calidad y correcto funcionamiento del sistema, se diseñaron y ejecutaron casos de prueba, los que arrojaron resultados satisfactorios, demostrando el cumplimiento de los requerimientos funcionales establecidos en la fase inicial del proceso de desarrollo del componente.
- El componente implementado como resultado de la investigación realizada será de gran utilidad en el proceso de estandarización de la gestión de los datos asociados a los archivos multimedia en las aplicaciones web del Departamento de Señales Digitales.

Recomendaciones

Al concluir el presente trabajo se recomienda:

- ✓ Implementar mecanismos de extracción de metadatos de forma automática con el objetivo de facilitar el trabajo de las personas que usen el componente.
- ✓ Agregar al componente una nueva funcionalidad que permita crear las tipologías de forma automática basadas en el estándar Dublin Core.
- ✓ Aplicar pruebas de caja blanca al componente para descubrir errores en el código fuente de la aplicación que no hayan sido explotados con las pruebas de caja negra.

Referencias bibliográficas

1. **Abreu, Ángel Dayán. 2009.** Desarrollo de la Plataforma VideoWeb. 2009.
2. **Adobe. 2010.** Adobe. Adobe CS5 After Effects Help. [En línea] 2010. [Citado el: 20 de noviembre de 2010.] http://help.adobe.com/es_ES/aftereffects/cs/using/WSb2dae976454572dc143e6a65126e3648da1-8000.html.
3. **Álvarez, Miguel Ángel. 2008.** Desarrolloweb.com. Desarrolloweb.com. [En línea] 11 de noviembre de 2008. [Citado el: 07 de noviembre de 2010.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.
4. **Ávila, Katty. 2008.** cavsi.com. ¿Qué es un Sistema Gestor de Bases de Datos o SGBD? [En línea] 2008. [Citado el: 13 de noviembre de 2010.] http://www.cavsi.com/.
5. **Ballesteros, Jordi Delcor. 2006.** DESCRIPCIÓN, INDEXACIÓN, BÚSQUEDA Y ADQUISICIÓN DE SECUENCIAS DE VÍDEO MEDIANTE DESCRIPTORES MPEG-7. [aut. libro] Verónica Pérez Noriega Jordi Delcor Ballesteros. Madrid, España: s.n., 2006, págs. 1-102.
6. **Castellanos, Henry Cordero.** [En línea]
7. **Cira Orta, Yennis Puente, Germán Orta. 2006.** Análisis y Diseño de Sistemas. Análisis y Diseño de Sistemas. Caracas: s.n., 2006.
8. **Definición. 2008.** Definición.org. Definición de Lenguaje de Programación. [En línea] 25 de Marzo de 2008. [Citado el: 11 de noviembre de 2010.] http://www.definicion.org/lenguaje-de-programacion.
9. **Exif. 2008.** Exif.org. Home Exif.org. [En línea] 2008. [Citado el: 20 de noviembre de 2010.] http://www.exif.org.
10. **Fabien Potencier, Francois Zaninotto. 2008.** Symfony la guía definitiva. 2008.
11. **INFLANET. 1998.** Federation of Library Association and Institution. INFLANET. [En línea] octubre de 1998. [Citado el: 06 de noviembre de 2010.] http://www.ifla.org/IV/ifla64/007-126s.htm.
12. **Jacaboson, Booch, Rumbaugh. 2000.** El Proceso Unificado de Desarrollo de Software. Madrid: Addison Wesley, 2000.
13. **Martínez, Rafael. 2010.** Portal de PostgreSQL en Español. Sobre PostgreSQL. [En línea] 02 de noviembre de 2010. [Citado el: 13 de noviembre de 2010.] http://www.postgresql-es.org/sobre_postgresql.
14. **Masa, Javier. 2008.** Dublin Core en castellano. Dublin Core en castellano. [En línea] 01 de diciembre de 2008. [Citado el: 20 de noviembre de 2010.] http://www.rediris.es/search/dces/.
15. **Morales, Julieta Sánchez. 2005.** El formato Dublin Core como sistema de catalogación electrónico. El formato Dublin Core como sistema de catalogación electrónico. [En línea] junio de 2005. [Citado el: 05 de noviembre de 2010.] http://www.mati.unam.mx/index.php.
16. **ONeill, Dan. 2010.** ID3.ORG. HOME ID3.ORG. [En línea] 02 de noviembre de 2010. [Citado el: 20 de noviembre de 2010.] http://www.id3.org/.

17. **Paulus, Cristian Vásquez. 2008.** METADATOS: Introducción e historia. METADATOS. Chile: s.n., 2008.
18. **Sánchez, Jordi. 2006.** Jordisan.net. Jordisan.net. [En línea] 26 de septiembre de 2006. [Citado el: 07 de noviembre de 2010.] http://jordisan.net/blog/2006/que-es-un-framework.
19. **2009. Scribd**. Desarrollo de Aplicaciones Interactivas en Java. [En línea] Julio de 2009. [Citado el: 11 de noviembre de 2010.] http://www.scribd.com/doc/967380/Tutorial-Netbeans.
20. **Serrano, Antonio Eleazaar López. 2008.** Historia de los metadatos. Historia de los metadatos. [En línea] 2008. [Citado el: 04 de noviembre de 2010.] http://sistemasavanzadosderecuperaciondeinformacion.iespana.es/historia.html.
21. **Skilar, David. 2005.** Introducción a PHP 5. Madrid: Norwich & Barston, 2005.
22. **UMurcia. 2006.** Universidad de Murcia. Metodologías de desarrollo de software. [En línea] 30 de Diciembre de 2006. [Citado el: 12 de noviembre de 2010.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.
23. **Valentine Chelsea, Minnick Chris. 2003.** Serie Práctica XHTML. s.l. : Editorial Prentice Hall, 2003.
24. **W3C. 2010.** W3C. Extensible Markup Language (XML). [En línea] 06 de Septiembre de 2010. [Citado el: 13 de noviembre de 2010.] http://www.w3.org/XML/.
25. **Wordreference. 2010.** Wordreference.com. Wordreference.com. [En línea] 2010. [Citado el: 07 de noviembre de 2010.] http://www.wordreference.com/definicion/catalogaci%C3%B3n.

Bibliografía

1. **Abreu, Ángel Dayán. 2009.** Desarrollo de la Plataforma VideoWeb. 2009.
2. **Adobe. 2010.** Adobe. Adobe CS5 After Effects Help. [En línea] 2010. [Citado el: 20 de noviembre de 2010.] http://help.adobe.com/es_ES/aftereffects/cs/using/WSb2dae976454572dc143e6a65126e3648da1-8000.html.
3. **Álvarez, Miguel Ángel. 2008.** Desarrolloweb.com. Desarrolloweb.com. [En línea] 11 de noviembre de 2008. [Citado el: 07 de noviembre de 2010.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.
4. **2011. Apache Software Foundation.** Apache Software Foundation. [En línea] The Apache Software Foundation, 2011. [Citado el: 06 de diciembre de 2010.] http://www.apache.org/foundation/sponsorship.html.
5. **Ávila, Katty. 2008.** cavsi.com. ¿Qué es un Sistema Gestor de Bases de Datos o SGBD? . [En línea] 2008. [Citado el: 13 de noviembre de 2010.] http://www.cavsi.com/.
6. **Ballesteros, Jordi Delcor. 2006.** DESCRIPCIÓN, INDEXACIÓN, BÚSQUEDA Y ADQUISICIÓN DE SECUENCIAS DE VÍDEO MEDIANTE DESCRIPTORES MPEG-7. [aut. libro] Verónica Pérez Noriega Jordi Delcor Ballesteros. Madrid, España: s.n., 2006, págs. 1-102.
7. **2009. CalidadySoftware.com**. CalidadySoftware.com. [En línea] CalidadySoftware.com, 2009. [Citado el: 12 de mayo de 2011.] http://www.calidadysoftware.com/testing/casos_de_prueba.php.
8. **Castellanos, Henry Cordero.** [En línea]
9. **Cira Orta, Yennis Puente, Germán Orta. 2006.** Análisis y Diseño de Sistemas. Análisis y Diseño de Sistemas. Caracas: s.n., 2006.
10. **Definición. 2008.** Definición.org. Definición de Lenguaje de Programación. [En línea] 25 de marzo de 2008. [Citado el: 11 de noviembre de 2010.] http://www.definicion.org/lenguaje-de-programacion.
11. **Exif. 2008.** Exif.org. Home Exif.org. [En línea] 2008. [Citado el: 20 de noviembre de 2010.] http://www.exif.org.
12. **Fabien Potencier, François Zaninotto. 2008.** Symfony la guía definitiva. 2008.
13. **IBM. 2011.** IBM. IBM Rational Unified Process (RUP). [En línea] IMB, 2011. [Citado el: 12 de diciembre de 2010.] http://www-01.ibm.com/software/awdtools/rup/#.
14. **INFLANET. 1998.** Federation of Library Association and Institution. INFLANET. [En línea] octubre de 1998. [Citado el: 06 de noviembre de 2010.] http://www.ifla.org/IV/ifla64/007-126s.htm.
15. **Jacaboson, Booch, Rumbaugh. 2000.** El Proceso Unificado de Desarrollo de Software. Madrid: Addison Wesley, 2000.
16. **Martínez, Rafael. 2010.** Portal de PostgreSQL en Español. Sobre PostgreSQL. [En línea] 02 de noviembre de 2010. [Citado el: 13 de noviembre de 2010.] http://www.postgresql-es.org/sobre_postgresql.

17. **Masa, Javier. 2008.** Dublin Core en castellano. Dublin Core en castellano. [En línea] 01 de diciembre de 2008. [Citado el: 20 de noviembre de 2010.] http://www.rediris.es/search/dces/.
18. **Morales, Julieta Sánchez. 2005.** El formato Dublin Core como sistema de catalogación electrónico. El formato Dublin Core como sistema de catalogación electrónico. [En línea] Junio de 2005. [Citado el: 05 de noviembre de 2010.] http://www.mati.unam.mx/index.php.
19. **2011. Netbeans.org.** Netbeans.org. [En línea] Oracle Corporation, 2011. [Citado el: 02 de 12 de 2010.] http://netbeans.org/index_es.html.
20. **ONeill, Dan. 2010.** ID3.ORG. HOME ID3.ORG. [En línea] 02 de noviembre de 2010. [Citado el: 20 de noviembre de 2010.] http://www.id3.org/.
21. **Paulus, Cristian Vásquez. 2008.** METADATOS: Introducción e historia. METADATOS. Chile: s.n., 2008.
22. **pgAdmin Tools. 2011.** pgAdmin Tools. pgAdmin Tools. [En línea] pgAdmin Tools, 2011. [Citado el: 13 de noviembre de 2010.] http://www.pgadmin.org/.
23. **2011. PHP.Net.** PHP.Net. [En línea] The PHP Group, 2011. [Citado el: 25 de abril de 2011.] http://php.net/manual/es/index.php.
24. **2011. PostgreSQL-ES.** PostgreSQL-ES. [En línea] PosgreSQL-es, 2011. [Citado el: 30 de noviembre de 2010.] http://www.postgresql.org.es/.
25. **Sánchez, Jordi. 2006.** Jordisan.net. Jordisan.net. [En línea] 26 de septiembre de 2006. [Citado el: 07 de noviembre de 2010.] http://jordisan.net/blog/2006/que-es-un-framework.
26. **2009. Scribd.** Desarrollo de Aplicaciones Interactivas en Java. [En línea] Julio de 2009. [Citado el: 11 de noviembre de 2010.] http://www.scribd.com/doc/967380/Tutorial-Netbeans.
27. **Serrano, Antonio Eleazaar López. 2008.** Historia de los metadatos. Historia de los metadatos. [En línea] 2008. [Citado el: 04 de noviembre de 2010.] http://sistemasavanzadosderecuperaciondeinformacion.iespana.es/historia.html.
28. **Skilar, David. 2005.** Introducción a PHP 5. Madrid: Norwich & Barston, 2005.
29. **UMurcia. 2006.** Universidad de Murcia. Metodologías de desarrollo de software. [En línea] 30 de diciembre de 2006. [Citado el: 12 de noviembre de 2010.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.
30. **Valentine Chelsea, Minnick Chris. 2003.** Serie Práctica XHTML. s.l: Editorial Prentice Hall, 2003.
31. **W3C. 2010.** W3C. Extensible Markup Language (XML). [En línea] 06 de septiembre de 2010. [Citado el: 13 de noviembre de 2010.] http://www.w3.org/XML/.
32. **Wordreference. 2010.** Wordreference.com. Wordreference.com. [En línea] 2010. [Citado el: 07 de noviembre de 2010.] http://www.wordreference.com/definicion/catalogaci%C3%B3n.

Anexo 1. Glosario de Términos.

Glosario de Términos.

Adobe: Adobe Systems Incorporated, es una empresa de software con sede en San José (California, USA). Destaca en el mundo del software por sus programas de edición de páginas web, vídeo e imagen digital.

API: es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Catalogación: Acción y efecto de catalogar.

Catálogo: relación ordenada de elementos pertenecientes al mismo conjunto, que por su número precisan de esa catalogación para facilitar su localización; por ejemplo, en un archivo o una biblioteca.

Descriptores visuales: describen las características visuales de los contenidos dispuestos en imágenes o en vídeos. Describen características elementales tales como la forma, el color, la textura o el movimiento, entre otros.

Drupal: es un sistema de gestión de contenido modular multipropósito y muy configurable que permite publicar artículos, imágenes u otros archivos.

Indexación: operación de registrar de forma ordenada la información para confeccionar un índice.

Metadatos: son objetos que describen o proporcionan información sobre otros datos.

Multimedia: elemento que utiliza conjunta y simultáneamente diversos medios, como imágenes, sonidos y texto, en la transmisión de una información.

Tipología: literalmente el estudio de los tipos, se encarga, en diversos campos de estudio, de realizar una clasificación de diferentes elementos.

Anexo 2. Casos de Prueba.

Caso de Prueba. Gestionar Campos.

Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Adicionar campo	EC 1.1: Adicionar campo satisfactoriamente.	La aplicación muestra un formulario para entrar los datos del campo, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.2: Adicionar campo falla.	La aplicación muestra un formulario para entrar los datos del campo a insertar y una vez enviados por el usuario le muestra un mensaje informando que estos no son correctos.
SC 2:Editar campo	EC 2.1: Editar campo satisfactoriamente.	La aplicación muestra los datos del campo seleccionado por el usuario, el usuario edita los datos y los envía. Los datos se modifican y se muestra un mensaje al usuario indicando que edición fue satisfactoria.
	EC 2.2: Editar campos falla.	La aplicación muestra los datos del campo, el usuario edita los datos y los envía. El sistema muestra un mensaje indicando que los datos no son correctos.
SC 3: Eliminar Campo	EC 4.1: Eliminar campo satisfactoriamente.	El usuario selecciona el campo a eliminar y la aplicación elimina el campo satisfactoriamente.
	EC 4.2: Eliminar campo falla.	El usuario selecciona el campo a eliminar, la aplicación muestra un mensaje indicando que ocurrió un error durante el proceso de eliminación.

SC 1: Adicionar campo

Id del escenario	Nombre_ campo	Tipo de dato	Requerido	Respuesta del Sistema	Resultado de la Prueba
EC1 Adicionar campo satisfactoriamente.	V	V	V	Se añade el campo y se muestra un mensaje informando que se añadió satisfactoriamente.	Satisfactoria
EC2 Adicionar campo falla.	V	I	V	Se muestra un mensaje indicando que el campo no se adicionó.	Satisfactoria.
	I	V	V		
	V	V	I		
	I	I	I		

SC 2: Editar campo

Id del escenario	Nombre_ campo	Tipo de dato	Requerido	Respuesta del Sistema	Resultado de la Prueba
EC1 Editar campo satisfactoriamente.	V	V	V	Se edita el campo correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC 2 Editar campo falla.	V	I	V	Se muestra un mensaje indicando que los datos del campo no se editaron correctamente.	Satisfactoria.
	I	V	V		
	V	V	I		
	I	I	I		

SC 3: Eliminar campo

Id del escenario	Id_campo	Respuesta del Sistema	Resultado de la Prueba
EC1 Eliminar campo satisfactoriam	V	Se elimina el campo correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC 2 Eliminar campo falla.	I	Se muestra un mensaje indicando que el campo no se pudo eliminar de forma satisfactoria.	Satisfactoria.

Caso de Pruebas. Gestionar Relaciones entre Archivos Multimedia.

Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Adicionar relación	EC 1.1: Adicionar relación satisfactoriamente.	La aplicación muestra un formulario para entrar los datos de la relación, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.2: Adicionar relación falla.	La aplicación muestra un formulario para entrar los datos de la relación a insertar y una vez enviados por el usuario, le muestra un mensaje informando que estos no son correctos.
SC 2:Editar relación	EC 2.1: Editar relación satisfactoriamente.	La aplicación muestra los datos de la relación seleccionada por el usuario, el usuario edita los datos y los envía. Los datos se modifican correctamente y se muestra un mensaje al usuario indicando que la edición fue satisfactoria.
	EC 2.2: Editar relación falla.	La aplicación muestra los datos de la relación, el usuario edita los datos y los envía. El sistema muestra un mensaje indicando que los datos no son correctos.

SC 4: Eliminar relación	EC 4.1: Eliminar relación satisfactoriamente	El usuario selecciona la relación a eliminar y la aplicación elimina el campo satisfactoriamente.
	EC 4.2: Eliminar relación falla.	El usuario selecciona la relación a eliminar, la aplicación muestra un mensaje indicando que ocurrió un error durante el proceso de eliminación.

SC 1: Adicionar relación

Id del escenario	Tabla relacionada	Tipo de relación	Respuesta del Sistema	Resultado de la Prueba
EC1 Adicionar relación satisfactoriamente.	V	V	Se añade la relación y se muestra un mensaje informando que se añadió satisfactoriamente.	Satisfactoria
EC 2 Adicionar relación falla.	V	I	Se muestra un mensaje indicando que la relación no se adicionó.	Satisfactoria.
	I	V		
	I	I		

SC 2: Editar campo

Id del escenario	Tabla relacionada	Tipo de relación	Id de la relación	Respuesta del Sistema	Resultado de la Prueba
EC1 Editar relación satisfactoriamente.	V	V	V	Se edita la relación correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC 2 Editar relación falla.	V	I	V	Se muestra un mensaje indicando que los datos de la relación no se editaron correctamente.	Satisfactoria.
	I	V	V		
	V	V	I		
	I	I	I		

SC 3: Eliminar relación

Id del escenario	Id de la relación	Respuesta del Sistema	Resultado de la Prueba
EC1 Eliminar relación satisfactoriamente.	V	Se elimina la relación correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC2 Eliminar relación falla.	I	Se muestra un mensaje indicando que la relación no se pudo eliminar de forma satisfactoria.	Satisfactoria

Caso de Pruebas: Gestionar Datos Asociados a un Archivo Multimedia.
Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Adicionar datos asociados a un archivo multimedia	EC 1.1: Adicionar datos asociados a un archivo multimedia satisfactoriamente.	La aplicación muestra un formulario para entrar los datos asociados a un archivo multimedia, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.2: Adicionar datos asociados a un archivo multimedia falla.	La aplicación muestra un formulario para entrar los datos asociados a un archivo multimedia a insertar y una vez enviados por el usuario, le muestra un mensaje informando que estos no son correctos.
SC 2:Editar datos asociados a un archivo multimedia	EC 2.1: Editar datos asociados a un archivo multimedia satisfactoriamente.	La aplicación muestra los datos asociados a un archivo multimedia seleccionados por el usuario, el usuario edita los datos y los envía. Los datos se modifican correctamente y se muestra un mensaje al usuario indicando que la edición fue satisfactoria.
	EC 2.2: Editar datos asociados a un archivo multimedia falla.	La aplicación muestra los datos asociados a un archivo multimedia, el usuario edita los datos y los envía. El sistema muestra un mensaje indicando que los datos no son correctos.
SC 4: Eliminar datos asociados a un archivo multimedia	EC 4.1: Eliminar datos asociados a un archivo multimedia satisfactoriamente	El usuario selecciona los datos asociados a un archivo multimedia a eliminar y la aplicación elimina los datos asociados a un archivo multimedia satisfactoriamente.
	EC 4.2: Eliminar datos asociados a un archivo multimedia falla.	El usuario selecciona los datos asociados a un archivo multimedia a eliminar, la aplicación muestra un mensaje indicando que ocurrió un error durante el proceso de eliminación.

SC 1: Adicionar datos asociados a un archivo multimedia.

Id del escenario	Tabla relacionada	Tipo de relación	Respuesta del Sistema	Resultado de la Prueba
EC1 Adicionar datos asociados a un archivo multimedia satisfactoriamente.	V	V	Se añade los datos asociados a un archivo multimedia y se muestra un mensaje informando que se añadió satisfactoriamente.	Satisfactoria
EC 2 Adicionar relación falla.	V	I	Se muestra un mensaje indicando que los datos asociados a un archivo multimedia no se adicionaron.	Satisfactoria.
	I	V		
	I	I		

SC 2: Editar datos asociados a un archivo multimedia

Id del escenario	Tabla relacionada	Tipo de relación	Id de la relación	Respuesta del Sistema	Resultado de la Prueba
EC1 Editar datos asociados a un archivo multimedia satisfactoriamente.	V	V	V	Se edita los datos asociados a un archivo multimedia correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC 2 Editar datos asociados a un archivo multimedia falla.	V	I	V	Se muestra un mensaje indicando que los datos asociados a un archivo multimedia no se editaron correctamente.	Satisfactoria
	I	V	V		
	V	V	I		
	I	I	I		

SC 3: Eliminar datos asociados a un archivo multimedia

Id del escenario	Id de la DatoAsoc AM.	Respuesta del Sistema	Resultado de la Prueba
EC1 Eliminar datos asociados a un archivo multimedia satisfactoriamente.	V	Se elimina los datos asociados a un archivo multimedia correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactoria
EC 2 Eliminar datos asociados a un archivo multimedia falla.	I	Se muestra un mensaje indicando que los datos asociados a un archivo multimedia no se pudieron eliminar de forma satisfactoria.	Satisfactoria