UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

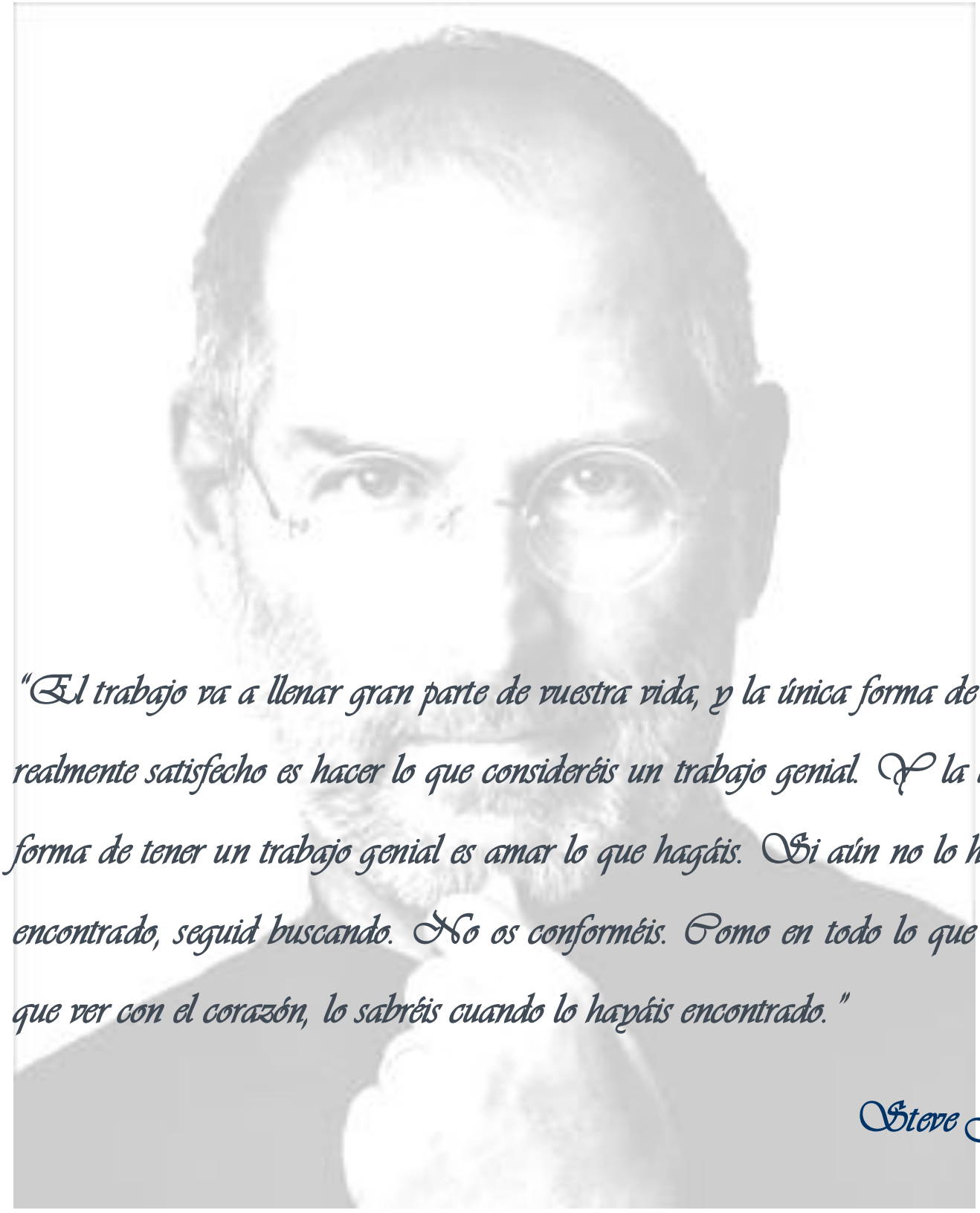
TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Título: Desarrollo del componente de gestión de ficheros para las aplicaciones web del Departamento de Señales Digitales.

Autor (a): Krysna Brenda Sorell Baez

Tutor: Ing. Frank Torres Rodríguez

Ciudad de La Habana, junio de 2011



"El trabajo va a llenar gran parte de vuestra vida, y la única forma de estar realmente satisfecho es hacer lo que consideréis un trabajo genial. Y la única forma de tener un trabajo genial es amar lo que hagáis. Si aún no lo habéis encontrado, seguid buscando. No os conforméis. Como en todo lo que tiene que ver con el corazón, lo sabréis cuando lo hayáis encontrado."

Steve Jobs

Dedicatoria

A Cuba, mi Patria querida.

A mi mamá: “Este es nuestro sueño, lo logramos”.

A mi papá por estar siempre tan presente.

A mi familia, la mejor que hubiera podido tener.

A los que ya no están entre nosotros: mi abuela Rosa y mi tía Margot.

A todos los compañeros de la universidad que me ayudaron a llegar al final y a los que se quedaron en el camino.

Agradecimientos

A la Universidad de las Ciencias Informáticas por existir, por verme crecer, por guardar mis secretos y mis recuerdos, por ser única y regalarme momentos maravillosos.

A mi tutor Frank Torres Rodríguez por su inagotable paciencia, regaños y enseñanzas. Por apoyarme a culminar mis sueños y permitirme aprender a no ser tan tiesa. Por ser el mejor tutor que hubiera podido tener, sin Ti este proyecto nunca hubiera cobrado vida.

A los profesores del tribunal Carlos Enrique, Carlos Luis, Yanisley y el oponente Geovanys por sus aportes y recomendaciones y por hacerme sentir capaz de lograrlo.

A los profesores que han contribuido en mi formación como ingeniera en estos 5 años y que de manera especial me quieren al igual que yo a ellos.

A mi mamá por apoyarme, por secundarme en todo con razón o sin ella, por estar siempre a mi lado, por alimentar mi espíritu, mis ilusiones, mis sueños, por toda la confianza.

A los extraordinarios abuelos que la vida me regaló. Mi abuelo Mario por ser mi guía, mi ejemplo, por mostrarme la vida hecha poesía, por hacerme sentir afortunada. A mi abuela Vicky por escucharme siempre, por todas sus enseñanzas.

A mi familia muchas gracias por simplemente quererlos tanto, por aguantarme y apoyarme: ya somos universitarios.

A mis 3 mosqueteras: Yessy, Lisbet y Yeni por permitirme compartir juntas sueños, locuras, alegrías, lágrimas y hacerme sentir que nunca estaré sola.

A mis queridas Kankas: Lisandra, Ivelin, Saily, Yisell, Yamilet, Glenda, Adaily y Cecilia que llegaron para quedarse siempre.

A mis amigas de la vieja escuela: Yaimelis y Marilyn.

A la familia Amigot por su cariño y a Magalys por tan valiosa ayuda.

A Carli por ser incondicional y por brindarme tanto cariño.

A la profesora más entregada que he conocido Yordanys Piñeiro y el piquete de Cojedes.

A mis hermanos de la FEU: Frank Emilio, Osvaldo, Liuver, Nilo, David y Yidian por las madrugadas, los sacrificios, las alegrías, las peleas, las enseñanzas que no hubiera podido aprender en ningún otro lugar.

A los amigos de mis aventuras artísticas, en especial a Aníbal.

A los amigos de VideoWeb: Ángel, Eridniel, Albrecht, Iván, Ivania entre otros.

Al resto de los amigos de la Universidad, técnicos, instructoras, bedel y todos mis compañeros de las brigadas por las que he pasado estos 5 años (9111, 9210, 9306, 9403, 6511).

A todos **Gracias**

Declaración de Autoría

Declaro que soy la única autora de este trabajo y autorizo al Departamento de Señales Digitales, del centro GEySED de la facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Krysna Brenda Sorell Baez	Frank Torres Rodríguez
_____	_____
Firma del Autor(a)	Firma del Tutor

Datos de Contacto

TUTOR: Ing. Frank Torres Rodríguez

Graduado de Ingeniería en Ciencias Informáticas en Julio de 2007

Dirección: Universidad de las Ciencias Informáticas (UCI), Edificio: 70 Apto: 102

Teléfono Apto: +5378372309 E-mail: ftrodriguez@uci.cu

Síntesis del Tutor

Profesión: Ingeniero en Ciencias Informáticas

Años de graduado: 2

Resumen

En la Universidad de las Ciencias Informáticas (UCI) bajo el modelo de formación estudio – trabajo se desarrollan aplicaciones informáticas. Dada la importancia de la comunicación para Cuba y el mundo, el Departamento de Señales Digitales de la facultad 6 se encarga de desarrollar aplicaciones informáticas en el área del procesamiento de señales (audio y video) e imágenes y reconocimiento de patrones. Los proyectos productivos de innovación y desarrollo con que cuenta el Departamento están en constante trabajo con ficheros multimedia. Los mismos se encuentran en servidores de almacenamiento y es indispensable la manipulación y el acceso a ellos. Actualmente cada proyecto trabaja el manejo de ficheros de diferentes maneras, algunos han implementado pequeños subsistemas que resuelven momentáneamente esta necesidad, otros debido a la fase de desarrollo en la que se encuentran comienzan a preocuparse por como solventar esta problemática. El presente trabajo se realizó con el objetivo fundamental de desarrollar un componente que le permita a los proyectos del Departamento gestionar los ficheros multimedia con los que diariamente trabaja, proveer el acceso a los servidores de almacenamiento donde se encuentran ubicados y facilitar la realización de operaciones con ellos como copiar, renombrar o eliminar archivos. Dicho componente sería de gran utilidad en los proyectos, se ahorraría tiempo y esfuerzo en el desarrollo de futuras aplicaciones, y contribuiría al repositorio de componentes del Departamento. Para materializar la idea anteriormente expresada fue llevada a cabo esta investigación, luego de un estudio detallado de las tendencias y tecnológicas actuales, fue seleccionado el Proceso Unificado de Desarrollo (RUP por sus siglas en inglés) como metodología de desarrollo, Lenguaje Unificado de Modelado (UML por sus siglas en inglés) como lenguaje de modelado, PHP como lenguaje de programación y PostgreSQL como gestor de base de datos. Se asumió como principal premisa la de lograr la libertad tecnológica del producto, teniendo siempre en cuenta el soporte sobre diferentes sistemas operativos y el cumplimiento con los principios de la comunidad de software libre.

PALABRAS CLAVES

componente, protocolo, servidor, almacenamiento, FTP.

Índice de Figuras

Figura 1: Modelo FTP 11
Figura 2: Diagramas UML 21
Figura 3: Modelo de dominio 30
Figura 4: Diagrama de Casos de Uso del Sistema 36
Figura 5: Diagrama de clase del análisis. Caso de Uso Gestionar almacenamiento 42
Figura 6: Diagrama de clase del Análisis. Caso de Uso Gestionar fichero. 43
Figura 7: Diagrama de clase del Análisis. Caso de Uso Gestionar carpeta. 43
Figura 8: Patrón Arquitectónico Modelo Vista Controlador 47
Figura 9: Diagrama de clases del diseño. 51
Figura 10: Diagrama de Secuencia CU "Gestionar almacenamiento", Sección Adicionar almacenamiento. 52
Figura 11: Modelo Entidad-Relación 53
Figura 12: Diagrama de componentes 56
Figura 13: Diagrama de despliegue 57

Índice de Tablas

Tabla 1: Descripción del flujo de sucesos del CU "Gestionar almacenamiento". 40
Tabla 2: Caso de prueba para el CU: "Gestionar almacenamiento" 60
Tabla 3: Caso de prueba para el CU: "Gestionar almacenamiento". Descripción de las variables. 62
Tabla 4: Caso de Prueba para el CU: "Gestionar almacenamiento", Sección Adicionar almacenamiento. Matriz de datos 65

Índice

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA 7

1.1 INTRODUCCIÓN 7
1.2 ESTADO DEL CONOCIMIENTO 7
1.3 CONCEPTOS ASOCIADOS AL DOMINIO DEL PROBLEMA 8
1.4 TENDENCIAS Y TECNOLOGÍAS ACTUALES 16
1.5 CONCLUSIONES PARCIALES 25

CAPÍTULO 2: CARACTERÍSTICAS DE LA SOLUCIÓN PROPUESTA 26

2.1 INTRODUCCIÓN 26
2.2. MODELAMIENTO DEL NEGOCIO 26
2.2.1 Modelo de dominio 27
2.2.2 Eventos principales del entorno 27
2.2.3 Glosario de términos del dominio 28
2.2.4 Diagrama de clases del modelo de dominio 29
2.4 DESCRIPCIÓN DEL SISTEMA PROPUESTO 34
2.5 CONCLUSIONES PARCIALES 40

CAPÍTULO 3: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA 41

3.1 INTRODUCCIÓN 41
3.2 DISCIPLINA DE ANÁLISIS 41
3.2.1 Modelo del análisis 41
3.3 DISCIPLINA DE DISEÑO 44
3.3.1 Arquitectura de Software 44
3.3.2 Estilos arquitectónicos. 45
3.3.3 Patrones de Diseño 48
3.3.4 Modelo de Diseño 50
3.3.5 Diagramas de interacción 52
3.3.6 Diseño de la Base de Datos 53
3.4 CONCLUSIONES PARCIALES 54

CAPÍTULO 4: CARACTERÍSTICAS DE LA SOLUCIÓN PROPUESTA 55

4.1 INTRODUCCIÓN 55

4.2 DISCIPLINA DE IMPLEMENTACIÓN 55

4.2.1 Diagrama de componentes 55

4.2.2 Diagrama de despliegue 56

4.3 DISCIPLINA DE PRUEBA 58

4.3.1 Pruebas de Caja Negra 58

4.3.2 Resultados de las Pruebas 65

4.4 CONCLUSIONES PARCIALES 66

CONCLUSIONES 67

RECOMENDACIONES 68

BIBLIOGRAFÍA CITADA 69

BIBLIOGRAFÍA CONSULTADA 71

ANEXOS 72

INTRODUCCIÓN

La comunicación es un factor fundamental para el funcionamiento de las sociedades humanas, además de transmitir información, atribuye conductas. Los distintos medios de comunicación que existen desde su origen han sido protagonistas en los grandes sucesos acontecidos en todo el planeta. Su uso ha influenciado el desarrollo de la educación, la cultura, el deporte, y la medicina, entre otras esferas de la sociedad. La tecnología juega hoy un rol de suma importancia para la vida del hombre, el cual ha aprendido a utilizarla en su beneficio en una amplia gama de actividades.

El conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de ordenadores (REA, 2010), conocido mundialmente con el término informática es un ejemplo representativo del auge de los adelantos científicos y tecnológicos. En la actualidad se han logrado avances significativos en esta ciencia, con un alto grado de calidad en los procesos que desencadenan la creación de un producto informático. Cuba no está exenta de este progreso que han tenido las tecnologías de la información y las comunicaciones (TIC). Por tal motivo la dirección del país, consciente de la facilidad que proporcionan las TIC para elevar el avance de la sociedad en general**,** no solo prioriza y promueve su informatización, para lo cual dedica innumerables esfuerzos; sino que aboga por el desarrollo de soluciones informáticas encaminadas a resolver diversos problemas.

Como parte de las TIC, las telecomunicaciones, se han convertido en uno de los sucesos más importantes de las últimas décadas. Dentro de la misma se pueden encontrar variadas formas de comunicación a distancia, destacándose entre estas la televisión. En la actualidad ha experimentado un gran auge tecnológico, insertándose en disímiles prácticas populares, por lo que constituye un medio del cual pocas personas pueden prescindir. Debido a la alta demanda que tienen estos medios audiovisuales como parte de la sociedad, en todo el mundo ha aumentado la creación de productos informáticos destinados a facilitar el trabajo relacionado con los mismos.

La Universidad de la Ciencias Informáticas (UCI) surgida en Cuba al calor de la batalla de ideas, constituye un pilar fundamental para desarrollar la informática en el país. Haciendo uso de las bondades de las TIC, se desarrollan en la institución soluciones integrales de software y se brindan diversos servicios. El objetivo principal de la UCI es desarrollar productos informáticos para satisfacer la demanda

de Cuba o a partir de compromisos establecidos en el mercado internacional. Para dar cumplimiento a este objetivo la UCI cuenta con una serie de centros de desarrollo compuestos por Departamentos según el área temática de los productos que desarrollan. En la Facultad 6 se encuentra el centro GEySED (Geoinformática y Señales Digitales), el cual está compuesto por dos departamentos. Uno de ellos es el Departamento de Señales Digitales, departamento productivo de excelencia en la formación de profesionales comprometidos con su Patria y la Revolución. A través de la actividad docente – científico – productiva en el área del procesamiento de señales (audio y video) e imágenes y reconocimiento de patrones lleva a cabos importantes compromisos productivos nacionales e internacionales. El mismo ofrece soluciones, productos y servicios para la automatización, transmisión y gestión en el área de procesamiento digital de imágenes y señales.

Hasta la fecha (junio de 2011) el Departamento de Señales Digitales cuenta con 7 proyectos en ejecución. Como parte de las actividades que los proyectos realizan, principalmente basadas en el trabajo con archivos y ficheros multimedia, estos se encuentran en constante manipulación y movimiento. Un ejemplo representativo de lo anterior constituye la Plataforma VideoWeb, la cual es un sistema web[1] encaminado a la publicación y administración de archivos multimedia, basándose en la tecnología de streaming[2] para brindar estos contenidos audiovisuales a través de la red de datos. Con la utilización de esta aplicación se manipulan y guardan con frecuencia archivos multimedia en un servidor de datos, los cuales se utilizan después para ser publicados en la plataforma.

Por la importancia de esta gestión se hace ineludible destacar el hecho de que no existe en el departamento un estándar para hacer que la transferencia tenga lugar, sino que cada proyecto ha implementado o implementará una manera para que el movimiento de archivos y ficheros multimedia ocurra. Esto provoca que se destine tiempo y esfuerzo extra a implementar funcionalidades que satisfagan estas necesidades que son denominador común para la mayoría de los proyectos, es posible entonces, advertir la existencia de una **situación problémica**.

[1] Un sistema web es un sistema que apoya parte de sus procesos a través de una red de computadoras o la Word Wide Web.

[2] La tecnología del streaming media posibilita la distribución en directo y en diferido de audio, video y multimedia en Internet.

Introducción

En aras de mejorar la gestión de esta información en los proyectos del Departamento de Señales Digitales y teniendo en cuenta la situación anteriormente señalada, se ha identificado como **problema a resolver** ¿Cómo facilitar la gestión de ficheros en servidores de almacenamiento empleando diferentes protocolos de transferencia en las aplicaciones web del Departamento de Señales Digitales?

Teniendo en cuenta lo planteado con anterioridad, fue definido como **objeto de estudio** el proceso de gestión de ficheros en servidores de almacenamiento, estableciendo el **campo de acción** en la gestión de ficheros en servidores de almacenamiento de los proyectos del Departamento de Señales Digitales de la Facultad 6 de la Universidad de las Ciencias Informáticas. Para dar solución al problema fue trazado como **objetivo general**: desarrollar un componente de software que permita gestionar ficheros en servidores de almacenamiento de ficheros empleando diferentes protocolos de transferencia para las aplicaciones web del Departamento de Señales Digitales.

Como **idea a defender** se ha establecido: El desarrollo de un componente para la gestión de ficheros en servidores de almacenamiento empleando diferentes protocolos de transferencia facilitará la manipulación de ficheros en las aplicaciones del Departamento de Señales Digitales.

Para lograr el cumplimiento del objetivo de este trabajo de diploma se ha propuesto desarrollar las siguientes **tareas de investigación**:

- Analizar las características de los principales tipos de servidores de almacenamiento.
- Definir y caracterizar los procesos que se realizan para la manipulación de ficheros en los servidores de almacenamiento.
- Definir y argumentar la metodología de desarrollo de software a usar en el proceso.
- Identificar las funcionalidades que debe brindar el componente.
- Definir las herramientas y tecnologías a utilizar en la construcción de la solución.
- Realizar el modelado de los artefactos necesarios para la implementación del componente.
- Implementar el núcleo del componente.
- Implementar la transferencia de fichero empleando almacenamiento local.
- Implementar la transferencia de fichero empleando el protocolo ftp.
- Realizar las pruebas al componente.

- Realizar manual de instalación y uso del componente.

Uno de los primeros pasos para la resolución de este problema fue la realización de un estudio, con el objetivo de encontrar y analizar algunos **antecedentes** a esta investigación. Como parte de esta pesquisa fue posible detectar que en el Departamento de Señales Digitales, el proyecto VideoWeb se encuentra en proceso de implementación de unas clases para lograr la transferencia de archivos multimedia. En el caso del proyecto Captura y Catalogación de Medias, aún no se ha llegado a la fase de desarrollo en la que se incluye la resolución de este problema, pero en entrevista con sus líderes la propuesta existente es la implementación de una solución similar a la que desarrolló VideoWeb. En el caso de otros proyectos que están en una fase inicial de desarrollo, este es un problema para el que todavía no se cuenta con una solución.

El principal resultado esperado de este trabajo es un componente que permita gestionar ficheros en los servidores de almacenamiento de los proyectos del Departamento, los cuales ganarán en eficacia en su accionar diario. Con la simple incorporación a las aplicaciones de este componente se ahorrará tiempo y esfuerzo en el desarrollo de las mismas y se contribuirá al repositorio de componentes del Departamento.

Para entender más a fondo las características del proceso de gestión de ficheros multimedia fueron empleados diferentes **métodos científicos de investigación,** tanto teóricos como empíricos.

Los métodos teóricos utilizados durante la investigación fueron el analítico - sintético, el análisis histórico – lógico, el inductivo – deductivo y la modelación.

El método **analítico - sintético** fue usado con el objetivo de llegar a conocer, mediante el análisis de las diferentes teorías y documentos encontrados a lo largo de la presente investigación, la esencia de los fenómenos relacionados con el proceso de gestión de ficheros multimedia, así como los rasgos que lo caracterizan y distinguen.

El **análisis histórico - lógico** se empleó para realizar un estudio analítico de la trayectoria histórica del objeto de estudio de la presente investigación y de cómo este ha influido en el desarrollo de soluciones informáticas.

Con el método **inductivo - deductivo** se hizo posible arribar a conocimientos generalizadores del proceso de gestión de ficheros multimedia, a partir de los aspectos particulares y generales encontrados en los documentos involucrados en la presente indagación, analizándolos tanto de lo particular a lo general como viceversa.

La **modelación** es un método que permite la creación de modelos, que no son más que una reproducción simplificada de la realidad y que proporciona una mejor comprensión de las características del objeto de estudio y el descubrimiento de nuevas relaciones.

Durante la investigación se utilizaron además dos de los **métodos empíricos**. El primero de ellos es la **entrevista**, empleada con el objetivo de obtener información valiosa y adquirir una mayor familiarización con respecto a las peculiaridades y características del proceso de gestión de ficheros multimedia en el Departamento de Señales Digitales. El otro método aplicado es el **experimento**, que sirvió para demostrar, mediante pruebas en las que se modifica de forma controlada las condiciones en las que funciona el sistema, la eficacia de las funcionalidades del mismo.

El contenido del presente trabajo de diploma está estructurado de la siguiente manera:

Capítulo 1. Fundamentación teórica: Se expone el estado del arte del objeto de estudio de la presente investigación y se definen los elementos teóricos que lo sustentan. Se enuncian conceptos que posibilitan un mejor entendimiento de lo planteado en la situación problemática y el marco del problema en sentido general. Se enuncian y argumentan otras aplicaciones ya existentes que pueden dar solución de alguna manera al problema científico del presente trabajo.

Capítulo 2. Características de la solución propuesta: Se comienza la investigación del medio donde se implementará la aplicación con el objetivo de definir las condiciones o capacidades que debe ofrecer el componente. Se abordan los principales aspectos de los flujos de trabajo Modelamiento del Negocio y Requerimientos. Se conceptualiza el entorno mediante un modelo de dominio, se especifican los requisitos funcionales y no funcionales que deberá cumplir la solución propuesta, se presentan los Diagramas de Casos de Uso del Sistema, se describen los actores y se detallan los Casos de Uso del Sistema.

Capítulo 3. Análisis y Diseño de la solución propuesta: El propósito de este capítulo es dejar expuestos los principales artefactos generados durante los flujos de trabajo Análisis y Diseño. Se define así la estructura del diseño de la aplicación desarrollada, y para ello será representado el diagrama de clases del diseño, el diagrama de clases persistentes y el modelo Entidad-Relación, así como el modelo de datos del componente.

Capítulo 4. Implementación y pruebas de la solución propuesta: La intención de este capítulo es dejar expuestos los principales artefactos generados durante los flujos de trabajo Implementación y Prueba. Con la implementación se comenzó con el resultado del diseño para implementar el sistema en términos de componentes, ficheros de código fuente, scripts, ejecutables y similares. Mediante la disciplina Prueba se enfocó la investigación en la evaluación y determinación de la calidad del producto.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

El dominio de los conceptos, definiciones y temas fundamentales referidos a la gestión de ficheros en servidores de almacenamiento es indispensable para comprender correctamente en qué consiste la presente investigación. En el mismo se exponen los conceptos y elementos que constituyen la base teórica para la realización del presente trabajo de diploma y lógicamente para dar solución al problema a resolver planteado.

Además, en este capítulo queda plasmado el marco teórico de la investigación, o sea, toda la teoría existente del objeto de estudio que ha surgido como resultado de anteriores investigaciones y que a su vez permite orientar el trabajo de una manera coherente. También se hace referencia a las técnicas, tecnologías y metodologías en las que se apoya la construcción del componente para la solución del problema que se enfrenta.

1.2 Estado del conocimiento

La gestión de ficheros es un proceso necesario que se utiliza en muchas aplicaciones que almacenan datos. En ocasiones se reduce a un conjunto de clases o a un módulo que permite la conexión al servidor de almacenamiento, almacenar los ficheros y manipularlos de acuerdo a las necesidades del negocio.

En la actualidad existen en el mercado varias soluciones profesionales para el tratamiento de archivos multimedia, específicamente para la gestión de estos. Algunos de los programas más interesantes creados con este fin son:

- Alfa Media Partner. Su solución se denomina alfa Mediastore basada en una base de datos Oracle, que permite la gestión y archivo tanto de imágenes, texto, sonido y vídeos.
- Artesia Technologies desarrolló una aplicación llamada TEAMS para la gestión de archivos multimedia.

- Digital Collections. DC4 es el sistema desarrollado por esta empresa alemana. Esta solución abarca todos los componentes de la organización, búsqueda y procesamiento de información digital en forma de archivos de texto, imagen, gráfico, sonido y vídeo.
- Pidcar. La empresa inglesa ofrece su gestor Media Mogul del que afirma que soporta y alimenta diversos flujos de trabajo (documentalistaenredado.net).

Estas soluciones van más allá de la necesidad del Departamento de Señales Digitales del centro GEySED, ya que son más que componentes, aplicaciones informáticas que incluyen variadas funcionalidades que además de gestionar solventan otras necesidades. Estas aplicaciones son producto de grandes compañías mundiales y no se conoce la forma en que fueron creadas.

Algunos de los 7 proyectos en ejecución con los que cuenta el departamento han implementado soluciones que aunque no son del todo similares a las mencionadas anteriormente, resuelven la funcionalidad fundamental de estas aplicaciones: la gestión de ficheros multimedia. Tal es el caso del proyecto que desarrolla la Plataforma VideoWeb. El equipo de desarrollo de esta aplicación está en fase de implementación de un módulo que permite la gestión de ficheros a través del protocolo FTP, el cual abstrae la transferencia de archivos de los demás módulos.

Existe otro proyecto llamado PETARTV que implementó una solución para la transferencia de archivos pero esta al ser una aplicación de escritorio no entra en el marco de esta investigación. El proyecto Captura y Catalogación de Medias entre otros proyectos del departamento necesita una solución para resolver el problema de la gestión de ficheros en sus servidores de almacenamiento. El desarrollo de este componente sería de gran utilidad para estos proyectos y más aún para los proyectos futuros que puedan surgir en el departamento, los cuales no tendrán que preocuparse por implementar una solución para transferir sus ficheros.

1.3 Conceptos asociados al dominio del problema

Para un mayor acercamiento a las características del proceso de gestión de ficheros en servidores de almacenamiento es imprescindible enunciar varios conceptos asociados a la temática de la presente investigación.

1.3.1 Componente de software

En los últimos años el creciente desarrollo de la industria del software ha estado orientado a reducir considerablemente el costo y tiempo de desarrollo de sistemas y aplicaciones sin comprometer los niveles de calidad del producto. Debido a esta particularidad ha surgido un nuevo elemento reutilizable denominado componente de software.

El Departamento de Señales Digitales pretendiendo lograr lo anteriormente planteado reveló la necesidad de construir componentes de software que tributen al eficiente desarrollo de las aplicaciones existentes en el departamento y que además sean reutilizados en la creación de futuras aplicaciones. Por este motivo se propone un componente de software como la solución de la presente investigación. Mundialmente se han propuesto numerosas definiciones, entre las cuales se destacan las siguientes:

Según Philippe Kruchten de Rational Rose, un componente “es una parte no trivial, casi independiente y reemplazable de un sistema que cumple una función dentro del contexto de una arquitectura bien definida. Un componente cumple con un conjunto de interfaces y provee la realización física de ellas” (C.Wallnau., 1998). Clemens Szyperski define un componente de software como “es una unidad de composición con interfaces especificadas contractualmente y solamente dependencias explícitas de contexto. Un componente de software puede ser desplegado independientemente y está sujeto a composición por terceros” (Szyperski., 2002). Según Peter Herzum y Oliver Sims, “un componente es un artefacto de software auto contenido y claramente identificable que describe ejecuta funciones específicas; que tiene, además, una interfaz claramente establecida, una documentación apropiada y un *status* de uso recurrente bien definido” (Sims_Herzum, 2000).

Recientemente, el Instituto de Ingeniería de Software (SEI, *Software Engineering Institute*) de la universidad Carnegie-Mellon en Pensilvania enunció una definición con el propósito de consolidar las diferentes opiniones acerca de lo que debía ser un componente de software. Según el SEI un componente es “una implementación opaca de funcionalidad, sujeta a composición por terceros y que cumple con un modelo de componentes” (Bachmann, et al., 2000).

De manera general se arriba a la conclusión de que un componente de software es un artefacto de software creado para integrarse a otras aplicaciones, proporcionándole a estas la ejecución de funciones específicas. Debe ser reemplazable, prácticamente independiente de estas aplicaciones y estar bien documentado. El componente fruto de esta investigación tiene como parte de su objetivo gestionar ficheros en servidores de almacenamiento por lo que es de gran importancia conocer con exactitud el concepto de fichero.

1.3.2. Fichero

El Diccionario de la Real Academia de la Lengua Española (REA) dice que un fichero es un "conjunto organizado de informaciones almacenadas en un soporte común"(REA, 2010). Aunque existen varios criterios sobre la diferencia de este concepto con la definición de archivo, en el campo de la informática en ocasiones se consideran sinónimos, a pesar que según la definición que da la REA un archivo es un "espacio que se reserva en el dispositivo de memoria de un computador para almacenar porciones de información que tienen la misma estructura y que pueden manejarse mediante una instrucción única" (REA, 2010). En el marco de esta investigación se usarán análogamente los dos términos. A grandes rasgos se puede exponer que la información que se guarda en una computadora se denomina ficheros.

Los ficheros serán gestionados por el componente en los servidores de almacenamiento sin importar el protocolo que estos utilizan, por lo que es importante definir que es un protocolo y los principales protocolos que se utilizan con estos fines.

1.3.3 Principales protocolos usados por los servidores de almacenamiento de ficheros

Un protocolo no es más que un conjunto de reglas o normas creadas para llevar a cabo una actividad en específico. Existen diversos tipos de protocolos tales como el protocolo de investigación, el protocolo a seguir en la celebración de ceremonias oficiales, el protocolo de comunicación entre otros. En el campo de la informática es un "conjunto de reglas que definen el intercambio de datos entre computadoras" (UH, 2010). Cada protocolo se encarga de una forma específica de intercambio, por ejemplo el protocolo SMTP de la familia de protocolos TCP/IP, se encarga de la recepción de mensajes de correo entrante.

Un protocolo importante de esta familia y probablemente el más conocido dentro de la transferencia de ficheros es el **FTP** (File Transfer Protocol), el cual permite el intercambio bidireccional de ficheros y manejo de directorios entre dos computadoras en la red (UH, 2010). Las aplicaciones que originan estas operaciones usando el protocolo FTP se conocen como clientes FTP y son similares a los clientes web como los navegadores, estos también intercambian información pero usando el protocolo HTTP.

FTP tiene como objetivo "promocionar el uso compartido de ficheros (programas y/o datos), animar al uso indirecto o implícito (a través de programas) de servidores remotos, hacer transparente al usuario las variaciones entre la forma de almacenar ficheros en diferentes ordenadores, y transferir datos fiable y eficientemente" (Reynolds, 2000). Puede ser usado para cargar y descargar archivos de casi cualquier tamaño desde o hacia un servidor central.

Este protocolo tiene dos ventajas muy importantes, cuenta con un soporte para todos los tipos de cliente; gracias a la implementación estandarizada de este protocolo prácticamente cualquier cliente FTP puede utilizar el servidor FTP aunque se ejecute en un sistema operativo de Microsoft o no. También posee un alto rendimiento y sencillez, esta característica lo hace una favorable opción para la transferencia de archivos a través de Internet.

El protocolo FTP está basado en el modelo cliente-servidor, es decir, un equipo denominado servidor espera las solicitudes que le envía otro equipo denominado cliente para proceder a realizar las operaciones requeridas.

Durante una conexión FTP se encuentran abiertos dos canales de comunicación:

- Un canal de comandos (canal de control)
- Un canal de datos

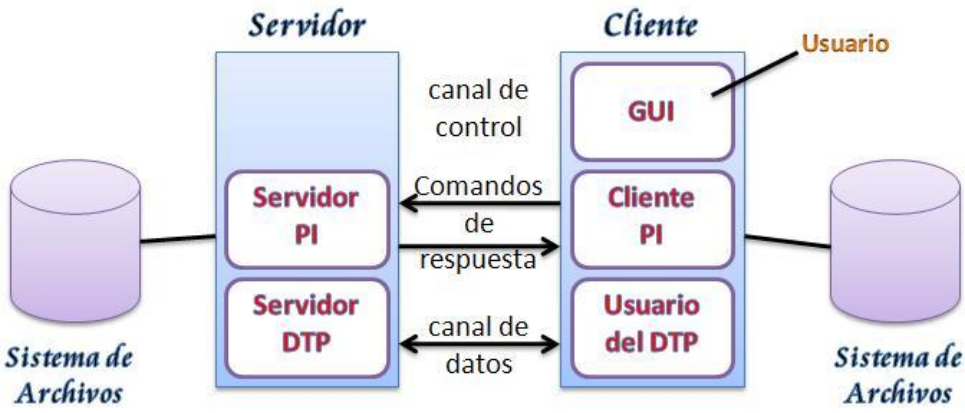


Figura 1: Modelo FTP

El cliente y el servidor cuentan con dos procesos que permiten la administración de estos dos tipos de información:

DTP (Proceso de transmisión de datos): Es el proceso encargado de establecer la conexión y de administrar el canal de datos. El DTP del lado del servidor se denomina Servidor de DTP y el DTP del lado del cliente se denomina Usuario de DTP

PI (Interprete de Protocolo): Interpreta el protocolo y permite que el DTP pueda ser controlado mediante los comandos recibidos a través del canal de control (Yerovi, et al., 2010).

La principal desventaja de este protocolo es que la información de inicio de sesión se envía sin encriptación a través de la red. Esto puede provocar que personas no autorizadas descubran las cuentas o contraseñas de inicio de sesión y la utilicen para acceder a otros sistemas (Microsoft, 2010). Con la utilización de aplicaciones como SCP (Secure Copy) y SFTP (Secure File Transfer Protocol) usando el protocolo SSH, que permiten transferir archivos bajo cifrado, se le da solución a este problema.

"SSH o Secure Shell es tanto una aplicación como un protocolo que permite conectar dos ordenadores a través de una red, ejecutar comandos de manera remota y mover ficheros entre los mismos" (Andrade, 2009). Todas las comunicaciones entre clientes y servidores están aseguradas bajo encriptación y protegidas de modificaciones, ya que SSH proporciona autenticación fuerte y comunicaciones seguras sobre canales no seguros. La mayor importancia que tiene este protocolo es que aporta seguridad a la red ya que todos los datos enviados desde un ordenador a la red son encriptados y desencriptados automáticamente sin que el usuario tenga conocimiento de ello. Existen dos versiones SSH (SSH1 y SSH2) con diferentes funcionalidades (SSH2 soluciona problemas de seguridad de SSH1), a pesar de esto existen razones para usar una o la otra como por ejemplo SSH1 tiene mejor rendimiento que SSH2. Este protocolo pretende ser un reemplazo seguro para aplicaciones tradicionales no seguras, como telnet, rlogin, rsh y rcp. Su segunda versión proporciona también sftp, un reemplazo seguro para FTP (Aspuru, 2008).

Su arquitectura está basada en el modelo Cliente-Servidor como se muestra en la figura 2 (Consultar anexos), donde existe un programa servidor SSH que acepta o rechaza peticiones de conexiones entrantes y otro programas SSH clientes, que realizan solicitudes a otra máquina que alberga al SSH servidor. Estas solicitudes realizan un proceso inicial de autenticación, principalmente de "login" (Conectarse a un ordenador mediante una identificación de usuario y contraseña) para acceder a una máquina, transferir archivos o ejecutar comandos de forma remota, o simplemente manipular una información específica (JBarret, et al., 2001).

SSH utiliza diferentes algoritmos de cifrado según su versión DES, TripleDES, IDEA, Blowfish para SSH1 y TripleDES, Blowfish, Twofish, Arcfour, Cast128-cbc en SSH2. El cifrado utilizado para cuestiones de autentificación es en SSH1 mediante el algoritmo RSA y DSA para SSH2. También permite autenticarse usando varios métodos como password, sistema de clave pública, kerberos (SSH1) y basado en el cliente (por relaciones de confianza en SSH1 y sistema de clave pública en SSH2).

SSH protege contra varios tipos de ataques como IP Spoofing, intercepción de contraseñas y datos a través de la red, manipulación de los datos en ordenadores intermediarios entre otros, implementado la política de que la red es un medio adverso en el que no se puede confiar. SSH podrá ser forzado a desconectarse pero el atacante no podrá descifrar los datos o introducirse en la conexión. Aunque SSH permite no usar cifrado (cifrado de tipo "none") esta opción no debería utilizarse ya que todo lo planteado anteriormente es posible si se utiliza algún tipo de cifrado. Hay que tener en cuenta que SSH no sirve de nada si el atacante consigue hacerse con el control de la máquina como root, ya que podrá hacer que SSH quede inutilizado (Aspuru, 2008).

SMB (Server Message Block) es el protocolo nativo de uso compartido de archivo e impresoras para ordenadores con sistemas operativos Windows (Microsoft, 2010). Aunque este protocolo fue originalmente inventado por la IBM, en la actualidad la versión más común es la considerablemente modificada de Microsoft, compañía que en 1998 renombró a este protocolo como CIFS (Common Internet File System) añadiéndole nuevas características.

Cuenta con numerosas ventajas como son:

- Mejor soporte para uso compartido de archivos: Los protocolos nativos de uso compartido de archivos de Microsoft proporcionan el mejor nivel de soporte para el uso compartido de archivos entre los clientes Microsoft y los servidores de archivo Microsoft.
- Acceso desde una amplia gama de clientes CIFS: Server Message Block habilita el acceso a los usos compartidos de datos desde una amplia variedad de clientes CIFS, que incluye una amplia gama de sistemas operativos cliente que no son de Microsoft.
- Mayor seguridad: Los dos niveles de seguridad para la autenticación, uso compartido y usuarios de Server Message Block, son bien aceptados y proporcionan una mayor seguridad

La principal ventaja que tiene SMB es que no cuenta con encriptación de la capa de transporte. El protocolo de túnel de Internet Protocol Security (IPSec) se debe utilizar para proteger los datos del archivo en tránsito por la red (Microsoft, 2010).

Samba es una implementación libre del protocolo SMB de Microsoft, en forma de suite de aplicaciones basada en la implementación de una docena de servicios y protocolos. Fue desarrollada por el Dr. Andrew Tridgell especialmente para ordenadores con sistemas operativos tipo Unix (Samba, 2010). La característica fundamental de Samba es que permite que ordenadores con sistema operativo Unix puedan comunicarse con el mismo protocolo de red que Microsoft Windows y aparecer como un sistema más de Windows en la red. Además cuenta con herramientas que posibilita a los usuarios de un sistema Unix acceder a los directorios e impresoras que los sistemas Windows y servidores Samba comparten en la red. Samba configura directorios Unix como recursos para compartir en la red, los cuales son vistos por usuarios Windows como carpetas normales de la red. Estas unidades de red pueden ser montadas en los sistemas de archivos de usuarios Linux como si fueran dispositivos locales o utilizar uno de los programas que incluye Samba, el smbclient que permite conectarse a los recursos compartidos SMB y operar con ellos muy similar a un cliente FTP.

Los laboratorios IT Week desarrollaron algunas pruebas que llegaron a la conclusión que la última versión de Samba le lleva ventaja a Windows en cuanto a rapidez en la compartición de archivos e impresoras. Hubo pruebas que demostraron que Samba podía soportar hasta cuatro veces más clientes que Windows 2000 antes de que su funcionamiento comenzara a disminuir. Una reciente prueba con 48 ordenadores como clientes simulando la carga de trabajo de 500 clientes en un ambiente de trabajo común, demostró

el buen funcionamiento del servidor de archivos de Samba (Samba, 2010).La última versión incluye soporte para la autenticación mediante Kerberos 5 y LDAP, novedad imprescindible para funcionar como un cliente en un dominio Active Directory. Otra nueva característica es el soporte de Unicode, lo que simplificará mucho el soporte de lenguajes internacionales (cad.com.mx, 2010).

Un protocolo es sin duda una pieza fundamental en el proceso de la comunicación mediante la red, que está definido como un conjunto de reglas para establecer cómo se llevará a cabo el intercambio de información entre dos computadoras a través de una red. Una de estas computadoras juega un papel fundamental en la transferencia, la cual se conoce como servidor.

1.3.4. Servidor

En la ciencia de la informática existen diferentes conceptos de servidores, pero en esencia se puede decir que es un ordenador que forma parte de una red, en ocasiones con potentes condiciones de hardware, aunque no siempre tiene que ser así, el cual proporciona servicios a otros ordenadores conocidos como clientes. También puede ser un proceso que proporciona información o sirve a otro proceso.

Un servidor es un componente de software que se encargan de realizar una o varias tareas en función de terceros, generalmente en el contexto de una red. Los componentes de software (y las computadoras en que estos se ejecutan) que se sirven o hacen uso de las funciones que llevan a cabo los servidores, se denomina clientes. Los servidores se ocupan de gestionar el tráfico en la red, manejar la identificación de cada una de las estaciones de trabajo que forman parte de esta y brindar una gran cantidad de servicios a los clientes, como acceso a ficheros e Internet, correo electrónico, etc. (Reynolds, 2000). Un servidor es un medio permanente y eficiente de acceder al conocimiento, de comunicación e interacción entre usuarios y publicación de sus ideas, experiencias y trabajos(Andrade, 2009).Los servidores que se encargan específicamente de proporcionar contenido web mediante el protocolo HTTP, a los clientes web generalmente exploradores o navegadores se son denominados **Servidores Web**.

Existen diferentes tipos de servidores tales como servidor de archivo, web, de impresiones, de correo, proxy, de acceso remoto, de base de datos entre otros. El servidor de archivo es un servidor de almacenamiento que contiene diferentes tipos de archivo y los comparte con otros clientes de la red.

1.3.5. Procesos que se realizan para la manipulación de ficheros en los servidores de almacenamiento

En los servidores de almacenamiento u otro sistema informático los ficheros tienen nombres y son ubicados en directorios. El nombre de cada fichero debe ser único, no pueden existir dos ficheros con el mismo nombre en una misma ubicación, ya que el nombre del fichero y la ruta a su ubicación lo diferencia entre el resto de los ficheros que se encuentran almacenados en el servidor.

Luego de un fichero ser subido al servidor donde se quiera almacenar, este quedará accesible para las aplicaciones o usuarios que posteriormente lo utilizarán y se puede realizar numerosas operaciones con él, así como organizarlos y manipularlos según la necesidad del usuario. Los principales procesos que se llevan a cabo en un servidor de almacenamiento sobre un fichero son: copiar, eliminar y renombrar, las cuales se explicarán a continuación:

Copiar: Un fichero puede ser copiado desde una computadora a cualquier directorio del servidor de almacenamiento especificado en la aplicación. También puede copiarse de una ubicación a otra dentro del propio servidor.

Eliminar: Con esta operación se borra totalmente el fichero deseado, dejando libre el espacio del soporte que ocupaba.

Renombrar: Esta operación permite cambiar el nombre del fichero.

El componente que gestione los ficheros para las aplicaciones web del Departamento de Señales Digitales debe ser capaz de realizar todas estas operaciones, sin importar el protocolo que estos utilizan para comunicarse.

1.4 Tendencias y tecnologías actuales

Las facilidades que en la actualidad brinda el desarrollo de la tecnología hacen que grandes y pequeñas empresas puedan dar soluciones informáticas efectivas a sus necesidades. Por ello, en medio de una

amplia gama de opciones se hizo necesario realizar un análisis de las tendencias y tecnologías actuales a considerar para ser usadas en el desarrollo del componente.

Con el fin de adoptar las mejores opciones para cumplir el objetivo general de este trabajo de diploma fueron analizadas las metodologías de desarrollo de software, los gestores de base de datos más usados en el mundo, algunos lenguajes de programación web y otros aspectos referentes a la tecnologías actuales que son de gran importancia en esta toma de decisiones. Luego del análisis de estos elementos, se estableció cuáles serán empleados para la construcción de la solución propuesta, se hizo una exposición y fundamentación de la tecnología escogida para modelar, implementar y garantizar el correcto funcionamiento del componente que gestione los ficheros multimedia en los servidores del Departamento de Señales Digitales.

1.4.1 Metodologías de desarrollo de software

La necesidad de que la forma de trabajo sea organizada, es un principio inviolable a la hora de desarrollar cualquier software informático. Debe contarse con un proceso que ordene e integre las múltiples etapas del desarrollo. Es importante tener una guía para organizar las actividades del equipo de trabajo, así como una estrategia para ordenar las tareas que debe acometer el mismo y establecer los artefactos que deben ser producidos para lograr los objetivos propuestos. Por todo esto es necesario definir una metodología de desarrollo de software, que no es más que un conjunto de procedimientos y pasos que deben ser seguidos para desarrollar un software. Entre las más utilizadas se encuentran la Programación Extrema (XP), Scrum y el Proceso Unificado de Software (RUP). Luego de un exhaustivo análisis de estas metodologías se deja plasmado en este epígrafe algunas características y ventajas de RUP, que fue la escogida para utilizar en la construcción de la solución propuesta.

1.4.1.1 RUP como base en el desarrollo de la solución propuesta

El Proceso Unificado de Desarrollo de Software (Rational Unified Process) conocido como RUP (siglas en inglés), es una metodología de software que permite el desarrollo de aplicaciones a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad, aunque con el inconveniente de generar mayor complejidad en los controles de administración

del mismo. Sin embargo, los beneficios obtenidos recompensan el esfuerzo invertido en este aspecto. El proceso de desarrollo constituye un marco metodológico que define en términos de metas estratégicas, objetivos, actividades y artefactos (documentación) requeridos en cada fase de desarrollo. Esto permite enfocar esfuerzo de los recursos humanos en términos de habilidades, competencias y capacidades a asumir roles específicos con responsabilidades bien definidas. El RUP es una de las metodologías robustas o pesadas, que presenta entre sus características ser un proceso de desarrollo orientado a objetos, utiliza el Lenguaje Unificado de Modelado (UML) como lenguaje de representación visual. Este proceso unificado define "Quién", "Cómo", "Cuándo" y "Qué" debe hacerse en el proyecto. Tiene tres características fundamentales: es iterativo e incremental, centrado en la arquitectura y dirigido por Casos de Usos.

Dirigido por Casos de Uso: "Los casos de uso representan los requisitos de software capturados durante el flujo de trabajo de requisitos, la planificación del proyecto se hace en términos de casos de uso, los desarrolladores crean realizaciones de casos de uso en términos de clases y subsistemas, los componentes se incorporan en los incrementos y cada uno realiza un conjunto de casos de uso, y por último se verifica que el sistema implementa los casos de uso correctos para el usuario. En otras palabras los casos de uso guían la arquitectura del sistema, enlazan todas las actividades del desarrollo y dirigen el proceso de desarrollo" (Jacobson, et al., 2004).

Centrado en la arquitectura: "La arquitectura representa la forma del futuro sistema en términos de vistas arquitectónicas, sobre la cual equipo de desarrollo y usuarios deben estar de acuerdo, ya que estas describen los elementos del modelo más importantes para su desarrollo, la arquitectura va madurando en las interacciones comenzando con los casos de uso relevantes desde el punto de vista arquitectónico (Jacobson, et al., 2004).

Iterativo e incremental: "El Proceso Unificado propone que cada fase se desarrolle en iteraciones, ya que el incremento en la complejidad de los sistemas actuales hace que sea factible dividir el trabajo en partes más pequeñas o mini-proyectos. Cada mini-proyecto es una iteración que resulta en un incremento. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros" (Jacobson, et al., 2004). El período de vida del software esta particionado en ciclos, cada ciclo consta de cuatro fases: concepción o inicio, elaboración, construcción y transición y cada

vez que termina un ciclo se produce una versión del sistema. Es ideal para proyectos cuyos requisitos no son variables y para grandes equipos de desarrollo. Sin embargo puede adaptarse a diferentes condiciones. En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo, los 6 primeros son flujos de ingeniería y los tres últimos de apoyo.

El proceso de gestión de ficheros multimedia con los que trabajan los proyectos del Departamento es un proceso que ya existe, lo que no se ha creado un estándar para que las aplicaciones del departamento lo utilicen. Mediante entrevistas a los principales líderes de los proyectos del departamento se pudo notar que las necesidades que estos presentan en cuanto a la gestión de ficheros son las mismas de manera general, por lo que se llegó a la conclusión de que los requisitos no son variables. La utilización de las metodologías ágiles estudiadas (XP y Scrum) serían subutilizadas teniendo en cuenta que una de sus principales ventajas radica justamente en que son apropiadas para proyectos cuyos requisitos varían con frecuencia, no ocurriendo así con la metodología RUP que es ideal para desarrollar aplicaciones con requisitos constantes. Debido a esta característica de RUP es posible, como se requiere en el actual proyecto, que las decisiones sean tomadas desde el principio y seguir al pie de la letra una planificación que desembocará en un producto con las funcionalidades deseadas.

Otra de las razones para escoger esta metodología y no otra es que RUP brinda la posibilidad de contar con una planificación de tareas y artefactos que deben ser generados en un orden, no es necesario que el cliente forme parte del equipo de desarrollo y basta con encuentros planificados entre ambas partes para esclarecer todos los aspectos necesarios durante el desarrollo del software y validar cada vez que sea necesario el trabajo de los desarrolladores. Debido a que el componente sería utilizado por las aplicaciones desarrolladas en los proyectos del Departamento de señales digitales, basado en las metodologías ágiles sería necesario que el Departamento de señales digitales y los líderes de los proyectos que lo componen fueran parte del equipo de desarrollo. Por el número de proyectos con que cuenta el Departamento y el escaso tiempo disponible con que cuentan los directivos del mismo se haría muy engorroso el desarrollo satisfactorio del componente. RUP brinda la posibilidad de obtener excelentes resultados sólo mediante la interacción entre ambas partes a través de reuniones.

RUP propone abundante documentación, característica que es fundamental para el componente, ya que en el futuro puede existir la necesidad de reutilizarlo, entenderlo y hasta modificarlo para ajustes a nuevas

necesidades que puedan surgir en el Departamento. Teniendo en cuenta la necesidad de una posterior integración con las aplicaciones que actualmente se desarrollan en el Departamento de Señales Digitales, RUP sería la metodología que más se ajusta para solucionar esta situación.

La razón principal por la que RUP fue seleccionada se debe a que las aplicaciones que se desarrollan en el Departamento de Señales Digitales usan RUP como metodología de desarrollo y el componente que se propone será parte de la mayoría de dichas aplicaciones. También RUP cuenta con una serie de características que facilitan la creación de dicho componente.

Basado en todo lo planteado, en los excelentes resultados ya obtenidos por la Universidad de Ciencias Informáticas con el uso de esta metodología y que es la misma que utilizan los proyectos del departamento para el desarrollo de sus aplicaciones, no queda duda de que el Proceso Unificado de Desarrollo (RUP) es la metodología más apropiada para lograr el objetivo general de este trabajo de diploma.

1.4.1.2. UML como soporte de la modelación de la solución propuesta

Debido a la necesidad de dar soporte a la metodología RUP en cuanto a modelado, diagramas y documentación fue necesario seleccionar al lenguaje unificado de modelado UML. Aunque este lenguaje no especifica a que metodología va orientado, en la práctica está muy relacionado con RUP.

El Lenguaje Unificado de Modelado (Unified Modeling Language), conocido como UML por sus siglas en inglés, es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema de software orientado a objetos. Este contiene diagramas que permiten la modelación de los diferentes componentes del sistema. Es utilizado en varias metodologías aunque presenta una estrecha relación con la que será utilizada en este trabajo de diploma: el Proceso Unificado de Desarrollo de Software (RUP), ya que esta metodología hace uso de todos los diagramas propuestos por este lenguaje.

UML surge debido a la necesidad de crear un estándar único que debía ser, ante todo, universal. Permite fusionar la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros de software que trabajan en el desarrollo orientado a objetos, de igual forma, los clientes, desarrolladores

y otras personas involucradas o interesadas, pueden comprender el funcionamiento y la estructura de un sistema determinado.

UML agrupa los diagramas en tres tipos diferentes.

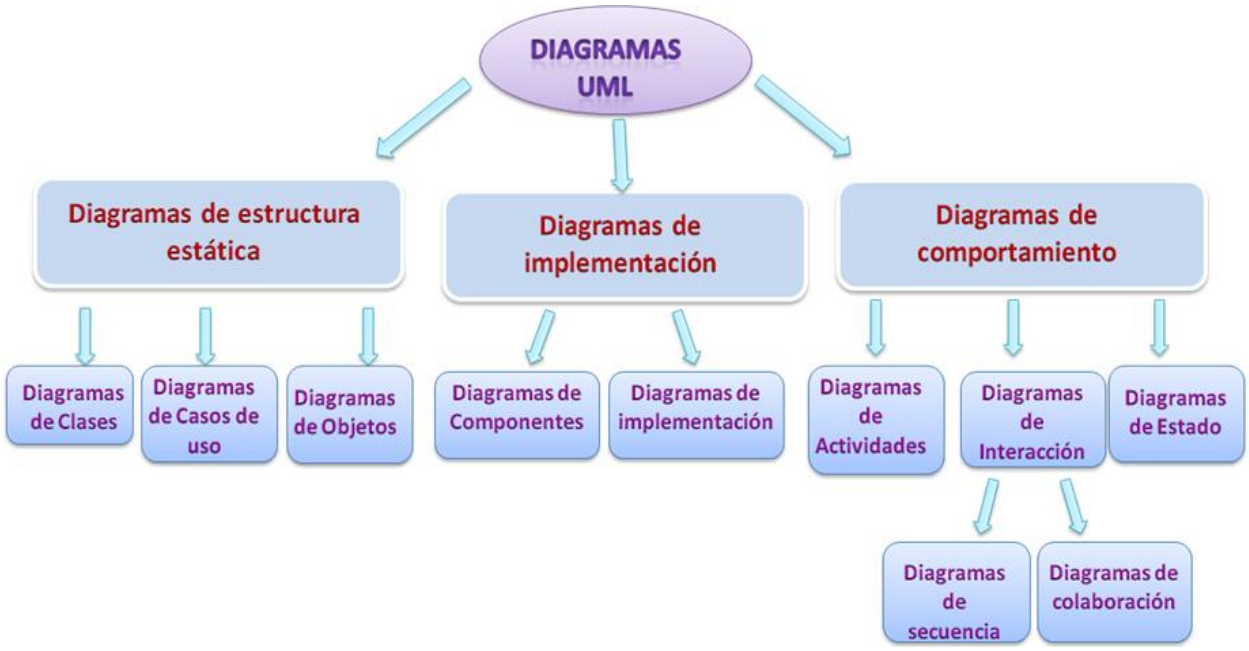


Figura 2: Diagramas UML

Algunas de las características que presenta este lenguaje son:

- Incluye “estereotipos” como mecanismo de extensibilidad.
- Aporta un lenguaje para expresar restricciones mediante fórmulas bien formadas como OCL (Object Constraint Language).
- Puede describir cualquier tipo de sistemas en términos de diagramas orientado a objetos.

Es ideal para el modelado de sistemas orientados a objetos ya que incluye la representación de la abstracción, herencia, polimorfismo, encapsulamiento o encapsulación, envío de mensajes, asociaciones

y agregación. Permite además detectar con mayor facilidad las dependencias y dificultades implícitas del sistema, con él se pueden modelar tanto sistemas de software y de hardware como organizaciones del mundo real.

1.4.2 Herramientas CASE (Computer-Aided Software Engineering)

Al definir una metodología de desarrollo en este caso RUP y un lenguaje que ayude a construir los elementos generados en ella, en este caso UML, es necesario entonces seleccionar una herramienta que permita la creación de los elementos necesarios haciendo uso del lenguaje de modelado. Las herramientas CASE son un conjunto de programas y ayudas que brindan asistencia técnica a analistas, ingenieros de software y desarrolladores para el análisis de requisitos, modelado visual y documentación durante parte o todo el ciclo de vida de un proyecto de software. La selección de esta herramienta está estrechamente relacionada con la metodología de desarrollo de software y el lenguaje de modelado a utilizar.

1.4.2.1. Visual Paradigm for UML Enterprise Edition como herramienta para el modelado

“El visual paradigm para el lenguaje unificado de modelado UML es una potente plataforma, la cual está diseñada para una amplia gama de usuarios, incluidos los analistas de sistemas, ingenieros de software, etc. Esta herramienta facilita la interoperabilidad con otras herramientas de modelado de UML además de permitir la transición de análisis para el diseño“(AU, 2007).

El Visual Paradigm for UML Enterprise Edition es una herramienta CASE poderosa y fácil de usar. Permite representar todo tipo de diagramas UML para las distintas fases como la captura de requisitos, análisis, diseño e implementación. Presenta, al igual que otras herramientas de modelado visual, una serie de ventajas tales como la generación del código fuente en java, C#, C entre otros a partir de diagramas de clases y además aplicar ingeniería inversa en los lenguajes Java, C++, CORBA IDL, PHP, XML, Ada y Python. Unas de las características que presenta es:

- Navegación intuitiva entre el modelo visual y el código.
- Modela todos los diagramas de UML.
- Validación de modelos en tiempo real.

- Presenta recursos centrado en la interfaz para mejorar la usabilidad Diagrama de diseño automático.
- Permite exportar diagramas como imagen en el formato JPG, PNG y SVG.
- Presenta sub-diagramas de apoyo para todos los modelos UML.
- Diseño centrado en Caso de Uso y enfocado al negocio que le permite generar un software con mayor calidad. Importa Racional Rose Project.

Esta herramienta CASE es una de las que soporta el análisis textual, una técnica que se utiliza para la captura de requisitos. Una característica fundamental que presenta esta herramienta es la disponibilidad de múltiples plataformas: es soportada tanto en el Sistema Operativo Windows como en el GNU/Linux. Es importante es señalar que existen algunas de sus versiones que son gratuitas.

Visual Paradigm for UML Enterprise Edition es sin dudas la elección más apropiada para modelar la solución propuesta. Además de ser la herramienta CASE establecida en el Departamento cuenta con una serie de características de relevancia para el desarrollo del componente. Esta es una herramienta que permite el modelado orientado a objetos, y está, a la vez, orientada al Lenguaje Unificado de Modelado (UML) que fue el escogido en este trabajo de diploma como soporte para la modelación.

1.4.3 Gestores de Base de Datos

Para almacenar los datos asociados al proceso de gestión de ficheros multimedia es necesario contar con un sistema gestor de base de datos que proporcione la seguridad requerida a dichos datos.

Un sistema gestor de base datos es el conjunto de programas que administran y gestionan la información contenida en una base de datos. Ayuda a llevar a cabo la definición de los datos, así como el mantenimiento de su integridad, el control de su seguridad, su privacidad y su manipulación.

“Un sistema gestor de base de datos está compuesto del gestor de la base de datos, que se trata de un conjunto de programas no visibles al usuario final que se encarga de la privacidad, integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporcionando una interfaz entre los datos, los programas que los manejan y los usuarios finales.”(Álvarez, 2008)

1.4.3.1 PostgreSQL

Según las políticas de desarrollo del Departamento de Señales Digitales el Sistema Gestor de Base de Datos que se emplea para la construcción de las soluciones informáticas que dicho Departamento ofrece es PostgreSQL, por ello se especificarán sus principales características y ventajas que ofrece para el desarrollo del componente.

PostgreSQL es un sistema de gestión de base de datos objeto-relacional pues incluye características de orientación a objetos como disparadores, reglas, restricciones, tipos de datos, herencia, funciones entre otros. La extensibilidad de su código fuente y la disponibilidad para todos sin costo alguno son unas de las ventajas que hacen que se encuentre entre la preferencia de muchos y a la altura de cualquier sistema gestor de base de datos comercial.

Estabilidad, potencia, robustez, facilidad de administración e implementación de estándares son algunas de las características que han definido a PostgreSQL desde el comienzo de su desarrollo hace más de 15 años (ALVAREZ, 2006). Mediante la utilización de un modelo cliente/servidor y multiprocesos en sustitución de los multihilos permite que un fallo en uno de los procesos no afecte al resto del sistema, garantizando así la estabilidad del sistema (Martínez, 2010).

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.

1.4.4 NetBeans como Entorno de Desarrollo Integrado (IDE)

NetBeans es un Entorno de Desarrollo Integrado (IDE) basado en código abierto y en una plataforma de aplicación de escritorio genérica. Una de las características que presenta esta herramienta es que puede cargar módulos dinámicamente y además permite montar una aplicación de módulos, lo cual posibilita

beneficiarse de trabajos realizados por otros. Cuenta también con las bibliotecas visuales API[3] que se encargan de la visualización de datos y la creación de fichas de un texto de entrada.

Además se debe agregar que 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 que 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. NetBeans permite vincular todas las tecnologías seleccionadas para desarrollar el componente, permite el acceso a datos usando PostgreSQL y aporta soporte para el lenguaje PHP.

1.5 Conclusiones Parciales

Durante el presente capítulo se enunciaron y caracterizaron los principales componentes tecnológicos a tener en cuenta en el desarrollo de la solución propuesta en este trabajo de diploma. Se fundamentaron las decisiones tomadas en cuanto a la metodología de desarrollo de software, el lenguaje de programación, el gestor de base de datos y el entorno de desarrollo integrado apropiados para conseguir con calidad y eficiencia el objetivo general de la presente investigación.

[3] Una interfaz de programación de aplicaciones o API (del inglés Application Programming Interface) 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.

Capítulo 2: Características de la solución propuesta

2.1 Introducción

Alcanzar cierto nivel de conocimientos sobre el problema en cuestión es un paso fundamental en los inicios del proceso de desarrollo de un software. En este capítulo se comenzó la investigación del medio donde se implementó la aplicación con el objetivo de definir las condiciones o capacidades que debe ofrecer el componente. Fueron abordados los principales aspectos de los flujos de trabajo Modelamiento del Negocio y Requerimientos, que son, entre los flujos de trabajo de ingeniería que propone RUP, los que tienen un mayor peso en la fase de Inicio.

Se conceptualizó el entorno mediante un modelo de dominio en el cual se analizó cada una de las entidades y conceptos presentes en el contexto donde se insertaría el componente, además de las relaciones existentes entre cada uno de estos. Se especificaron los requisitos funcionales y no funcionales que debe cumplir la solución propuesta, se presentaron los diagramas de Casos de Uso del Sistema, se describieron los actores y se detallaron los Casos de Uso del Sistema.

2.2. Modelamiento del negocio

Un sistema, por pequeño que sea, generalmente es complicado, por eso se necesita dividirlo en piezas para una mejor comprensión de su esencia y gestión de su complejidad. Esas piezas se pueden representar a través de modelos que permitan abstraer sus características principales. Uno de los modelos útiles previo al desarrollo de un software es el modelo del negocio. Para lograr esos propósitos, el proceso de modelamiento permite obtener una visión de la organización que permita definir los procesos, roles y responsabilidades de la organización en los modelos de Casos de Uso del negocio y de objetos. En inicio se hace una evaluación de la organización en la cual se implantará el futuro sistema y, basado en los resultados, se tomarán decisiones sobre cómo continuará esa iteración.

Dependiendo de la situación o escenario que se presente, hay varias alternativas de desarrollar este proceso. Debido a que el componente que gestione los ficheros multimedia para las aplicaciones web del Departamento de Señales Digitales que se va a desarrollar está concebido para ser posteriormente

integrado a dichas aplicaciones, el cual puede ser personalizado según las necesidades de esta, resulta difícil encontrar procesos de negocio bien estructurados que permitan realizar un modelado completo de dicho negocio, por lo que se decidió realizar un modelo de dominio y un glosario de términos para identificar y describir todos los términos presentes en el modelo.

2.2.1 Modelo de dominio

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema (Jacobson, y otros, 2004).

El modelado de dominio tiene como objetivo describir y comprender las clases y objetos más significativos dentro del contexto del problema, lo cual ayuda a definir los procesos y roles más trascendental para el sistema a desarrollar. Esto ayuda a los usuarios, desarrolladores y clientes a manejar un vocabulario común, que permita compartir el conocimiento manteniendo un lenguaje único y consistente que evite confusiones y posibilite el entendimiento de todas las partes interesadas.

2.2.2 Eventos principales del entorno

El proceso de **transferencia** de **ficheros multimedia** consiste en **almacenar** ficheros multimedia en **servidores** de almacenamiento estableciendo la comunicación mediante **protocolos** de transferencia. En una primera versión del componente serán empleados los protocolos **http** y **ftp** para efectuar la transferencia. El protocolo http será empleado para establecer la transferencia de ficheros hacia un servidor que se denominará **local** ya que constituye el mismo servidor donde estará ejecutándose la aplicación. El protocolo ftp será empleado para establecer la transferencia de ficheros hacia un servidor **externo.** Un servidor externo no es más que un servidor diferente al servidor donde está ejecutándose la aplicación. En una primera versión del componente será hacia un servidor ftp ya que es uno de los más utilizados para el almacenamiento de ficheros multimedia en el Departamento de Señales Digitales.

El proceso de gestionar un almacenamiento consiste en la **creación** de un punto de **almacenamiento** ya sea en un servidor local o externo mediante una **ruta** especificada por el usuario**.** Un almacenamiento

podrá ser **actualizado** para cambiar sus datos y **eliminado** a necesidad o gusto del usuario, además de **listar** su contenido y los almacenamientos existentes.

El proceso de gestión de ficheros comienza cuando un usuario necesita **copiar** un **fichero** en un almacenamiento. Copiar un fichero multimedia localmente consiste en subir un fichero a una carpeta temporal del servidor, luego, al usuario especificar un directorio dentro del propio servidor, el fichero multimedia será movido a esa dirección. Un fichero también podrá copiarse en un servidor externo haciendo una copia del fichero en una carpeta temporal del servidor local, luego, al usuario especificar un directorio dentro de uno de los servidores ftp existentes (previamente adicionados) el fichero multimedia será movido a esa dirección. El usuario también podrá **eliminar**, **buscar** y **renombrar** un fichero determinado.

El proceso de gestión de carpetas es semejante al proceso de gestión de ficheros con la diferencia de que el usuario podrá **crear** una **carpeta**. El usuario podrá crear una carpeta seleccionando un directorio dentro de un almacenamiento determinado para archivar ficheros multimedia de una manera organizada. También podrá eliminar y renombrar carpetas.

2.2.3 Glosario de términos del dominio

Buscando comprender el dominio donde se desplegará el componente se realizó un estudio de varios proyectos del Departamento de Señales Digitales donde fueron identificados los siguientes elementos pertenecientes al dominio del proceso de gestión de ficheros multimedia:

Almacenamiento: Ruta o directorio dentro de un servidor donde se guarda físicamente los ficheros multimedia.

> Externo: Se refiere al tipo de almacenamiento que requiere transferir los ficheros multimedia hacia un servidor diferente a donde se está ejecutando la aplicación a través del protocolo FTP.
>
> Local: Se refiere al tipo de almacenamiento empleado para almacenar los ficheros multimedia en el mismo servidor donde se está ejecutando la aplicación.

Archivo: Conjunto de información que se almacena en algún medio de escritura que permita ser leído o accedido por una computadora. Es identificado por un nombre y la descripción de la carpeta o directorio que lo contiene.

Base de datos: Entidad en la cual se almacenan de manera estructurada y con la menor redundancia posible los datos referentes a los ficheros multimedia.

Carpeta: Denominación metafórica que se le da a un directorio en el entorno gráfico de un sistema operativo. (Ver directorio)

Dirección del fichero: Dirección física del fichero multimedia en el servidor de medias incluyendo el nombre y la extensión del fichero.

Directorio: Agrupación de archivos de datos, atendiendo a su contenido, a su propósito, o a cualquier criterio que decida el usuario. Técnicamente almacena información de los archivos que contiene.

Ficheros multimedia: Archivo multimedia de audio o video.

Protocolo: En el campo de la informática es un conjunto de reglas que definen el intercambio de datos entre computadoras.

Ruta: Es la forma de referenciar una archivo informático o directorio en un sistema operativo. Señala la ubicación exacta de un archivo o directorio mediante una cadena de caracteres concreta.

Servidor: Servidor donde se van a almacenar los ficheros multimedia.

Usuario: Persona que interactúa con el componente para la gestión de ficheros multimedia.

2.2.4 Diagrama de clases del modelo de dominio

Luego de identificados los eventos y términos más significativos presentes en el entorno del dominio de la gestión de ficheros, el Diagrama de clases del modelo de dominio donde intervienen estos elementos queda estructurado de la forma siguiente:

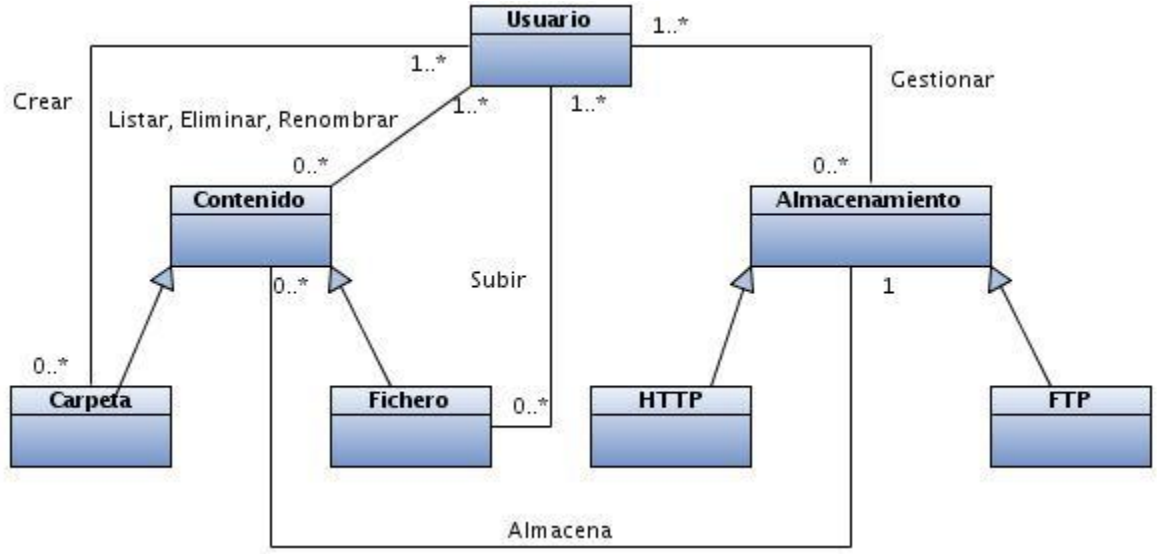


Figura 3: Modelo de dominio

2.3 Requisitos

Roger Pressman en su libro "Ingeniería de Software. Un enfoque práctico"plantea: "La parte más difícil en la construcción de sistemas software es decidir precisamente qué construir. Ninguna otra parte del trabajo conceptual es tan dificultosa como establecer los requerimientos técnicos detallados, incluyendo todas las interfaces con humanos, máquinas y otros sistemas software. Ninguna otra parte del trabajo puede perjudicar tanto el resultado final si es realizada en forma errónea. Ninguna otra parte es tan dificultosa de rectificar posteriormente" (Pressman, 2003).

El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema (es decir, las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores sobre que debe y que no debe hacer el sistema (Jacobson, y otros, 2004).

La captura de requisitos es la actividad mediante la cual el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. El

proceso de captura de requisitos puede resultar complejo, principalmente si el entorno de trabajo es desconocido para el equipo de analistas, por lo que se recomienda trazar una estrategia a seguir que permita llevar a cabo este proceso correctamente obteniendo los requerimientos reales que debe cumplir el sistema a implementar. La Ingeniería de Requisitos establece modelos de procesos que incluyen técnicas, métodos y plantillas que permiten realizar la captura de requisitos de una manera más eficiente y segura.

2.3.1 Requisitos funcionales

Los requisitos funcionales describen lo que el sistema debe hacer: son todas las condiciones y capacidades que debe cumplir el software, o producto en general, para que las peticiones del cliente queden satisfechas. El Componente que gestione los ficheros multimedia de las aplicaciones web del Departamento de Señales Digitales debe ser capaz de:

RF1: Gestionar almacenamiento: El componente debe ser capaz de gestionar un almacenamiento ya sea local o externo usando diferentes protocolos de transferencia.

RF 1.1 Gestionar almacenamiento externo: El componente debe permitir gestionar un almacenamiento mediante el protocolo ftp. Crear, Eliminar, Editar y Listar almacenamiento externo.

RF 1.2: Gestionar almacenamiento local: El componente debe permitir gestionar un almacenamiento mediante el protocolo ftp. Crear, Eliminar, Editar y Listar almacenamiento local.

RF 2: Gestionar fichero

RF2.1 Copiar fichero: El componente debe permitir subir un fichero multimedia al servidor de almacenamiento ya sea por los protocolos ftp o http.

RF2.2 Eliminar fichero: El componente debe permitir borrar un fichero multimedia de cualquier servidor de almacenamiento.

RF2.3 Renombrar fichero: El componente debe permitir editar los datos de un fichero multimedia de cualquier servidor de almacenamiento.

RF 3: Gestionar carpeta

RF3.1 Crear carpeta: El componente debe permitir crear carpetas dentro de un almacenamiento.

RF3.2 Eliminar carpeta: El componente debe permitir eliminar carpetas en cualquier almacenamiento

RF3.3 Renombrar carpeta: El componente debe permitir cambiarle el nombre a una carpeta.

RF 4.1 Listar carpetas y ficheros: El componente debe permitir explorar el contenido de cualquier almacenamiento.

2.3.2 Requisitos no funcionales

Los requerimientos no funcionales pueden ser definidos como propiedades o cualidades que el producto debe tener, son aspectos importantes que este debe cumplir para lograr un aprovechamiento óptimo de las funcionalidades del sistema teniendo en cuenta el entorno en el que será utilizado. A continuación se enuncian, separados en categorías, los diferentes requisitos no funcionales que el componente debe satisfacer.

Usabilidad

- El proceso de subir ficheros multimedia al servidor debe caracterizarse por un alto grado de flexibilidad.
- Mostrar la información de forma lógica y correctamente estructurada.
- El componente podrá será usado por cualquier persona que tenga un conocimiento medio de computación e informática[4].

[4] Se define como conocimiento medio a una persona capaz de manipular un ordenador y conocer los procesos esenciales que con él se pueden realizar

Soporte

- El soporte y/o mantenimiento del componente no debe detener el servicio de la aplicación a la que esté integrado.
- El componente debe estar bien documentado para una exitosa integración con aplicaciones futuras.
- El componente contará con un manual de usuario donde este podrá suplir las dudas que se le puedan presentar durante la utilización del mismo.

Seguridad

- Los datos de las conexiones con los servidores deberán estar encriptados.
- El sistema garantizará la confidencialidad, integridad y disponibilidad de la información almacenada en la base de datos.
- Se deberán garantizar por parte de la aplicación a la que se integre el componente la autenticación de un usuario antes de que pueda realizar cualquier acción y asegurar que las funcionalidades del componente se muestren de acuerdo al rol del usuario que esté activo (Autorizar), denegando acciones no autorizadas que puedan afectar la integridad de los datos y la del sistema.

Hardware

El servidor donde se ejecute el componente en conjunto con la aplicación a la que posteriormente se integre debe tener los siguientes requerimientos mínimos de hardware aunque estos pueden variar según las características de la propia aplicación:

Requisito Mínimo: Procesador: Pentium III 800 MHz, Memoria: 256 Mb, Disco Duro: 80 Gb.

Requisitos Recomendados: Procesador: Pentium IV 1.8 GHz, Memoria: 512 Mb, Disco Duro: 120 Gb.

Software

Servidores

- Se debe utilizar como servidor web el “Apache”.
- Se debe utilizar como gestor de base de datos “PostgreSQL”.
- Se debe utilizar como lenguaje de programación del lado del servidor “Php”.

Cliente

- La aplicación debe correr en sistemas operativos Windows, Unix y GNU/Linux. Deberá disponer de un navegador web, estos pueden ser Internet Explorer, Opera, Google Chrome y Firefox o versiones superiores de estos.

Restricciones en el diseño y la implementación

- Diseño e implementación de una arquitectura flexible, que permita la fácil integración o desintegración del componente.
- El patrón arquitectónico que se debe emplear en el desarrollo es el Modelo-Vista-Controlador.
- La abstracción en la implementación debe llegar a un grado en el que sea posible permitir nuevas formas de transferencias.
- Los protocolos para la comunicación con el servidor de almacenamiento que se deben usar son HTTP y FTP.

2.4 Descripción del sistema propuesto

El sistema que se propone debe estar compuesto por los actores y Casos de Uso que se describen a continuación.

2.4.1 Definición de los actores

Un actor no es más que la entidad externa al sistema que guarda una relación con este y que le solicita una funcionalidad.

Usuario: Persona que interactúa con el componente para la gestión de los ficheros multimedia.

El actor del sistema hace referencia a un usuario que de manera general interactúa con el componente para probar el correcto funcionamiento del componente. La definición de los actores, sus privilegios y responsabilidades deberá ser especificada por la aplicación a la que se integrará el componente de acuerdo a las características de esta.

2.4.2 Casos de Uso del Sistema

Los Casos de Uso se utilizan para capturar el comportamiento deseado del sistema en desarrollo, sin tener que especificar cómo se implementa ese comportamiento. Cada Caso de Uso proporciona uno o varios escenarios que indican cómo debería actuar el sistema con el usuario o con otro sistema para alcanzar un objetivo determinado. Proporcionan un medio para que lo desarrolladores, los usuarios finales del sistema y los expertos del dominio lleguen a una comprensión común del sistema.

Cada Caso de Uso del Sistema puede ser catalogado como crítico, secundario, auxiliar u opcional en correspondencia a la importancia que estos tengan dentro del sistema y atendiendo a las peticiones del cliente. A continuación se relacionan los Casos de Uso correspondientes al presente trabajo de diploma y sus respectivas clasificaciones:

Críticos: Gestionar almacenamiento, Gestionar fichero, Gestionar carpeta.

Secundarios: Listar Contenido.

Como es posible apreciar el componente cuenta con cuatro Casos de Uso, de ellos tres son catalogados como críticos y arquitectónicamente significativos: Gestionar almacenamiento, Gestionar fichero y Gestionar carpeta pues estos son los que representan las partes más críticas de la arquitectura del sistema y demuestran la funcionalidad del mismo. Dentro de estos tres Casos de Uso, Gestionar almacenamiento, fue creado bajo los planteamientos del patrón CRUD de Casos de Uso que propone el agrupamiento de las funcionalidades que se llevan a cabo sobre una misma entidad, tales como listar, insertar, modificar y eliminar. Aunque los tres Casos de Uso comienzan su nombre con la palabra “Gestionar”, solo el Caso de Uso “Gestionar almacenamiento” se corresponde con el patrón CRUD. El resto se denominan así además de la similitud que tienen con el patrón CRUD de acuerdo a la función que realizan, también para una mejor forma de organización y entendimiento por parte de los usuarios.

2.4.2 Diagrama de Casos de Uso del Sistema

En el Diagrama de Casos de Uso del Sistema aparecen representados los requerimientos funcionales como Casos de Uso y se especifica la comunicación y el comportamiento del sistema mediante su interacción con el usuario.

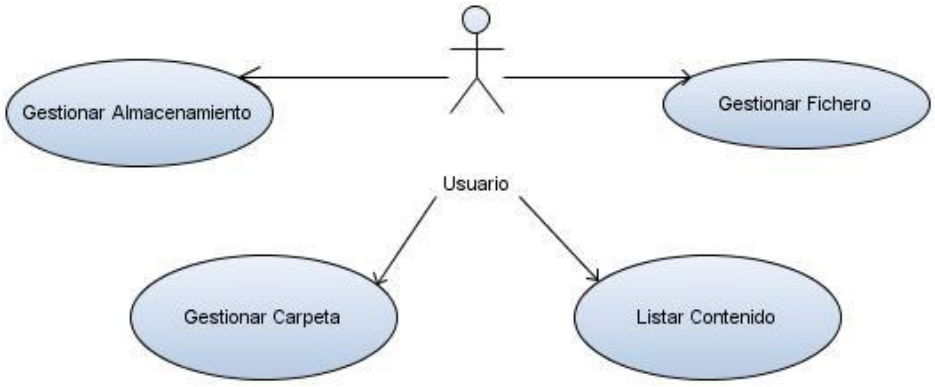


Figura 4: Diagrama de Casos de Uso del Sistema

2.4.3 Descripción textual de los Casos de Uso del Sistema.

Descripción del flujo de sucesos **Gestionar almacenamiento.**

Caso de Uso:	Gestionar almacenamiento.
Actores:	Usuario
Resumen:	Este Caso de Uso tiene lugar cuando el usuario gestiona los almacenamientos del sistema.
Precondiciones:	Que el usuario este autenticado
Referencias	RF1
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del actor	Respuesta del sistema

1. El usuario selecciona la opción “Almacenamientos” del menú principal de la aplicación.	2. El sistema muestra un listado con los tipos de almacenamientos existentes (ftp o http (local)) y las opciones adicionar y listar para cada uno.
3. El usuario escoge una de las opciones: a. Adicionar nuevo almacenamiento, seleccionando el vínculo de “Adicionar”. Sección Adicionar almacenamiento. b. Listar almacenamientos, seleccionando el vínculo de “Listar”. Sección Listar almacenamiento.	
Adicionar almacenamiento.	
Acción del actor	Respuesta del sistema
	4. El sistema muestra un formulario para entrar los datos del almacenamiento y el botón “Guardar”.
5. El usuario entra los datos del almacenamiento y selecciona la opción guardar.	6. El sistema valida los datos entrados por el usuario.
	7. El sistema verifica que el almacenamiento no exista en la ruta seleccionada.
	8. El sistema almacena los datos del almacenamiento y muestra el mensaje “El almacenamiento se adicionó satisfactoriamente”.
Flujos Alternos	
Acción del actor	Respuesta del sistema

	6 Si alguno de los campos obligatorios no fueron llenados el sistema muestra un mensaje indicando el campo que falto por llenar no puede ser vacío y pasa al flujo 3 de la sección “Listar almacenamientos”.
	7 Si existe un almacenamiento en la ruta especificada el sistema muestra un mensaje indicando que la ruta ya existe.
5 El usuario selecciona el botón cancelar.	8 El sistema pasa a la acción 2 del flujo normal de eventos.
Listar almacenamientos	
Acción del actor	Respuesta del sistema
	4. El sistema muestra un listado con los almacenamientos de ese tipo existentes y las opciones Editar, Eliminar y Explorar para cada uno de ellos. Además del botón Cancelar.
5. El usuario escoge una de las opciones: a. Explorar el contenido del almacenamiento. Sección Explorar almacenamiento. b. Editar los datos del almacenamiento. Sección Editar almacenamiento. c. Eliminar un almacenamiento. Sección Eliminar almacenamiento.	
Explorar almacenamiento	
Acción del actor	Respuesta del sistema
	6. El sistema muestra el contenido (ficheros y carpetas) del almacenamiento deseado con

	los vínculos editar y eliminar para cada uno.
Editar almacenamiento	
Acción del actor	Respuesta del sistema
	6. El sistema muestra un formulario para entrar los datos del almacenamiento y los botones “Guardar” y “Cancelar”.
7. El usuario entra los datos del almacenamiento y puede elegir la ruta mediante el botón explorar y selecciona la opción guardar.	8. El sistema valida los datos entrados por el usuario.
	9. El sistema verifica que el almacenamiento no exista en la ruta seleccionada.
	10. El sistema almacena los datos del almacenamiento y muestra el mensaje “Datos actualizados satisfactoriamente”.
Flujos Alternos	
	8. Si alguno de los campos obligatorios no fueron llenados el sistema muestra un mensaje indicando que el campo que falto por llenar no puede ser vacío y pasa a la acción 1 del flujo normal de eventos.
	9. Si existe un almacenamiento con el mismo nombre en la misma ruta el sistema muestra un mensaje indicando que la ruta ya existe.
7. El usuario selecciona el botón cancelar.	10. El sistema pasa a la acción 5 del flujo normal de eventos.
Eliminar almacenamiento	
	6. El sistema muestra un mensaje preguntándole al usuario si está seguro que desea eliminar el almacenamiento

	seleccionado.
7. El usuario confirma la eliminación del almacenamiento	8. El sistema muestra el mensaje "El almacenamiento se eliminó satisfactoriamente".
Flujos Alternos	
7. El usuario selecciona el botón cancelar.	8. El sistema pasa a la acción 5 del flujo normal de eventos.
Poscondiciones	Quedan actualizados los almacenamientos en el sistema

Tabla 1: Descripción del flujo de sucesos del CU "Gestionar almacenamiento".

Consultar anexos para ver la descripción del resto de los Casos de Uso del Sistema.

2.5 Conclusiones parciales

El Modelo de Dominio realizado a partir de los procesos identificados permitió conocer todos los términos y conceptos presentes en el entorno, los cuales fueron descritos y especificados en un glosario de términos y representados en un diagrama de clases, el cual ayudó a comprender mejor el funcionamiento e interrelación de los mismos. Mediante la realización de la disciplina Levantamiento de Requisitos, se consiguió identificar las funcionalidades que el componente para la gestión de ficheros multimedia debe brindar y las restricciones sobre las que va a operar.

La conformación del Diagrama de Casos de Uso del Sistema permitió traducir los requisitos funcionales en interacciones del actor con el sistema, lo que hizo posible que se obtuviera una visión de cómo sería la interacción del usuario con las funcionalidades que brinda el componente. La descripción detallada del flujo de eventos de cada Caso de Uso en un lenguaje formal y entendible posibilitó comprender en detalle los procesos incluidos en el cumplimiento de los objetivos trazados para cada Caso de Uso.

Capítulo 3: Análisis y Diseño de la solución propuesta

3.1 Introducción

El propósito de este capítulo es dejar expuestos los principales artefactos generados durante el flujo de trabajo Análisis y Diseño. Se aborda la disciplina de Análisis y se construyó el diagrama de clases del análisis que sirvió de entrada para el estudio de la disciplina de Diseño.

Mediante un estudio de los patrones de diseño se logró alcanzar un entendimiento de cómo debe ser la estructura del diseño del componente a desarrollar, y para ello fueron presentados los elementos del Modelo de Diseño y la arquitectura del componente.

3.2 Disciplina de Análisis

Según la metodología empleada para la construcción de la solución, RUP, en la cual se contempla el Análisis y Diseño en un mismo Flujo de Trabajo por estar muy relacionadas, es importante destacar que son actividades diferentes con artefactos diferentes. Las actividades contempladas en el análisis comienza adentrarse en el problema a resolver por lo que a través de ellas se representa una vista interna del sistema en la que, usando el lenguaje de los desarrolladores, se refinan los requisitos y se estructuran en base a clases y paquetes. Este proceso continúa en el diseño hasta obtener los objetos que interactúan para cumplir los requisitos funcionales y no funcionales obtenidos.

3.2.1 Modelo del análisis

“En el ámbito técnico la ingeniería de software comienza con una serie de tareas de modelado que conducen a una especificación de requisitos y a una representación completa del diseño del software que se construirá. El modelo del análisis, que en realidad es una serie de modelos, es la primera representación técnica de un sistema” (Pressman, 2003).

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.

En la construcción del modelo de análisis se tienen que identificar las clases que describen la realización de los Casos de Uso, los atributos y las relaciones entre ellas. Con esta información se construye el Diagrama de clases del análisis, que por lo general se descompone para agrupar las clases en paquetes. Esta descomposición tiene impacto por lo general en el diseño e implementación de la solución.

3.2.1.2 Diagrama de clases del análisis

Un Diagrama de clases del análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Representa el funcionamiento del mundo real, no de la implementación automatizada del mismo. A continuación se refleja el diagrama de clases de análisis correspondiente a los casos de Usos críticos descritos en el capítulo anterior.

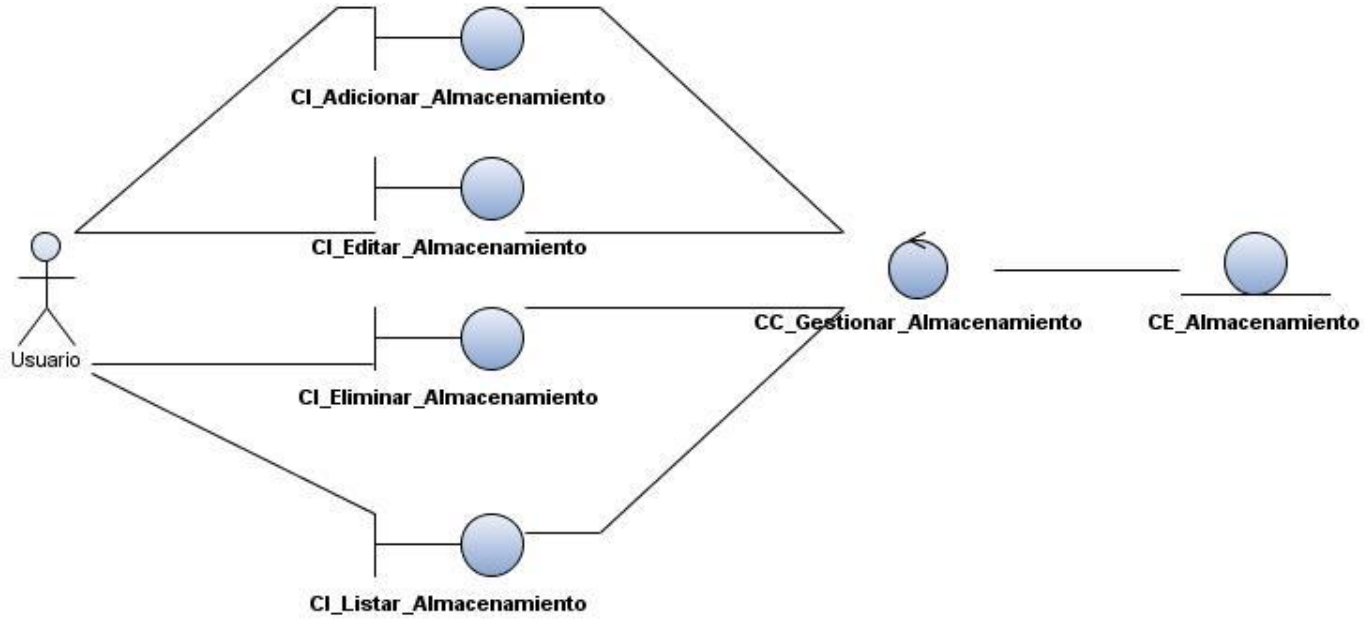


Figura 5: Diagrama de clase del análisis. Caso de Uso Gestionar almacenamiento

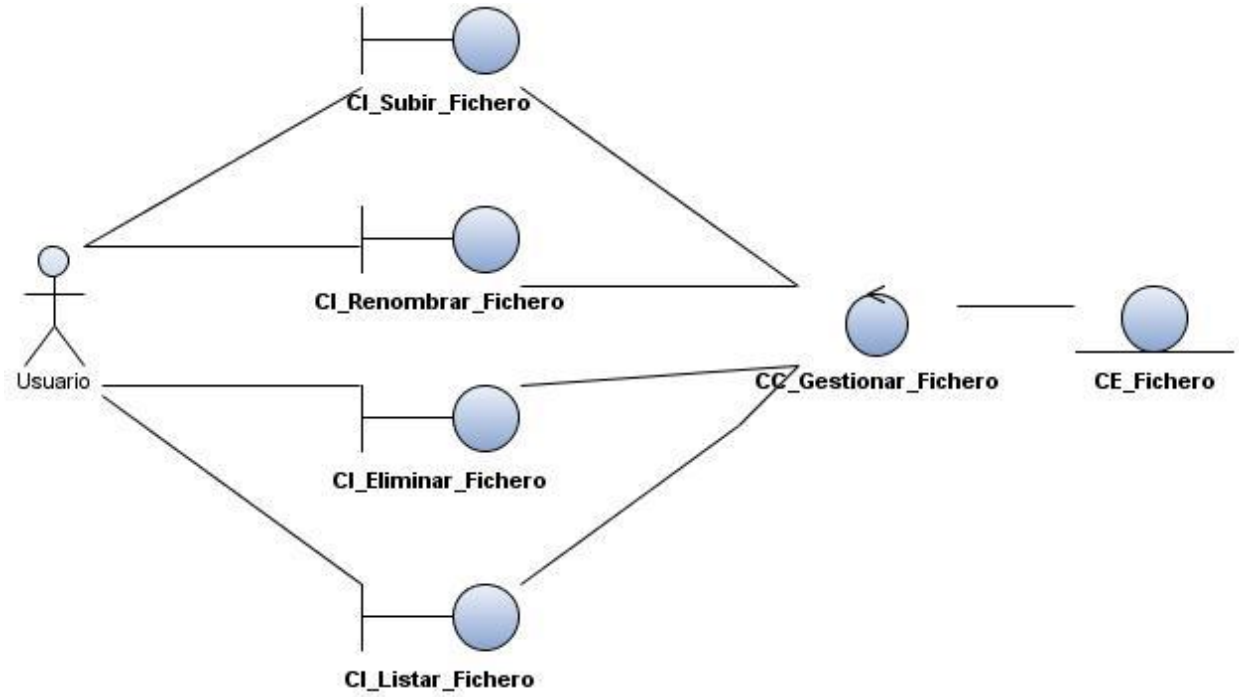


Figura 6: Diagrama de clase del Análisis. Caso de Uso Gestionar fichero.

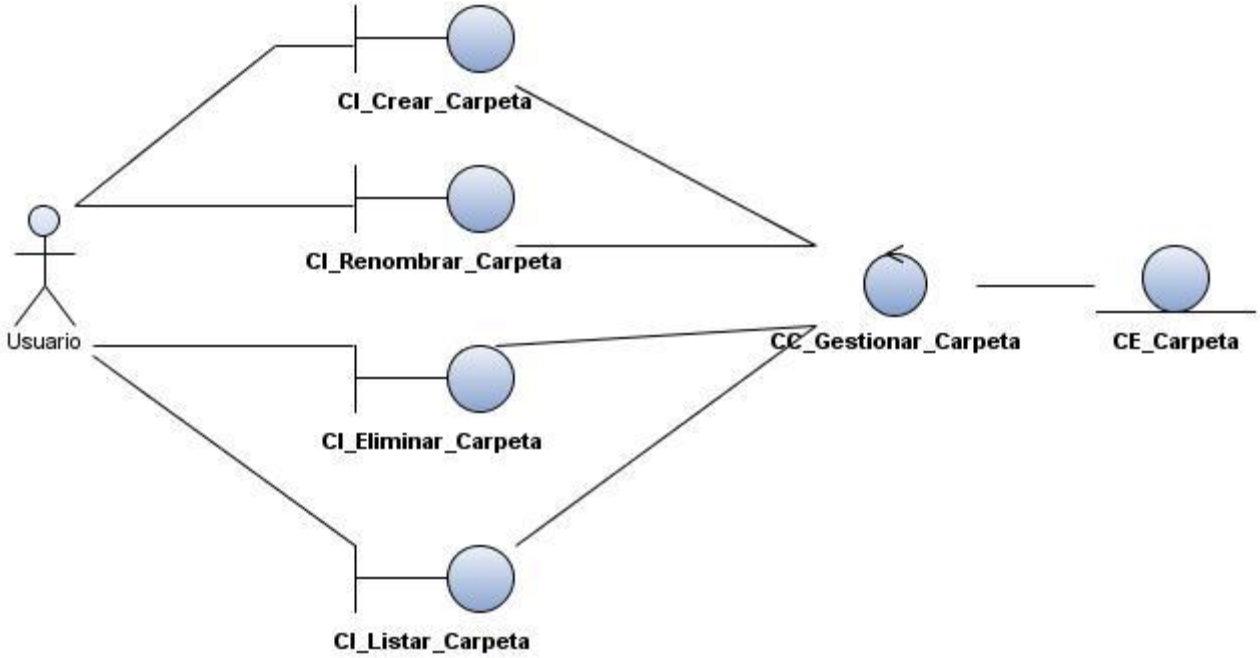


Figura 7: Diagrama de clase del Análisis. Caso de Uso Gestionar carpeta.

3.3 Disciplina de Diseño

El diseño tiene el propósito de modelar el sistema y encontrar su forma (incluida la arquitectura) para que soporte todos los requisitos, incluyendo los no funcionales y las restricciones que se le suponen. Además prepara las bases para la implementación y prueba del sistema. Una entrada esencial en el diseño es el resultado del análisis, o sea el modelo de análisis, que proporciona una comprensión detallada de los requisitos. Además impone una estructura del sistema que se debe conservar lo más fielmente posible a la hora de dar forma al sistema. Durante esta disciplina se crean fundamentalmente los siguientes artefactos:

- Modelo de Diseño.
- Modelo de Despliegue.
- Documento de la Descripción de la Arquitectura.
- Modelo de Datos.

De estos artefactos fueron creados durante el desarrollo de la investigación el Modelo de Diseño y el Modelo de Despliegue.

3.3.1 Arquitectura de Software

Una definición reconocida de Arquitectura de Software (AS) es la de Clements, quien plantea que: La AS es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones (Reynoso, 2004).

En medio de una amplia gama de definiciones de AS se ha acordado que la definición “oficial” de AS sea la que brinda el documento de IEEE Std 1471-2000, adoptada también por Microsoft, la cual expresa que la Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.

Una arquitectura define varios estilos arquitectónicos entre los que se destacan:

- Arquitectura centrada en los datos.
- Arquitectura centrada en los flujos de datos.
- Arquitectura llamada y respuesta (call and return).
- Arquitectura Orientada a Objetos.
- Arquitectura en capas.

3.3.2 Estilos arquitectónicos.

En la metodología RUP, después de un buen análisis de los requisitos y Casos de Uso se propone un documento de la arquitectura del software donde se expresa la arquitectura a través de un conjunto de vistas, empleando para ellos Estilos y Patrones de Arquitectura.

Un estilo arquitectónico define una familia de sistemas en términos de un patrón de organización estructural. En particular, de acuerdo a Mary Shaw y David Garlan en su clásico libro “Software Architecture” (1996), un estilo arquitectónico define tanto un vocabulario de tipos de componentes y conectores –como en el caso de filtros y tubos[5]- como un conjunto de restricciones sobre cómo combinar esos componentes y conectores. Existen numerosos estilos que agrupan patrones de diseño entre los que se destacan:

- Estilos de Flujo de Datos:

  Tubería y filtros

- Estilos Centrados en Datos:

  Arquitecturas de Pizarra o repositorio

- Estilos de Llamada y Retorno:

  Modelo-Vista-Controlador (MVC)

[5] Estilo arquitectónico que se basa en que cada componente, tiene un conjunto de entradas y un conjunto de salidas; cada componente lee corrientes de datos en sus entradas y produce corrientes de datos en las salidas.

Arquitectura en capas

Arquitecturas orientadas a objetos

Arquitecturas basadas en componentes

- Estilos de Código Móvil:

  Arquitectura de máquinas virtuales

- Estilos Heterogéneos:

  Sistemas de control de procesos

  Arquitecturas basadas en atributos.

- Estilos Peer-to-Peer:

  Arquitecturas basadas en eventos

  Arquitecturas orientadas a servicios

  Arquitecturas basadas en recursos.

Un patrón arquitectural expresa un esquema de estructura de organización fundamental para el sistema software, proporciona un conjunto de subsistemas predefinidos, especifica responsabilidades y contiene reglas y guías para organizar las relaciones entre ellas. Después de realizado un estudio de sobre los estilos arquitectónicos más importantes y los patrones que engloban se decidió utilizar el Estilo “Llamada y Retorno” el cual comprende al patrón arquitectónico Modelo Vista Controlador para la construcción del diseño de la solución propuesta.

Modelo Vista Controlador

El patrón arquitectónico Modelo Vista Controlador (MVC) ha demostrado ser muy apropiado para el desarrollo de aplicaciones web debido a que cuenta con características que hicieron que fuera el patrón seleccionado para el desarrollo de las funcionalidades a implementar en el componente que gestione los ficheros en las aplicaciones web del Departamento de Señales Digitales.

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 base de datos utilizado por la aplicación (Potencier, y otros, 2008).

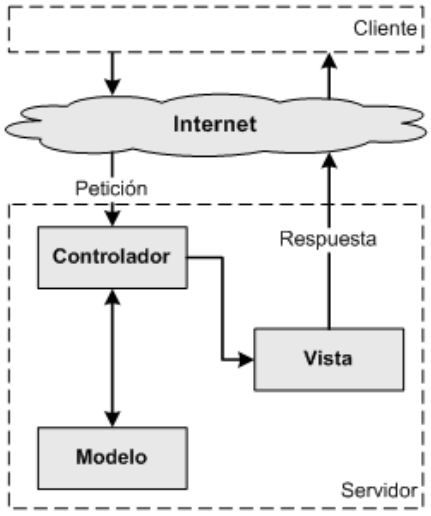


Figura 8: Patrón Arquitectónico Modelo Vista Controlador

Comportamiento pasivo

El Modelo Vista Controlador pasivo es el que utiliza la gran mayoría de las aplicaciones de uso común y es el utilizado en la solución propuesta. En este escenario el controlador manipula el modelo exclusivamente. Es decir que el controlador modifica el modelo y le informa a la vista que este ha cambiado y debe ser actualizada. A diferencia del comportamiento activo donde el modelo detecta los cambios a su estado interno cuando estos ocurren, el modelo deberá notificar a la vista para refrescarla, pero esto crearía una dependencia entre el modelo y la vista, lo cual iría en contra de uno de los principios

del patrón MVC. Como solución, se introduce el patrón Observer, el cual provee un mecanismo para alertar a otros objetos de cambios de estado sin introducir dependencias entre ellos.

3.3.3 Patrones de Diseño

"Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular."(Larman, 2007). Los patrones de diseño se dividen en dos grandes grupos: patrones GRASP (Patrones de asignación de responsabilidades) y patrones GOF (Grupo de los Cuatro).

La utilización de patrones de diseño en la construcción del componente ofrece numerosas ventajas ya que estos proponen una forma de reutilizar la experiencia de los desarrolladores, describiendo formas de solucionar problemas que ocurren de forma frecuente en el desarrollo de un software. Es una experiencia real, probada y que funciona. Ayuda a no cometer los mismos errores.

Los **Patrones GRASP** describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. El acrónimo GRASP significa General Responsibility Asignment Software Patterns (Patrones generales de Software para asignar responsabilidades) y se eligió precisamente para resaltar la importancia de captar (GRASPing) estos principios, si se quiere diseñar eficazmente el software orientado a objetos. En el modelo de diseño del componente de gestión de ficheros para las aplicaciones web del Departamento de Señales Digitales fueron utilizados los patrones GRASP Controlador, Experto, Bajo Acoplamiento y Alta Cohesión.

El patrón **Experto** propone la asignación de responsabilidades específicas a las clases creadas; es un principio básico que suele utilizarse en el diseño orientado a objetos. Expresa simplemente la "intuición" de que cada clase contiene los métodos relacionadas con la información que posee. Este patrón fue aplicado a todas las clases creadas durante la implementación del componente, en las cuales se pueden encontrar, en algunos casos, las funcionalidades correspondientes al manejo de los atributos que pertenecen a cada una, y en otros, simplemente aquellas que son necesarias para cumplir el objetivo con que fueron creadas. Por ejemplo, la clase controladora "HTTP_File_Storage_Management_Class", agrupa todos los métodos necesarios para la creación, modificación y eliminación de almacenamientos de tipo

http y la clase controladora “FTP_File_Storage_Management_Class”, agrupa todos los métodos necesarios para la creación, modificación y eliminación de almacenamientos de tipo ftp. Igual pasa con las clases “HTTP_Acceso:_a_Datos” y “FTP_Acceso:_a_Datos” las cuales se encargan de conectarse a la base de datos para realizar las operaciones necesarias con un almacenamiento http o ftp respectivamente.

El patrón **Controlador** actúa como intermediario entre una determinada interfaz y el algoritmo que implementa la funcionalidad, de tal forma que es la que recibe los datos del usuario y los envía a las distintas clases según el método llamado. En el caso del componente la clase controladora “File_Storage_Management_ API” es la que sirve como intermediario entre las clases interfaz donde el usuario introduce los datos a la hora de crear un almacenamiento por ejemplo y la API los recibe y los envía para las clases “HTTP_File_Storage_Management_Class” y “FTP_File_Storage_Management_Class” según el tipo de almacenamiento que quiera crear el usuario.

El patrón **Bajo Acoplamiento** propone la independencia de las clases del diseño con el fin de reducir el impacto de los cambios y permitir una mayor reutilización del código. Asigna las responsabilidades de forma tal que las clases se comuniquen entre sí lo menos posible. Este patrón es respetado durante la implementación del componente pues se ha incluido la menor cantidad posible de dependencias entre las clases.

El patrón **Alta Cohesión** propone la colaboración entre clases para llevar a cabo tareas con cierta complejidad. En el caso del componente, existe colaboración entre las clases controladoras para ejecutar funcionalidades complejas como subir un fichero multimedia a un almacenamiento. Es necesario resaltar que las relaciones entre clases son mínimas, solo las necesarias para llevar a cabo satisfactoriamente las funcionalidades deseadas y respetar los principios del patrón Bajo Acoplamiento.

Los **patrones GOF** aparecieron a principios de la década de 1990 en el libro “Desing Patterns” publicado por el grupo Gang of Four (Grupo de los Cuatro), de ahí proviene su nombre. En este libro se recogen un grupo de 23 patrones de diseños divididos en 3 categorías: Estructurales, de Comportamiento y Creacionales. Los patrones GOF utilizados son los siguientes:

Singleton (Patrón Creacional) o instancia única, propone la restricción de la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Este patrón es utilizado para garantizar que la clase "Conexión" que es la encargada de establecer la conexión con la base de datos, sólo tenga una instancia y proporcionar un punto de acceso global a ella desde las clases "HTTP_Acceso_a_Datos", "FTP_Acceso_a_Datos" y "Acceso_a_Datos".

Fachada (Patrón Estructural) consiste en implementar una interfaz simplificada para el acceso a un conjunto de clases o interfaces, de esta forma se proporciona una interfaz unificada de un subsistema sin ocultar sus interfaces y se reduce la dependencia entre clases. Este patrón ofrece un punto de acceso al resto de clases y así si éstas cambian o se sustituyen por otras, sólo hay que actualizar la clase Fachada sin que el cambio afecte a las aplicaciones cliente. En este caso, la clase "File_Storage_Management_API" sería la Fachada del componente y en la futura ampliación de la implementación del mismo, a la hora de agregar nuevas formas de transferencia bastaría con modificar esta clase y esto no afectaría al cliente.

3.3.4 Modelo de Diseño

El modelo de diseño puede verse en dos dimensiones diferentes. La dimensión del *proceso* indica la evolución del modelo de diseño conforme se ejecutan las tareas del diseño como una parte del proceso de software. La dimensión de *abstracción* representa el grado de detalle a medida que cada elemento del modelo del análisis se transforma en un equivalente y después se refina de una manera iterativa (Pressman, 2003).

Los elementos del modelo de diseño utilizan muchos de los diagramas en UML aplicados en el modelo de análisis. La diferencia es que estos diagramas están refinados y elaborados como parte del diseño; se proporciona un mayor detalle para la implementación específica y se resaltan la estructura y el estilo arquitectónico que residen dentro de la arquitectura y las interfaces entre los componentes.

El modelo de diseño está compuesto por el Diagramas de clases del diseño y Diagramas de interacción (Colaboración y/o Secuencia), Paquetes y Subsistemas de Diseño representados en una jerarquía y una breve descripción de ellos; y las clases, interfaces, relaciones, etc. contenidas en los Paquetes.

3.3.4.1 Diagrama de clases del diseño

Los diagramas de clases se realizan por cada Caso de Uso, exponen las diferentes clases que componen un sistema y la relación entre ellas. 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.

Durante el proceso de construcción del componente no se conocen características específicas de las aplicaciones web a las cuales este se integrará. Por ejemplo, las interfaces de usuario pueden variar de acuerdo a las particularidades de dichas aplicaciones. Por ello es necesario realizar un Diagrama de clases del diseño independiente de la lógica de presentación de dichas aplicaciones. A continuación se muestra un conjunto de clases que junto a la lógica de presentación de la aplicación que integrará al componente constituyen el Diagrama de Clases del Diseño. Estas interfaces deben conectarse a la API la cual contiene todas las funcionalidades del componente y se encarga de distribuir las responsabilidades a las clases correspondientes según la acción que desee realizar el usuario. De esta manera para incorporar nuevas funcionalidades al componente basta con adicionarlas a la API. Este diagrama fue desarrollado bajo el patrón arquitectónico Modelo Vista Controlador sin concentrar el diseño en la lógica de presentación.

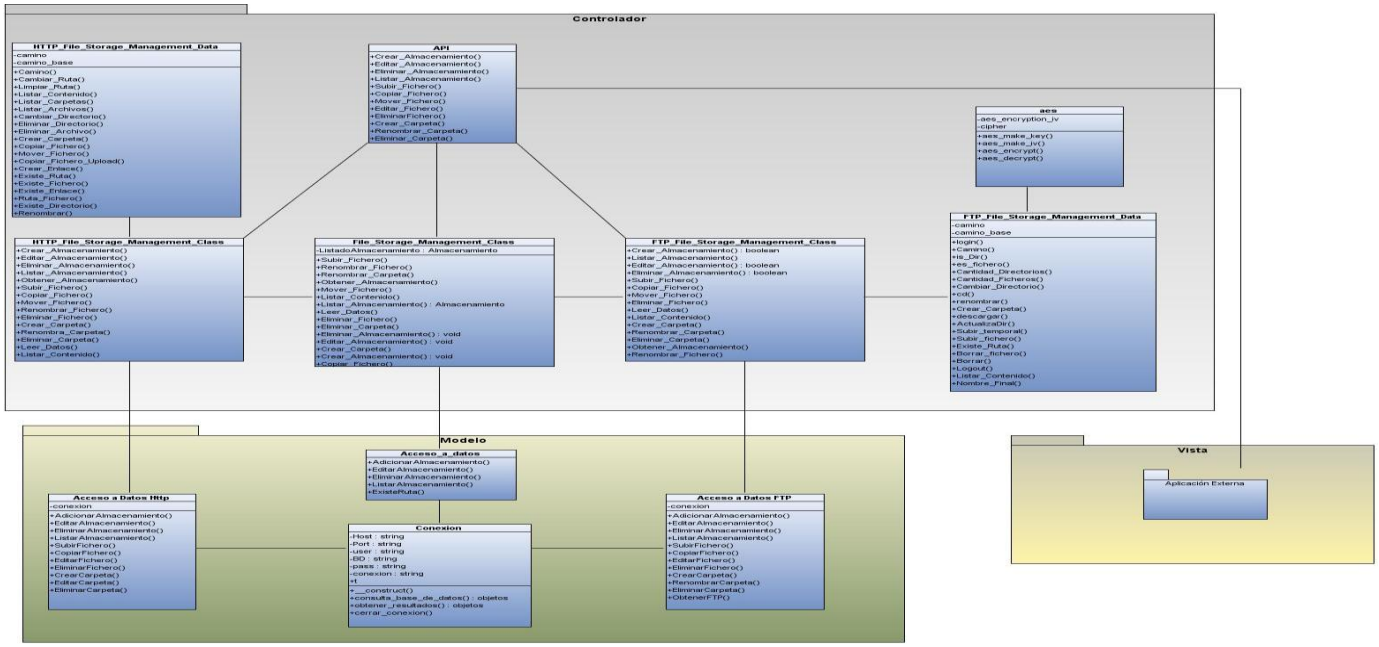


Figura 9: Diagrama de clases del diseño.

3.3.5 Diagramas de interacción

Los diagramas de secuencia y los diagramas de colaboración (ambos llamados diagramas de interacción) son dos de los cinco tipos de diagramas UML que se utilizan para modelar los aspectos dinámicos de los sistemas. Un diagrama de interacción muestra una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos.

Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes; un diagrama de colaboración es un diagrama de interacción que destaca la organización estructural de los objetos que envían y reciben mensajes. En la presente investigación se procede al desarrollo del diagrama de secuencia del Caso de Uso Gestionar Almacenamiento, específicamente de la sección Adicionar Almacenamiento.

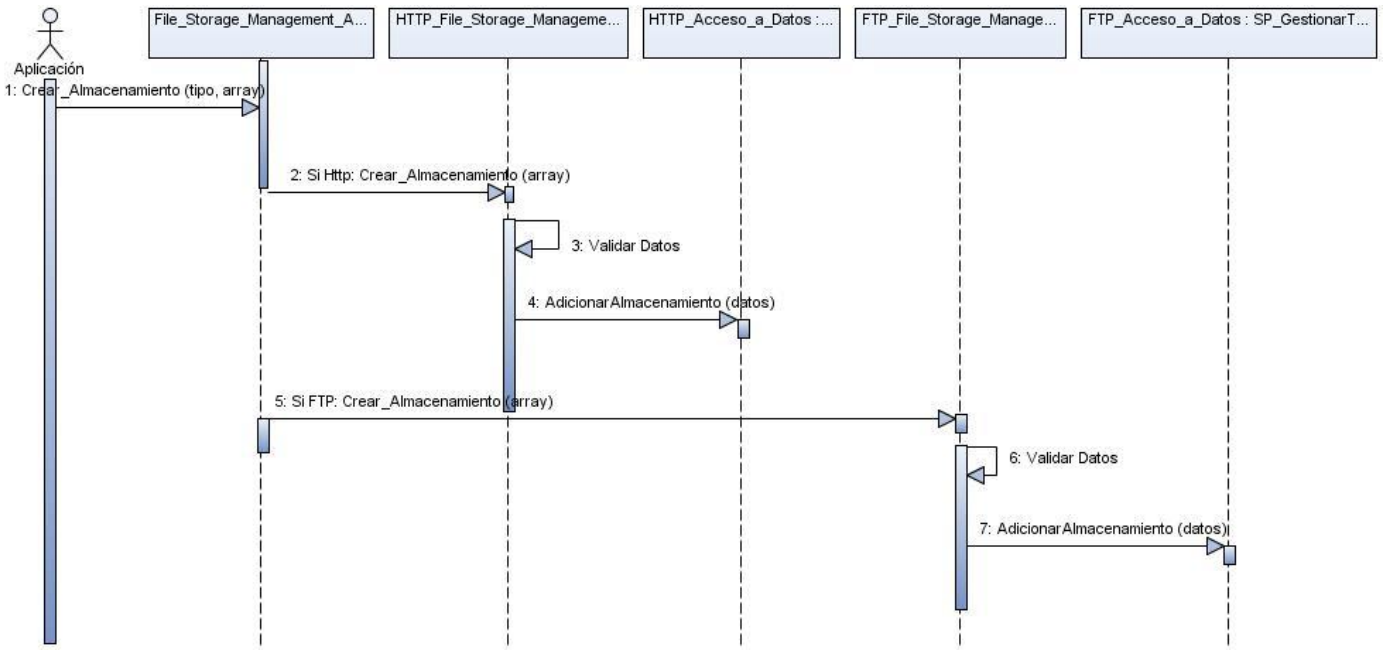


Figura 10: Diagrama de Secuencia CU "Gestionar almacenamiento", Sección Adicionar almacenamiento.

Consultar en los anexos los diagramas de secuencia correspondientes al resto de los Casos de Uso.

3.3.6 Diseño de la Base de Datos

El diseño de la base de datos es un paso crucial en el desarrollo del componente que dará solución al problema a resolver de la presente investigación. Una correcta selección de las tablas, de la información que contendrá cada una y sus relaciones, y teniendo en cuenta las prácticas adecuadas de normalización, proporcionarán un rendimiento del sistema adecuado a las exigencias del cliente.

Las Bases de datos necesitan de una definición de su estructura que le permitan almacenar datos, reconocer el contenido, y recuperar la información. A partir de las clases del diseño y las realizaciones de Casos de Uso del diseño se realiza el diseño de la base de datos.

3.3.6.1 Modelo Entidad-Relación

Luego de analizar las funcionalidades que deberá llevar a cabo el sistema propuesto, se procedió a identificar las clases persistentes, que son aquellas que contienen la información valiosa y que necesariamente debe ser almacenada en una base de datos. En el modelo Entidad-Relación se representan gráficamente las clases persistentes como tablas de la base de datos, con sus propiedades y relaciones. Durante la creación de este modelo, son realizadas las transformaciones necesarias para ajustar el diseño a la forma normal deseada.

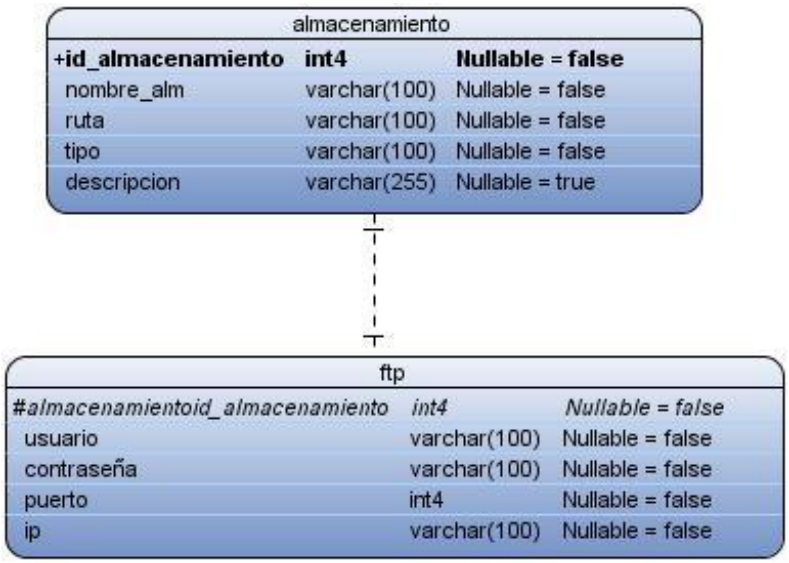


Figura 11: Modelo Entidad-Relación

3.4 Conclusiones Parciales

Con la realización del Diagrama de Clases del Análisis se llegó a una aproximación del funcionamiento real del componente. Este diagrama sirvió como punto de partida para la realización del Modelo de Diseño, el cual proporcionó un mayor detalle para la implementación específica del componente. Los patrones de diseño utilizados ayudaron eficientemente y con resultados positivos, a lograr un adecuado rendimiento del componente a desarrollar.

Capítulo 4: Características de la solución propuesta

4.1 Introducción

La intención de este capítulo es dejar expuestos los principales artefactos generados durante los flujos de trabajo Implementación y Prueba. Comenzando con el resultado del diseño durante la disciplina de Implementación se construyó el sistema en términos de componentes, ficheros de código fuente, scripts, ejecutables y similares. Mediante la disciplina Prueba se le dio un enfoque a la investigación en función de la evaluación y determinación de la calidad del producto.

4.2 Disciplina de Implementación

Durante el flujo de trabajo Implementación se describe cómo los elementos del Modelo de Diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el Modelo de Despliegue. La implementación es el centro durante las iteraciones de la Fase de Construcción y sirve para solucionar a menudo pequeños problemas tácticos relacionados con el entorno de implementación que no deberían afectar al Modelo de Diseño

4.2.1 Diagrama de componentes

Mediante el diagrama de componentes se muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. Un componente es la parte modular de un sistema, desplegable y reemplazable que encapsula implementación, un conjunto de interfaces y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc.). Son las piezas reutilizables de alto nivel a partir de las cuales se pueden construir los sistemas.

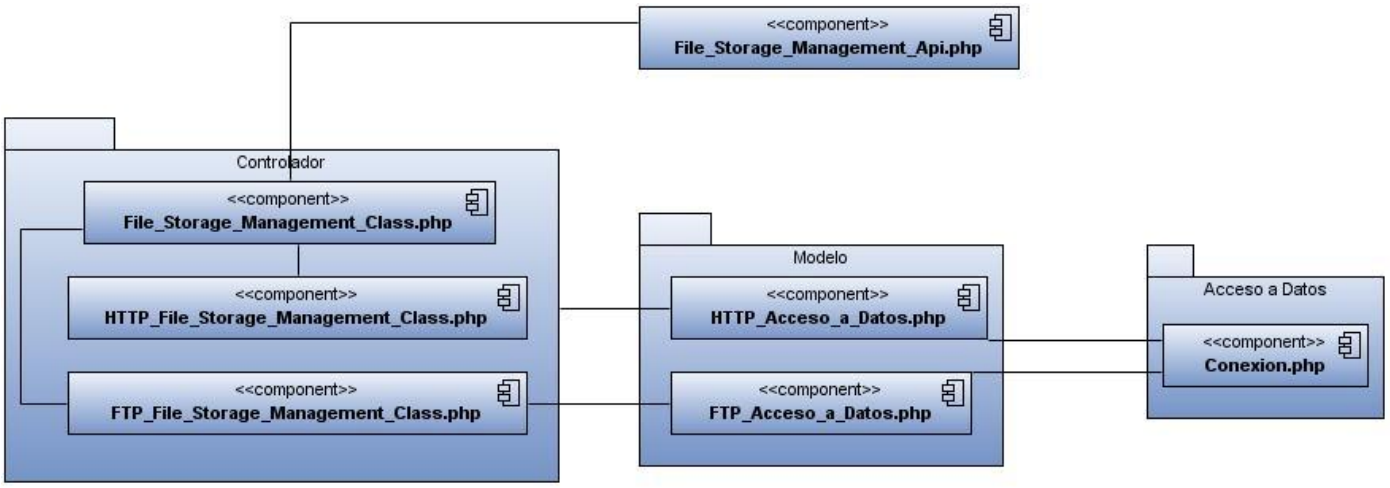


Figura 12: Diagrama de componentes

4.2.2 Diagrama de despliegue

El Modelo o Diagrama de Despliegue se utiliza para obtener los elementos de configuración del procesamiento y las conexiones entre esos elementos. Describe la arquitectura física del sistema durante la ejecución, en términos de Nodos, Dispositivos y Conectores.

- Los **Nodos** son objetos físicos que existen en tiempo de ejecución, y que representan algún tipo de recurso computacional (capacidad de memoria y procesamiento), por ejemplo computadoras con procesadores u otros dispositivos como impresoras, lectoras de códigos de barras, dispositivos de comunicación etc.
- Los **Dispositivos:** Nodos generalmente estereotipados para identificar el tipo de dispositivo, que no cuentan con capacidad de procesamiento en el nivel de abstracción que se modela.
- Los **Conectores** expresan algún tipo de ruta de comunicación entre los nodos, los cuales intercambian objetos o envían mensajes a través de esta ruta. El tipo de comunicación se identifica con un estereotipo que indica el protocolo de comunicación o la red.

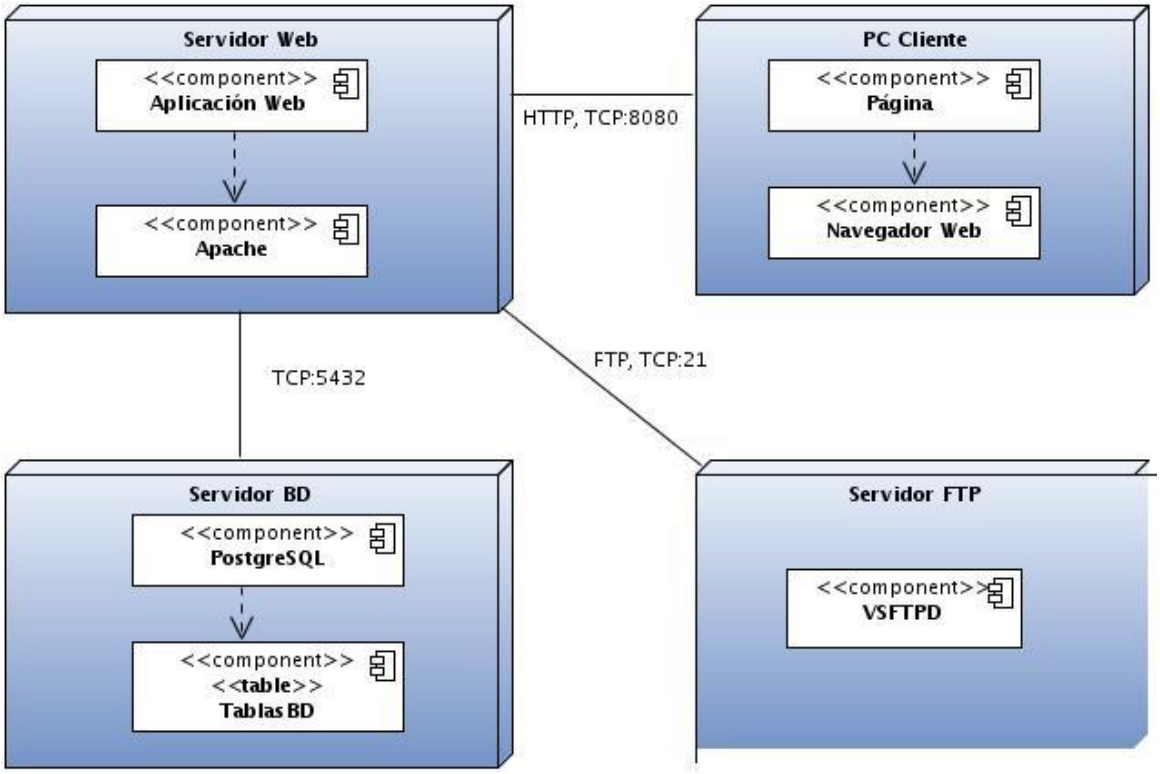


Figura 13: Diagrama de despliegue

Descripción de los Nodos

PC Cliente: Nodo cliente mediante el cual el usuario podrá acceder a la aplicación a través de un navegador web.

Servidor Web: Nodo servidor donde se ejecuta la aplicación web alojada en el servidor web Apache y que además funcionara como almacén local.

Servidor Ftp: Nodo servidor que almacena ficheros multimedia al cual se accede mediante el protocolo FTP.

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

4.3 Disciplina de Prueba

La disciplina de Prueba está enfocada principalmente en la evaluación y determinación de la calidad del producto. En esta disciplina el sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. La creciente inclusión del software como un elemento más de muchos sistemas y la importancia de los costos asociados a un fallo del mismo, han motivado la creación de pruebas más minuciosas y bien planificadas.

Al componente desarrollado se le aplicaron varias pruebas para validar el cumplimiento de los requisitos enunciados por el cliente, dar una indicación de la calidad del producto y comprobar que las consultas hechas a la base de datos respondieran eficientemente y en el menor tiempo posible. Para lograr lo anteriormente expresado, fueron realizadas Pruebas de Caja Negra.

4.3.1 Pruebas de Caja Negra

Estas pruebas son las 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. Se centran principalmente en los requisitos funcionales del software y permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

4.3.1.1 Casos de Prueba

Un caso de prueba permite detallar la forma en que se va a probar el sistema, incluyendo los datos de entrada con las que se realizará la prueba correspondiente, las condiciones de ejecución y resultados obtenidos.

Deben verificar:

- Si el producto satisface los requerimientos del usuario, tal y como se describe en las especificación de los requerimientos.
- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

4.3.1.1.1 Caso de Prueba para el Caso de Uso del Sistema: "Gestionar almacenamiento"

El Caso de Uso se inicia cuando desde la aplicación se desea adicionar un nuevo almacenamiento, editar, eliminar o listar los almacenamientos de un mismo tipo. Termina con la creación, modificación o eliminación del almacenamiento o con un mensaje de proceso fallido.

Nombre de la Sección	Escenarios de la Sección	Descripción de la funcionalidad
SC 1: Adicionar almacenamiento	EC 1.1: Adicionar almacenamiento http satisfactoriamente.	La aplicación muestra un formulario para entrar los datos asociados al almacenamiento, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.2: Adicionar almacenamiento http falla.	La aplicación muestra un formulario para entrar los datos del almacenamiento. Una vez enviados por el usuario el sistema le muestra un mensaje informando que estos no son correctos.
	EC 1.3: Adicionar almacenamiento ftp satisfactoriamente.	La aplicación muestra un formulario para entrar los datos asociados al almacenamiento, el usuario entra los datos correspondientes y los envía; estos se adicionan satisfactoriamente.
	EC 1.4: Adicionar almacenamiento ftp falla.	La aplicación muestra un formulario para entrar los datos del almacenamiento.

		Una vez enviados por el usuario el sistema le muestra un mensaje informando que estos no son correctos.
SC 2: Editar Almacenamiento	EC 2.1: Editar almacenamiento satisfactoriamente	La aplicación muestra los datos del almacenamiento seleccionado 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 almacenamiento falla.	La aplicación muestra los datos del almacenamiento seleccionado 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: Eliminar Almacenamiento	EC 3.1: Eliminar Almacenamiento satisfactoriamente.	El usuario selecciona el almacenamiento a eliminar y el almacenamiento es eliminado de la base de datos satisfactoriamente.
	EC 3.2: Eliminar Almacenamiento falla.	El almacenamiento no se elimina de la base de datos.
SC 4: Listar Almacenamiento	EC 4.1	El sistema muestra los almacenamientos http o ftp existentes según el caso.
	EC 4.2	No se muestra ningún almacenamiento ya que no existen o no se pudo conectar con la base de datos.

Tabla 2: Caso de prueba para el CU: "Gestionar almacenamiento"

Descripción de las variables

No	Nombre de Campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Este campo admite letras, números y los caracteres punto, guión alto (-), guión bajo (_) y espacio sin importar mayúsculas y minúsculas. Hace referencia al nombre del almacenamiento.
2	Ruta	Campo de texto	No	Este campo hace referencia a la ruta del almacenamiento, debe comenzar con / y ser un directorio valido dentro de la computadora.
3	IP	Campo de texto	No	Este campo admite 4 combinaciones de números separados por punto, las cuales tienen que ser menor que 255 y mayor que 0.
4	Puerto	Campo de texto	No	Este campo solo admite números y hace referencia al puerto por el que se conectará la aplicación al ftp.
5	Usuario	Campo de texto	No	Este campo hace referencia al nombre de usuario para conectarse al ftp.
6	Contraseña	Password	No	Este campo hace referencia a la contraseña correspondiente al nombre de usuario indicado para conectarse al ftp.
7	Guardar	Botón	No	Permite Guardar los datos

				introducidos por el usuario.
8	Editar	Vinculo	No	Permite acceder a la interfaz para editar los datos de un almacenamiento.
9	Eliminar	Vinculo	No	Permite eliminar un almacenamiento.
10	Explorar	Vinculo	No	Permite explorar el contenido de un almacenamiento.

Tabla 3: Caso de prueba para el CU: "Gestionar almacenamiento". Descripción de las variables.

Matriz de Datos

Id del escenario	Nombre almacenamiento	Ruta almacenamiento	Respuesta del sistema	Resultado de la prueba
EC 1 Adicionar Almacenamiento http	(Almacén) V	(/home/krysna/Escritorio/almacén) V	Se adiciona el almacenamiento	Satisfactoria
	(Almacen1) V	(/home/krysna/Escritorio/almacén) V	Se muestra un mensaje indicando que la ruta ya existe	Satisfactoria
EC 2 Adicionar Almacenamiento http falla.	(Almacen1) V	(/home/krysna/Escritorio/pepe) I	Se muestra un mensaje indicando que la ruta no es correcta.	Satisfactoria
	($ Almacén) I	(/home/krysna/Escritorio/almacén) V	Se muestra un mensaje indicando que el nombre no es correcto.	Satisfactoria
	($ Almacén) I	(/home/krysna/Escritorio/pepe) I	Se muestra un mensaje indicando	Satisfactoria

			que el nombre y la ruta no son correctos	
	(nada) I	(/home/krysna/Escritorio/almacén) V	Se muestra un mensaje indicando que el nombre no puede ser vacío.	Satisfactoria
	(Almacen1) V	(nada) I	Se muestra un mensaje indicando que la ruta no puede ser vacía.	Satisfactoria
	(nada) I	(nada) I	Se muestra un mensaje indicando que el nombre y la ruta no pueden ser vacíos.	Satisfactoria

Id del escenario	Nombre almacenamiento	Ruta almacenamiento	IP	Puerto	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC 1 Adicionar Almacenamiento ftp	(Almacén) V	(/krysna) V	(10.34.15.25) V	(2100) V	(ftpuser) V	(admin videoweb) V	Se adiciona el almacenamiento	Satisfactoria
EC 2 Adicionar Almacenamiento ftp falla.	(Almacén) V	(/krysna) V	(10.34.15.25) V	(2100) V	(ftpuser) V	(admin videoweb) V	Se muestra un mensaje indicando que no se pudo	Satisfactoria

							conectar con el servidor	
	(Almacén%) I	(/krysna) V	(10.34.15.25) V	(2100) V	(ftpuser) V	(adminvideoweb) V	Se muestra un mensaje indicando que el nombre no es correcto.	Satisfactoria
	(Almacén) V	(/carpeta) I	(10.34.15.25) V	(2100) V	(ftpuser) V	(adminvideoweb) V	Se muestra un mensaje indicando que la ruta no es correcta.	Satisfactoria
	(Almacén) V	(/krysna)V	(10.34.300.25) I	(2100) V	(ftpuser) V	(adminvideoweb) V	Se muestra un mensaje indicando que el ip no es correcto.	Satisfactoria
	(Almacén) V	(/krysna) V	(10.34.15.25) V	(abcd) I	(ftpuser) V	(adminvideoweb) V	Se muestra un mensaje indicando que el puerto no es correcto.	Satisfactoria
	(nada) I	(/krysna) V	(10.34.15.25) V	(2100) V	(ftpuser) V	(adminvideoweb) V	Se muestra un mensaje indicando que el	Satisfactoria

							nombre no puede ser vacío.	
	(Almacén) V	(nada) I	(10.34.15.25) V	(2100) V	(ftpuser) V	(admin videoweb) V	Se muestra un mensaje indicando que la ruta no puede ser vacía.	Satisfactoria
	(Almacén) V	(/krysna) V	(10.34.15.25) V	(nada) I	(ftpuser) V	(admin videoweb) V	Se muestra un mensaje indicando que el puerto no puede ser vacío.	Satisfactoria
	(Almacén) V	(/krysna)V	(nada) I	(2100) V	(ftpuser) V	(admin videoweb) V	Se muestra un mensaje indicando que el ip no puede ser vacío.	Satisfactoria

Tabla 4: Caso de Prueba para el CU: "Gestionar almacenamiento", Sección Adicionar almacenamiento. Matriz de datos.

Ver anexos para consultar los casos de prueba del resto de los Casos de Uso del Sistema.

4.3.2 Resultados de las Pruebas

El resultado de las pruebas realizadas fue satisfactorio demostrando que el componente cumple con los requisitos y las necesidades del Departamento de forma eficiente. Se evidenció que las respuestas del

sistema son las esperadas y que coinciden con las definidas en la descripción textual de cada uno de los Casos de Uso del Sistema.

4.4 Conclusiones Parciales

Durante el Capítulo 4 y final del presente trabajo de diploma quedaron expuestos los principales artefactos de los flujos de trabajo: Implementación y Prueba. Se arribó a la conclusión que mediante las pruebas hechas a la aplicación, el componente cumple de manera satisfactoria con las funcionalidades identificadas durante el proceso de desarrollo del software llevado a cabo.

Conclusiones

El componente para la gestión de ficheros que podrá ser integrado a las aplicaciones web del Departamento de Señales Digitales constituye una herramienta eficiente y de gran utilidad para los proyectos del departamento.

Luego de la realización de las tareas de la investigación científica, es posible afirmar que se cumplieron los objetivos propuestos a lo largo de este trabajo de diploma. Durante la presente investigación se ha arribado a las siguientes conclusiones:

- Las herramientas, metodología de desarrollo y lenguajes utilizados, brindaron el soporte necesario para lograr un producto con los requerimientos deseados, además de proporcionarle al mismo una calidad y rendimiento acordes a las exigencias necesarias para su correcto funcionamiento.
- La entrevista realizada a los líderes de proyecto dio paso al surgimiento de los primeros requisitos funcionales del componente.

Como es posible notar, como resultado de este trabajo de diploma se otorgará al Departamento de Señales Digitales de una herramienta inexistente hasta el momento y que puede ser de gran utilidad para la gestión anteriormente mencionada. Es necesario destacar que el componente fue desarrollado utilizando básicamente herramientas de software libre, entre las cuales se encuentran el PostgreSQL, utilizado como gestor de base de datos, y el entorno de desarrollo Integrado NetBeans, logrando la libertad tecnológica del producto y cumpliendo con los lineamientos de producción de software de la Universidad de Las Ciencias Informáticas (UCI) y el país.

Recomendaciones

El objetivo general de la investigación realizada ha sido logrado y por tanto se ha creado un componente con las funcionalidades necesarias para facilitar el proceso de gestión de ficheros multimedia para las aplicaciones web del Departamento de Señales Digitales. Sin embargo es necesario destacar que durante el desarrollo del mismo se han identificado ciertas mejoras que podrían implementarse en un futuro en aras de darle una mayor efectividad y utilidad al producto obtenido. Es por ello que se recomienda:

- Ampliar la funcionalidad del componente incluyendo otros protocolos para realizar la transferencia de ficheros como el protocolo SSH.
- Continuar investigando sobre el tema relacionado con los procesos de transferencia de ficheros enfatizando en la transferencia mediante protocolos utilizados en software libre.
- Que se le aplique pruebas de tensión y seguridad, una vez que el componente este integrado a alguna aplicación web, para verificar cómo funciona el mismo antes situaciones anormales y que los datos solo sean accedidos por las personas que cuentan con los privilegios necesarios.

Bibliografía Citada

1. 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.
2. Andrade, Luisa. 2009. *Servidor de Educación Matemática "una empresa docente", Universidad de los Andes, Bogotá, Colombia.* 2009.
3. Aspuru, Jorge García Ochoa de. 2008. *El protocolo SSH. Doctorado en Ciencias de la Computación – ESIDE. .* 2008.
4. AU, Stella. 2007. *Paradigma Visual para UML Enterprise Edition. [en línea] Javalobby.org. .* 2007.
5. Bachmann, F., y otros. 2000. *Volumen II: Technical concepts of component-based software engineering. Technical report, Software Engineering Institute, Carnegie Mellon Univ.* 2nd edition. 2000.
6. C.Wallnau., A. W. Brown and K. 1998. *The current state of CBSE. IEEE Software, 15(5):37-46.* 1998.
7. cad.com.mx. 2010. cad.com.mx. [En línea] 2010. [Citado el: 15 de 10 de 2010.] http://www.cad.com.mx/comparacion_entre_samba_vs_windows_2003.htm.
8. documentalistaenredado.net. www.documentalistaenredado.net. [En línea] [Citado el: 21 de 10 de 2010.] http://www.documentalistaenredado.net/31/software-para-la-gestin-de-archivos-multimedia.
9. Jacobson, Ivar y Booch, Grady y Rumbaugh, James. 2004. *El proceso unificado de desarrollo de software.* s.l.: Félix Varela, 2004. Vol. I.
10. JBarret, Daniel y Silverman, and Richard E. 2001. *SSH, the Secure Shell The Definitive Guide.* 2001.
11. Larman, Craig. 2007. *UML y Patrones.* 2007.
12. Martínez, Rafae. 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.
13. Microsoft. 2010. Microsoft. [En línea] 2010. [Citado el: 17 de 10 de 2010.] http://www.microsoft.com/latam/technet/mediana/25-50/soluciones/wndwstrg_14.mspx.
14. Potencier, Fabien y Zaninotto, François. 2008. *Symfony la guía definitiva.* 2008.

Bibliografía Citada

15. Pressman, Roger. 2003. *Ingeniería de Software, un enfoque práctico.* New York: MacGraw-Hill, 2003.
16. REA. 2010. Diccionario Real de la Lengua Española. [En línea] 2010. [Citado el: 20 de 10 de 2010.] http://buscon.rae.es/draeI.
17. Reynolds, Postel &. 2000. RFC 959: Referencia que define las características actuales del protocolo FTP. [En línea] 2000. [Citado el: 17 de 10 de 2010.] http://www.rfc-es.org/rfc/rfc0959-es.txt.
18. Reynoso, Carlos Billy. 2004. *Introducción a la Arquitectura de Software.* Buenos Aires, Argentina: s.n., 2004.
19. Samba. 2010. Sitio Web Oficial de Samba. [En línea] 2010. [Citado el: 23 de 10 de 2010.] www.samba.org.
20. Sims_Herzum. 2000. *Business component factory: A comprehensive overview of component-based development for the enterprise.* 2000.
21. Szyperski., C. 2002. *Component software: Beyond object-oriented programming. Addison-Wesley Pub Co, 2da edición.* 2002.
22. UH, Fac. Bioinformática de la. 2010. Facultad de Bioinformática de la Universidad de La Habana. [En línea] 2010. [Citado el: 1 de 11 de 2010.] http://fbio.uh.cu/sites/bioinfo/glosario.html.
23. Yerovi, Carlos Fabián Moyano y Yánez, Verónica Alexandra Villa. 2010. *Análisis de seguridad de los protocolos de internet (TCP/IP) y su prevención, aplicado a los servidores de la academia CISCO (ESPOSCH).* 2010.
24. ZANINOTTO, F. y F., POTENCIER. 2007. *The Definitive Guide to symfony (1ra edición). apress. pp. 486. ISBN 978-1-59059-786-6.* 2007.

Bibliografía Consultada

- Desarrollo de Software Basado en Componentes Jonás A. Montilva C.1, Nelson Arapé2 y Juan Andrés Colmenares2. 1Universidad de Los Andes Facultad de Ingeniería Escuela de Ingeniería de Sistemas Departamento de Computación Mérida, Venezuela, 2Universidad del Zulia Facultad de Ingeniería Instituto de Cálculo Aplicado Maracaibo, Venezuela.
- Dr. C. María Elena Guardo García. I.S.C.F. Manuel Fajardo. Facultad Cultura Física. Matanzas. Los componentes del diseño teórico de la investigación científica. Una reflexión praxiológica.
- Ing. Lázaro J. Ramos Alfonso y MScMiriel Martín Mesa, Implementación de un Servidor Samba con autenticación LDAP como alternativa Libre a los Servidores de Dominio Windows. Capítulo 2. Samba + Ldap como Controlador de dominio primario (PDC).
- Modelado de Sistemas com UML, Popkin Software and Systems.
- Martin Fowler, Kendall Sccott, "UML Gota a Gota", 1999.
- Rafael Orea Area, Xabiel G. Pañeda, Roberto García, David Melendi, Sergio Cabrero. Departamento de Informática, Universidad de Oviedo, Gijón, España. *V Congreso Iberoamericano de Telemática. CITA 2009.* Análisis y caracterización de la reproducción de vídeo con mediacenters en redes LAN.
- Steven M. French, “A New Network File System is Born: Comparison of SMB2, CIFS and NFS”. IBM. Samba Team.

Anexos

Anexo 1: Glosario de Términos

Apache: Servidor web HTTP de código abierto con versiones para varias plataformas incluyendo Unix (BSD, GNU/Linux, etc.), Windows y Macintosh.

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

GNU: Proyecto para la creación de un sistema operativo de libre distribución. Como núcleo del sistema se suele usar el de Linux, dando lugar al conjunto llamado “GNU/Linux”.

Hardware: Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen de los programas (software) porque son tangibles.

IP: (Internet Protocol, Protocolo de Internet). Protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.

ISO: (International Organization for Standardization, Organización Internacional para la Normalización). Organismo encargado de promover el desarrollo de normas internacionales de fabricación, comercio y comunicación para todas las ramas industriales a excepción de la eléctrica y la electrónica.

Java: Lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90.

Linux: Núcleo o kernel del sistema operativo libre denominado GNU/Linux. Lanzado bajo la licencia pública general (GPL - General Public License) de GNU y desarrollado gracias a contribuciones provenientes de todo el mundo.

Media: Contenido digital, entiéndase audio o video, para informar o entretener al usuario.

OSI: (Open System Interconnection, Interconexión de Sistemas Abiertos). Marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

Software: Equipamiento lógico o soporte lógico de un computador digital, comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema (hardware).

SQL: (Structured Query Language, Lenguaje de Consulta Estructurado). Lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

TCP: (Transmission Control Protocol, Protocolo de Control de Transmisión). Protocolo de comunicación orientado a conexión y fiable ubicado en el nivel de transporte del modelo OSI, es uno de los protocolos fundamentales en Internet.

Tecnología Streaming: La tecnología de streaming permite escuchar y visualizar los archivos mientras se están descargando y de esta forma aligerar la descarga y ejecución de audio y vídeo en la web. Si no se utiliza esta tecnología para mostrar un contenido multimedia en la Red, es necesario descargar el archivo entero primero en la computadora para luego ejecutarlo, y finalmente ver y escuchar el archivo.

Telnet: (TELecommunication NETwork). Es el nombre de un protocolo de red (y del programa informático que implementa el cliente), que sirve para acceder mediante una red a otra máquina, para manejarla remotamente.

URL: (Uniform Resource Locator, Localizador Uniforme de Recurso): Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos como documentos e imágenes en Internet para su localización.

Windows: Familia de sistemas operativos desarrollados y comercializados por Microsoft.

WWW: (World Wide Web, Telaraña o Malla Mundial). Sistema de información distribuido con mecanismos de hipertexto. Es el universo de servidores HTTP, que permiten mezclar texto, gráficos y archivos de sonido juntos.

XP: (eXtremeProgramming, Programación Extrema). Enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia. Es el más destacado de los procesos ágiles de desarrollo de software.

Anexo 2: Entrevista realizada a los líderes de proyecto del Departamento de Señales Digitales.

1. ¿Qué importancia considera usted que pueda tener un componente que permita gestionar los ficheros multimedia con los que trabajan las aplicaciones web del Departamento de Señales Digitales?
2. En caso de desarrollado dicho componente ¿Con qué funcionalidades debería contar para satisfacer las necesidades del proyecto al cual usted pertenece?