Universidad de las Ciencias Informáticas
Facultad 2

Título: “Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles”.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autor(es): Karelys Franco Pérez.

Alfonso de Jesús Pantoja Rosales.

Tutor(es): MSc. Bárbara Laborí de la Nuez.

Ing. Darién Jesús Álvarez de la Cruz.

Ciudad de La Habana, Junio del 2011.

PENSAMIENTO

Saber no es suficiente, debemos aplicar. Desear no es suficiente, debemos hacer.

Johann W. Von Goethe

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al <nombre área> de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los **_16__** días del mes de **__Junio___** del año **__2011__**.

Karelys Franco Pérez

_____

Firma de Autor

Alfonso de Jesús Pantoja Rosales

_____

Firma de Autor

MSc. Bárbara Laborí de la Nuez

_____

Firma de la Tutora

Ing. Darien Jesús Alvarez de la Cruz

_____

Firma del Tutor

DATOS DE CONTACTO

Tutor: Bárbara Laborí de la Nuez.
Profesión: Máster en Informática.
Área: Facultad 2. Dpto. Sistemas Digitales.
Años de Graduado: 20.
Labor que Desempeña: Profesor.

Tutor: Ing. Darien Jesús Alvarez de la Cruz (dalvarez@uci.cu).
Profesión: Ingeniero en Ciencias Informáticas.
Área: Facultad 2. Centro de Telemática.
Años de graduado: 4.
Labor que desempeña: Jefe de Dpto. de Telecomunicaciones del Centro de Telemática, Facultad 2 de la UCI.

AGRADECIMIENTOS

Agradecemos a todas a todas las personas que nos ayudaron en la realización de la tesis, a nuestros tutores, a los compañeros del proyecto, a todas las personas con que hemos compartidos en estos 5 años, y nuestro agradecimiento especial para la familia en especial a nuestros padres pero todos el apoyo que nos han brindado y que gracias a ellos y la revolución hoy estamos optando por este titulo de ingenieros en ciencias informáticas.

DEDICATORIA

Dedicamos este trabajo de diploma a nuestros familiares en especial a nuestros padres pero toda la ayuda y apoyo que nos han dado siempre y que gracias a ellos hoy podamos alcanzar este sueño.

RESUMEN

En la actualidad las tecnologías inalámbricas y principalmente la telefonía celular han presentado un gran desarrollo en estos últimos años, lo cual ha convertido al teléfono celular en una herramienta primordial para la comunicación. Actualmente la mayoría de las aplicaciones de telefonía celular ofrecen un gran número de valores agregados, la comunicación e integración entre aplicaciones es un ejemplo de que la telefonía móvil se encuentra en una etapa progresiva. El Departamento de Telecomunicaciones del Centro de Telemática de la Facultad 2 en la Universidad de la Ciencias Informáticas (UCI), es el encargado del desarrollo y puesta en marcha de productos relacionados con la telefonía móvil o celular; sin embargo a pesar de los esfuerzos realizados, no cuenta con mecanismos de comunicación que permitan que sistemas externos puedan comunicarse e integrarse con la “Plataforma de Gestión de Contenidos para Dispositivos Móviles”.

Por estas razones se propone el desarrollo de un Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles, que tiene como objetivo proporcionar el espacio de trabajo necesario para cumplir con las funciones relacionadas con la comunicación e integración de sistemas externos con la plataforma, realizando una serie de actividades que permitan establecer las comunicaciones con integridad y seguridad.

PALABRAS CLAVE:

Telefonía Celular, UCI, Mecanismo de Comunicación, Sistemas Externos, Plataforma de Gestión de Contenidos para Dispositivos Móviles.

ÍNDICE

Índice

INTRODUCCIÓN

Las tecnologías inalámbricas en los últimos años han presentado un gran desarrollo, principalmente en la rama de la telefonía celular. El teléfono celular se ha convertido en una herramienta primordial para la gente común y de negocio a nivel mundial; debido a los múltiples servicios que proporciona. A pesar que su principal función es la comunicación de voz, como el teléfono convencional, su rápido desarrollo ha incorporado otras funciones como son cámara fotográfica, agenda, acceso a Internet, reproducción de video, reproductor mp3, servicio de mensajes SMS[1], entre otros.

Cuba no se encuentra excluida en cuanto al desarrollo de la telefonía celular, pues actualmente cuenta con una modernizada red de telefonía digital que cubre todo el país. La Empresa de Telecomunicaciones de Cuba (ETECSA); es la encargada de expandir y desarrollar las comunicaciones móviles en el país, dentro de la cual se encuentra inmersa la unidad de negocios móviles CUBACEL.

El desarrollo de la telefonía celular y la informática en Cuba ha sido impetuoso en los últimos años debido a la adquisición de nuevas tecnologías y la inserción de otras en el mundo moderno. La Universidad de las Ciencias Informáticas (UCI) juega un papel importante en el desarrollo de la Industria Cubana del Software, y en la materialización de los proyectos asociados al programa cubano de informatización.

Con la creciente demanda de productos relacionados con la telefonía móvil o celular, se requiere la búsqueda de soluciones informáticas que permitan establecer canales de comunicación para la transmisión de datos, teniendo en cuenta el tamaño, la integridad y seguridad en el envío.

El Departamento de Telecomunicaciones[2] del Centro de Telemática en la UCI, desea brindar mecanismos de comunicación e integración con sistemas externos[3], de esta manera proveer el intercambio de información con integridad, fiabilidad y seguridad. Como propuesta del Dpto. se planteó la necesidad de implementar el Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles que posibilite que sistemas externos se comuniquen e integren con dicha plataforma.

[1] SMS (Short Messages Standard) es un servicio disponible en los teléfonos móviles que permite el envío de mensajes cortos.
[2] Departamento de Telecomunicaciones es el encargado del desarrollo y puesta en marcha de productos relacionados con la telefonía móvil o celular.
[3] Son plataformas o sitios web pertenecientes a instituciones como Artex. Ejemplo de ellos es el sitio web perteneciente Artex MallCubano.com.

Esto condujo a plantearse el siguiente **problema a resolver**: ¿Cómo llevar a cabo la comunicación e integración de la "Plataforma de Gestión de Contenidos para Dispositivos Móviles" con sistemas externos?

Acorde con el problema planteado se define como **objeto de estudio**: la comunicación e integración de aplicaciones; teniéndose además como **campo de acción**: el módulo de integración de la plataforma.

Con el propósito de dar solución al problema anteriormente planteado se establece como **objetivo general:** Desarrollar un Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

Se considera como **idea a defender**: el desarrollo de un módulo de integración, permitirá que sistemas externos puedan comunicarse e integrarse con la "Plataforma de Gestión de Contenidos para Dispositivos Móviles".

Para dar solución al problema y cumplir con el objetivo se proponen las siguientes **tareas de investigación:**

- Realización de un estudio de la comunicación e integración entre aplicaciones.
- Investigación de la metodología de desarrollo de software, herramientas y lenguaje de programación óptimo para el desarrollo del sistema.
- Realización de un estudio de la "Plataforma de Gestión de Contenidos para Dispositivos Móviles".
- Realización de pruebas de software que permitan verificar y revelar la calidad del subsistema.

En el transcurso de la investigación se utilizaron como guía los Métodos de la Investigación Científica.

Métodos Teóricos:

Analítico-Sintético: Sirvió para realizar el procesamiento de toda la información referente a la comunicación entre aplicaciones aisladas, pudiendo sintetizar y diferenciar cada una de ellas y de esta forma enfocarlas hacia la investigación.

Histórico-Lógico: Posibilitó conocer los antecedentes y tendencias actuales de los mecanismos de comunicación para la integración entre aplicaciones aisladas; pudiendo identificar los mecanismos lógicamente óptimos a implementar en el módulo de integración.

Modelación: Permitió modelar los procesos que se establecen para la comunicación e integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles con sistemas externos.

Métodos Empíricos:

Experimento: Posibilitó mediante un modelo de prueba, verificar las funcionalidades del subsistema en cada iteración del desarrollo del software; evidenciándose la correcta implementación de los mecanismos de comunicación, identificados para el desarrollo del módulo de integración.

La estructura del documento se especifica a continuación:

Capítulo I Fundamentación Teórica: Contiene el estudio sobre la comunicación entre aplicaciones; la investigación sobre cada una de las tecnologías, metodologías, lenguajes, conceptos y herramientas utilizadas para el desarrollo del módulo de integración.

Capítulo II Características del Sistema: Contiene la descripción de los principales conceptos mediante un Modelo de Dominio, la captura de requisitos funcionales y no funcionales, así como el diagrama de casos de usos del sistema y la descripción de los mismos.

Capítulo III Diseño del Sistema: Contiene el análisis y diseño del sistema, obteniendo como resultado principal, los diagramas de clases del diseño, así como los diagramas de interacción para cada realización de los casos de usos. Muestra la descripción de la arquitectura y patrones de diseño utilizados.

Capítulo IV Implementación y Prueba: Contiene las principales características del flujo de trabajo de implementación, representando el diagrama de despliegue y el de componentes; así como la realización de casos de prueba para mitigar los posibles fallos.

Capítulo V Estudio de Factibilidad: Contiene el tiempo de duración del proyecto de software, así como la evaluación del esfuerzo, el análisis de costos y beneficios tangibles e intangibles asociados al desarrollo del módulo de integración, proporcionando una visión general de la factibilidad a la hora de realizarlo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se enuncian los principales conceptos que se tuvieron en cuenta para la investigación y desarrollo del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles. Con el desarrollo de este trabajo se pretende que el subsistema brinde información a otros usuarios (aplicaciones). Teniendo en cuenta lo antes expuesto se describen las herramientas, tecnologías y metodología a utilizar en el desarrollo de la solución.

1.2 Fundamentación del Tema

1.2.1 Necesidad de Integración entre Aplicaciones

A mediados de la década de los 90 y con la aparición de Internet y su posterior masificación a niveles jamás pensados, ha existido siempre la necesidad e inquietud por parte de las empresas desarrolladoras de software de buscar o contar con la manera de lograr la integración entre sistemas heterogéneos, tanto al software como al hardware. Para tal efecto muchas compañías fueron creando de forma individual la mejor manera de lograr la integración. Muchas empresas comenzaron una larga carrera para generar la mejor tecnología integradora de sistemas, pero a medida que la competencia se hacía cada vez más fuerte, la integración era cada vez más difícil.

Debido a la gran masificación de Internet a niveles insospechables y al gran impacto causado por las tecnologías de la información en las últimas dos décadas del siglo pasado, la manera de hacer negocios y la comunicación entre las personas y las empresas cambió de una manera rotunda. Bajo este contexto se hacía cada vez mayor la necesidad de integrar y compartir información entre distintas plataformas de software y hardware.

La UCI como máxima desarrolladora de software en el país no se encuentra ajena a la búsqueda de soluciones informáticas que posibiliten a sus productos integrarse con sistemas externos. La “Plataforma de Gestión de Contenidos para Dispositivos Móviles” producto perteneciente al Dpto. de Telecomunicaciones del Centro Telemática en la UCI, en la búsqueda de satisfacer demandas de los clientes desea brindar nuevos mecanismos de comunicación e integración con sistemas externos como un

servicio de valor agregado a sus funcionalidades, de esta manera establecer las comunicaciones con fiabilidad, integridad y seguridad.

1.2.2 Comunicación entre Aplicaciones

Con el desarrollo tecnológico que ha alcanzado la humanidad y la aparición de numerosas aplicaciones se ha hecho imprescindible la búsqueda de alternativas para establecer la comunicación entre dos o más aplicaciones aisladas, ya sea para proveer el intercambio de información, el flujo de datos o simplemente mostrar un mensaje o una notificación. Para lograr dichas comunicaciones se usan en todo el mundo diversos mecanismos, lenguajes de programación, protocolos, frameworks y se llevan a la práctica las arquitecturas más indicadas en cada caso.

1.2.3 Mecanismos de comunicación

Los mecanismos de comunicación no son más que la comunicación entre procesos; que permiten que dos o más aplicaciones distribuidas se comuniquen entre sí. A continuación se exponen algunos de los mecanismos utilizados para la comunicación entre aplicaciones con sistemas externos:

1.2.3.1 RMI (Java Remote Method Invocation)

RMI es un mecanismo ofrecido por Java para invocar un método de manera remota. Forma parte del entorno estándar de ejecución de Java y provee de un mecanismo simple para la comunicación de servidores en aplicaciones distribuidas basadas exclusivamente en Java. RMI se caracteriza por la facilidad de su uso en la programación por estar específicamente diseñado para Java. Por medio de RMI, un programa Java puede exportar un objeto, lo que significa que éste queda accesible a través de la red y el programa permanece a la espera de peticiones en un puerto TCP[4]. A partir de este momento, un cliente puede conectarse e invocar los métodos proporcionados por el objeto.

1.2.3.2 Web Service (Servicios Web)

Web Service es una pieza de software que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de

[4] TCP (Transmission Control Protocol) es uno de los protocolos fundamentales en Internet.

programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. Entre sus ventajas se encuentran:

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los servicios web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar y abiertos.(1)

1.2.3.3 CORBA (Common Object Request Broker Architecture)

CORBA es un estándar que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación de métodos remotos bajo un paradigma orientado a objetos. CORBA fue definido y está controlado por el Object Management Group (OMG) que define las APIs, el protocolo de comunicaciones y los mecanismos necesarios para permitir la interoperabilidad entre diferentes aplicaciones escritas en diferentes lenguajes y ejecutadas en diferentes plataformas, lo que es fundamental en computación distribuida.(2)

En un sentido general, CORBA *envuelve* el código escrito en otro lenguaje, en un paquete que contiene información adicional sobre las capacidades del código que contiene y sobre cómo llamar a sus métodos. Los objetos que resultan, pueden entonces ser invocados desde otro programa (u objeto CORBA) desde la red.

1.2.3.4 RMI vs. CORBA

Pero, ¿Por qué utilizar RMI y no utilizar CORBA? Sin duda, es casi obligatorio realizar una comparativa entre ambas tecnologías debido a que, efectivamente, se puede elegir entre las dos para implementar la comunicación e integración con otras aplicaciones. Numerosos artículos cubren esta discusión y prácticamente llegan a una misma conclusión: si la aplicación se desarrolla totalmente en JAVA, utiliza RMI. (3)

Por otro lado, CORBA es una tecnología asentada con el paso de los años. RMI no está en la misma situación y si se elige utilizarlo, las aplicaciones podrán seguir funcionando en el futuro pues las intenciones de Sun Microsystems[5] es seguir soportando esta tecnología.

1.2.4 Selección de los Mecanismos de Comunicación

Luego del estudio de los mecanismos de comunicación anteriormente expuestos, basándose en las características y ventajas de cada uno, se considera utilizar para el desarrollo del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles el diseño y la implementación de los mecanismos de comunicación RMI y Web Service. La principal ventaja de RMI es su implementación 100% en Java, provee además un mecanismo simple para la comunicación de servidores en aplicaciones distribuidas basadas exclusivamente en Java.

Web Service es muy práctico, puede aportar gran independencia entre la aplicación que usa el servicio web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más utilizada.

1.3 Principales Conceptos

1.3.1 XML (Extensive Markup Language)

XML es considerado como un metalenguaje de definición de documentos estructurados mediante marcas o etiquetas. Se trata de un estándar de World Wide Web Consortium cuyo objeto es crear reglas básicas para permitir el intercambio de información estructurada entre aplicaciones, y en particular, entre aplicaciones web.

1.3.2 WSDL (Web Services Description Language)

WSDL es un formato XML que se utiliza para describir servicios web. Está basado en XML y describe la forma de comunicación. Las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje. WSDL permite tener una descripción de un

[5] Sun Microsystems :empresa líder en servidores web y estaciones de trabajo, también desarrolladora de software.

servicio web. Especifica la interfaz abstracta a través de la cual un cliente puede acceder al servicio y los detalles de cómo se debe utilizar.

1.3.3 SOAP (Simple Object Access Protocol)

SOAP es un protocolo que permite la comunicación entre aplicaciones a través de mensajes por medio de Internet. Está basado en XML y es la base de los Web Services. Es independiente de la plataforma y del lenguaje. Utiliza cualquier protocolo que permita transportar mensajes de texto, siendo HTTP el más utilizado.

1.4 Metodología de desarrollo y lenguaje de modelado utilizado

La metodología es un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. (4)

1.4.1 Metodología de desarrollo de software: RUP

RUP[6] es una metodología orientada al proceso de desarrollo de software, que junto con el lenguaje unificado de especificación UML constituye las herramientas más utilizadas para el análisis, diseño, implementación, y documentación de sistemas orientados a objetos (5). Se decide utilizar ésta metodología pues genera gran cantidad de documentación, lo que posibilita mayor comprensión de todo el ciclo de vida del desarrollo del subsistema. Además permite llevar en cada iteración una versión del producto al cliente, obteniendo mejoras; logrando como resultado un producto de alta calidad.

Beneficios que aporta RUP:

- Permite mejorar la comunicación, unificando todo el equipo de desarrollo de software, brindándoles una base de conocimiento y un punto de vista de cómo desarrollar el subsistema.
- Permite desarrollar aplicaciones sacando el máximo provecho de las nuevas tecnologías, mejorando la calidad, el rendimiento, la reutilización, la seguridad y el mantenimiento del software mediante una gestión sistemática de los riesgos.

[6] RUP (Rational Unified Process) es un proceso de desarrollo de software.

RUP tiene además como objetivo guiar a los desarrolladores en la implementación y distribución eficiente de sistemas que se ajusten a las necesidades de los clientes.

1.4.2 Lenguaje Unificado de Modelado: UML

UML es un lenguaje de modelado que permite visualizar, especificar, construir y documentar los artefactos que involucran el desarrollo del subsistema. Provee beneficios significativos ayudando a construir modelos rigurosos a los que se puede dar mantenimiento; además soporta el ciclo de vida de desarrollo de software completo.

UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. (6) Debido a que UML es un lenguaje, cuenta con reglas para combinar tales elementos. Es importante destacar que un modelo UML describe lo que supuestamente hará un sistema, pero no dice cómo implementar dicho sistema.

1.4.3 Herramienta de Modelado: Visual Paradigm

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta proporciona abundantes tutoriales, demostraciones interactivas y proyectos en UML. (7)

Se decidió utilizar esta herramienta CASE para el modelado del diseño de la aplicación, ya que entre sus principales características y facilidades se encuentran: multiplataforma, producto de calidad, varios idiomas, generación de código para Java y exportación como HTML y se integra con herramientas Java como Eclipse.

1.5 Tendencias, lenguajes y tecnologías utilizadas

Uno de los primeros puntos a tener en cuenta en el desarrollo del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles, es definir los requisitos no funcionales. Atendiendo a estos se ha definido utilizar Java como lenguaje de programación utilizando como Plataforma de

Desarrollo Java 2 Enterprise Edition con la herramienta de desarrollo Eclipse, con los frameworks Spring y JUnit. Se utilizará el servidor Apache Tomcat 6.0.

1.5.1 Plataforma de desarrollo: Java 2 Enterprise Edition (J2EE)

Para el desarrollo del Módulo de Integración se seleccionó la Edición Empresarial: J2EE. Es una plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de n niveles distribuidos basándose ampliamente en componentes de software modulares ejecutados sobre un servidor de aplicaciones. Incluye varias especificaciones de APIs, tales como EJB, Servlets[7], Portlets[8], JSP[9] y varias tecnologías de servicios web.

Java 2 Enterprise Edition (J2EE) es el estándar de la industria para desarrollar aplicaciones Java portables, robustas, escalables y seguras en el lado del servidor (server-side). J2EE proporciona APIs para servicios web, modelo de componentes, gestión y comunicación que lo convierten en el estándar de la industria para implementar aplicaciones web, web 2.0 y aplicaciones con arquitectura orientada a servicios (SOA). (8)

1.5.2 Herramienta de desarrollo: Eclipse

Eclipse es una completa y potente plataforma de programación. Programa compuesto por un grupo de herramientas necesarias para un desarrollador de software. Esta plataforma es de código abierto, es posible implementar en distintos lenguajes de programación y a través de plugins se le puede agregar soportes de lenguajes adicionales. Es todo un entorno de trabajo escrito en Java capaz de adaptarse a cualquier tipo de desarrollo ya sea por el lenguaje o por la metodología, puede ser utilizado para diseñar programas de primer nivel, sitios web, componentes, entre otros elementos. Tiene atrayentes utilidades para Java, es gratuito, su instalación es muy sencilla y los errores de compilación se muestran en tiempo real subrayando el fragmento de código apropiado con una línea de color rojo.

[7] Servlets: código en Java que se ejecuta en un servidor web
[8] Portlets: son componentes modulares de las interfaces de usuario gestionadas y visualizadas en un portal web.
[9] JSP (Java Server Pages) es una tecnología web, del lado del servidor, que se usa generalmente para generar documentos HTML y XML dinámicos.

Eclipse ofrece características fundamentales entre las que se encuentran un administrador de proyectos e interfaces para el control estándar de sistemas, como CVS[10] y ClearCase. Las herramientas integradas a Eclipse operan en archivos del espacio de trabajo (workspace) del usuario. El workspace consta de uno o más proyectos donde cada uno se mapea a un directorio especificado por usuario en el sistema de ficheros. (9)

1.5.3 Frameworks

Un framework no es más que una estructura de soporte definida que permite la organización y desarrollo de otro software. Puede incluir soportes a programas, APIs y lenguajes scripting entre otros software para ayudar a desarrollar y a unir los diferentes componentes de un proyecto.

1.5.3.1 Spring

Spring es un framework de software libre para Java. Brinda soluciones muy bien documentadas, fáciles de usar y ofrece mucha libertad a los desarrolladores en Java por su diseño.

Entre sus principales características se pueden mencionar:

- Contiene y administra el ciclo de vida y la configuración de los objetos de la aplicación.
- El soporte para la programación orientada a aspectos (AOP), que permite separar la lógica del negocio de los servicios del sistema, eliminando la necesidad de tratar de forma repetitiva temas como el soporte de transacciones o la autenticación.
- Provee la mayor parte de la funcionalidad de infraestructura, es decir administración de transacciones, integración con el framework de persistencia, entre otras.
- Spring ofrece mecanismos de integración con otros frameworks y librerías, liberando al desarrollador de implementar su unión. Por lo que no es necesario basar toda la aplicación en Spring, también existe la posibilidad de trabajar con algunos de sus módulos y complementarlos con otras alternativas cuando estas sean necesarias.

[10] CVS (Concurrent Versions System) permite que varios programadores trabajen de forma colaborativa en un mismo proyecto llevando un control de las versiones de los ficheros.

1.5.3.1.1 Spring Web Services

Spring Web Services en un producto de Spring que facilita la creación de servicios web lo cual se basa en el intercambio de documentos. Se basa en servicios los cuales se establece un contrato inicialmente y posteriormente se implementa, evitando atar al contrato, como ocurre en los casos que se genera el mismo a partir de las clases Java, facilitando que se apliquen las mejores prácticas para la creación de los Web Services.

1.5.3.1.2 Spring Security

Spring Security es un framework de seguridad para las aplicaciones basadas en Spring. Proporciona una solución completa a la seguridad al implementar servicios de autenticación y autorización a recursos, tanto a niveles de peticiones web empleando filtros de Servlets; dichos filtros interceptan las peticiones hechas a los Servlets; como a nivel de invocación de métodos. Sobre la base de Spring, Spring Security aprovecha al máximo las técnicas de inyección de dependencia. Posee seguridad basada en roles, en lista de control de acceso, entre otros. Es capaz de gestionar seguridad en varios niveles: URLs que se solicitan al servidor, acceso a métodos, clases Java y acceso a instancias concretas de las clases (10). En los Web Service del módulo de integración se utilizó la seguridad a nivel de peticiones web y a nivel de métodos para el caso del servicio RMI.

1.5.3.2 JUnit

JUnit es un framework de código abierto desarrollado especialmente para crear, ejecutar y hacer reportes de estado de conjuntos de pruebas unitarias automatizadas hechas en lenguaje Java. JUnit es uno de los frameworks más populares en Java para realizar pruebas unitarias.

1.5.4 Servidor: Apache Tomcat 6.0

El Servidor Apache HTTP es un servidor Web de tecnología Open Source y para uso comercial desarrollado por la Apache Software Foundation (Fundación de Software Apache). Apache es uno de los logros más grandes del software libre por lo que diariamente gana más uso y aceptación por parte de los usuarios a nivel mundial. Funciona como un contenedor de servlets que implementa sus especificaciones y las de Java Server Page. Además de tener incorporado un compilador, que recoge las JSP en Servlets,

por lo que proporciona un entorno que permite que el código Java se ejecute juntamente con un servidor Web.

Fue escrito en Java por lo que funciona en cualquier sistema operativo que disponga de una máquina virtual Java. Apache Tomcat es desarrollado en un entorno abierto. Se ha seleccionado esta herramienta principalmente debido a su tecnología de código abierto, la cual es una característica fundamental para los sistemas que se quieren desarrollar en la universidad, además Apache es considerado como uno de los mejores servidores web de internet.

Conclusiones

En este capítulo se realizó un análisis de los principales conceptos que deben conocerse para comprender la solución que se propone. Incluye un estudio de los mecanismos más utilizados para la comunicación entre aplicaciones. De igual manera se fundamentó la selección de cada tecnología y herramienta a utilizar en el desarrollo del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

CAPÍTULO 2: CATACTERÍSTICAS DEL SISTEMA

Introducción

En este capítulo se presentan los aspectos referentes a la fase inicial de la metodología de desarrollo RUP ya que el objetivo de esta fase inicial es alcanzar los objetivos del ciclo de vida del proyecto. En el presente capítulo se realizará la propuesta del sistema a desarrollar, se presentará para comprender y gestionar la complejidad del mismo, el Modelo del Dominio; se expondrán los requisitos funcionales y no funcionales, los cuales permitirán darle solución a la problemática. Se describirá además desde un punto de vista del usuario las formas de acciones y reacciones del comportamiento del sistema mediante el Modelo de Caso de Uso del Sistema.

2.1 Problema y situación problémica

Debido que el Dpto. de Telecomunicaciones del Centro Telemática en la UCI busca mejorar sus servicios y de esta forma la satisfacción de sus clientes, se ha propuesto desarrollar un Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles, que posibilite a sistemas externos, comunicarse e integrarse, y proveer contenidos a dicha plataforma; como valor agregado a sus funcionalidades; de esta manera posibilitar que sistemas externos puedan consumir sus servicios con confidenciabilidad, disponibilidad e integridad.

Dicho módulo de integración deberá contar con los mecanismos de comunicación RMI y Web Service, combinando la integridad con la seguridad en la transmisión de la información. Con estos tipos de mecanismos se esperan los siguientes beneficios:

- Manejar los posibles fallos que puedan ocurrir de conexión.
- Los sistemas externos podrán acceder a los servicios que ofrece la plataforma de una manera segura.
- Permitirá el envío de contenido por parte de los sistemas externos de una modo integro.
- Proporcionar nuevas posibilidades y servicios para el Dpto. de Telecomunicaciones.
- Permitirá que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.

2.2 Objeto de Automatización

El Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles tiene como objetivo la comunicación e integración con múltiples sistemas externos, permitiéndoles acceder a la información que brinda la plataforma. Como resultado se debe desarrollar un módulo de integración que sea capaz de permitir la comunicación e integración con varios sistemas soportados en disímiles plataformas que utilizan diferentes protocolos de comunicación permitiendo que los servicios de la plataforma puedan ser consumidos con seguridad.

2.3 Sistema propuesto

Se propone un módulo de integración que entre sus funcionalidades permita la comunicación e integración de aplicaciones externas con la “Plataforma de Gestión de Contenidos para Dispositivos Móviles” mediante los mecanismos de comunicación RMI y Web Service. Este módulo de integración posibilitará brindarles a sus clientes servicios de autenticación, modificación de datos; búsqueda, actualización y eliminación de una categoría, subcategoría o un contenido, la obtención de reporte; así como la inserción de contenido lo cual permitirá proveer a la plataforma, contenidos como imágenes, videos, música, películas entre otros; sin limitación de tamaño.

Insertar Contenido es la principal funcionalidad con que contará el módulo de integración; esto se realiza con la utilización de hilos de ejecución, donde existe un hilo principal el cual está siempre a la escucha de alguna petición de envío de contenido o de consumir algún servicio, por sistemas externos; por cada envío de contenido se crea un hilo el cual se activará por el hilo principal una vez que los hilos en cola; agregados anteriormente; hallan terminado su ejecución. Para insertar cualquier contenido en la plataforma se validará que no exista en el directorio especificado otro archivo con el mismo nombre; en caso de existir; se modificará el nombre del contenido que se desee copiar. La copia de un archivo se realiza llevándolo a un arreglo de bytes y luego restaurándolo en la posición destino. El archivo a insertar es fragmentado en segmentos de arreglos de bytes; permitiendo enviar archivos de disímiles tamaños sin saturar la memoria; permite además configurar los tamaños de los fragmentos a enviar. Se valida que no se cancele la copia por problemas momentáneos de conexión; en caso de persistir este problema; es cancelada la operación automáticamente, borrando los segmentos del archivo temporalmente copiados; de igual manera sucederá en caso que el cliente desee cancelar la operación.

2.4 Modelo del Dominio

No se presenta un modelo completo del negocio sino un modelo de dominio, el cual se considera en RUP como un subconjunto del llamado modelo de objetos del negocio. El Modelo de Dominio[11] es una representación visual estática del entorno real objeto del proyecto. Es decir, un diagrama con los objetos que existen (reales) relacionados con el módulo de integración que se va a realizar y las relaciones que hay entre ellos. (11)

Este modelo permite mostrar de manera visual los principales conceptos que se manejan en el dominio de los mecanismos y de esta forma utilizar un vocabulario común que ayude a usuarios, desarrolladores e interesados a entender el contexto en que se ubica el sistema, logrando una captura correcta de requisitos.

Conceptos:

- **Integration Module**: Objeto que representa el módulo de integración.
- **M.Content:** Módulo encargado de gestionar los contenidos en la plataforma.
- **User**: Objeto encargado de la gestión de contenido.
- **Content**: Objeto que representa los contenidos que se gestionan.
- **Category**: Objeto que representa las categorías de los contenidos.
- **Sub category**: Objeto que representa las subcategorías de los contenidos.

[11] El Modelo de Dominio ayuda a comprender los conceptos que utilizan los usuarios, los conceptos con los que trabajan y con los que deberá trabajar el subsistema.

2.4.1 Diagrama de Clases del Dominio

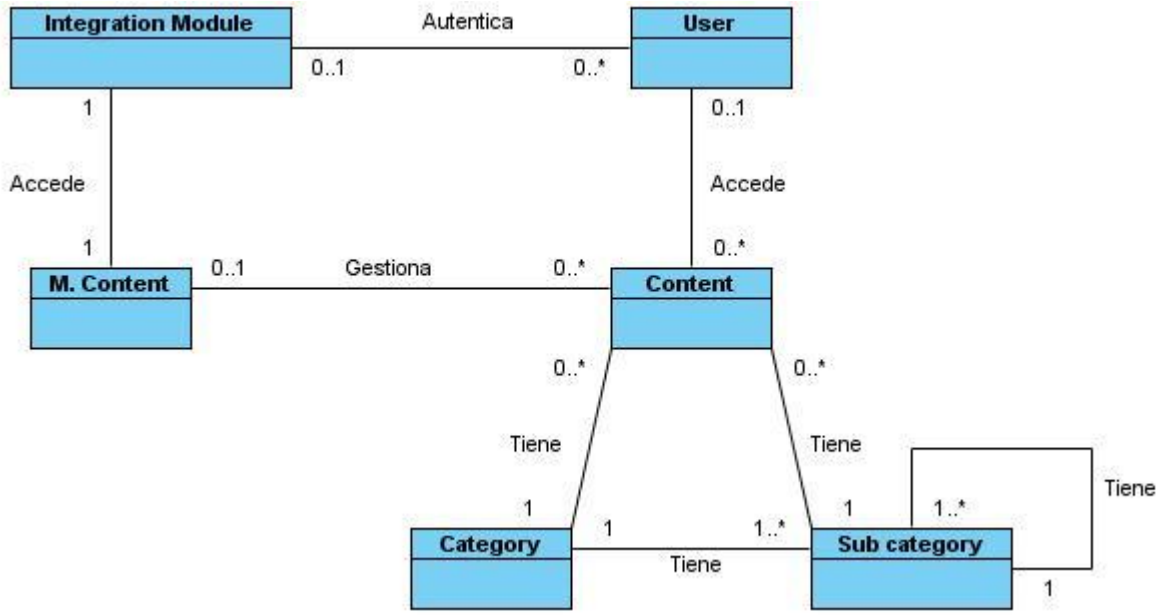

Figura 1: Modelo de Dominio.

2.5 Especificación de requisitos del sistema

2.5.1 Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir (12).A continuación se enuncian los requerimientos funcionales que debe cumplir el subsistema:

RF 1: Autenticar Usuario.

Permitirá al sistema externo autenticarse en el sistema.

- Usuario.
- Contraseña.

RF 2: Modificar Datos.

Permitirá que el sistema externo modifique los datos que desee.

- Campaña.
- Código Zip.
- Dirección.

- Descripción.

RF 3: Modificar Contraseña.

Permitirá que el sistema externo modifique su contraseña.

- Contraseña.

RF 4: Obtener Reporte.

Permitirá al sistema externo seleccionar que el reporte del contenido más descargado.

- Mostrar el contenido más descargado.

RF 5: Buscar Categoría.

Permitirá realizar una búsqueda por categoría, a través del criterio introducido por el sistema externo.

- Nombre.
- Subcategoría.

RF 6: Modificar Categoría.

Permitirá que el sistema externo pueda modificar los datos de la categoría.

- Nombre.
- Descripción.
- Fichero de Imagen de Vista Previa.
- Colección de subcategorías.

RF 7: Eliminar Categoría.

Permitirá al sistema externo eliminar una determinada categoría.

RF 8: Buscar Sub-categoría.

Permitirá realizar una búsqueda por sub-categoría, a través del criterio introducido por el sistema externo.

- Nombre.

RF 9: Modificar Subcategoría.

Permitirá que el sistema externo pueda modificar los datos de la categoría.

- Nombre.
- Descripción.

- Fichero de Imagen de Vista Previa.
- Categoría.
- Colección de subcategoría.
- Colección de Contenido.

RF 10: Eliminar Subcategoría.

Permitirá al sistema externo eliminar una determinada subcategoría.

RF 11: Insertar Contenido.

Permitirá que el sistema externo pueda insertar el contenido que desee.

- Fichero del Contenido.

RF 12: Buscar Contenido.

Permitirá realizar una búsqueda por contenido, a través del criterio introducido por el sistema externo.

- Nombre.
- Subcategoría.
- Categoría.

RF 13: Modificar Contenido.

Permitirá que el sistema externo pueda modificar los datos del contenido.

- Nombre.
- Precio.
- Subcategoría.
- Fichero de Imagen de Vista Previa.
- Descripción.
- Fichero del Contenido.

RF 14: Eliminar Contenido.

Permitirá al sistema externo eliminar un determinado contenido.

RF 15: Particionar Fichero del Contenido.

Permitirá al módulo de integración particionar el Fichero del Contenido que desea insertar el sistema externo. Se particionará el fichero del contenido en segmentos de arreglos de bytes, evitando sobrecargar la memoria.

RF 16: Guardar Fichero del Contenido.
Permitirá al módulo de integración guardar el Fichero del Contenido que el sistema externo desea insertar en la plataforma. Se irá guardando cada uno de los segmentos recibidos del fichero del contenido particionado (para que no exista la posibilidad de pérdida de algún dato del contenido).

2.5.2 Requisitos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el sistema debe cumplir. A continuación se enuncian los requerimientos no funcionales por categoría que debe poseer la aplicación:

Soporte

- ✓ Brindar soporte a los usuarios que necesiten capacitación para operar con el sistema; puede ser entrenamiento del personal e integración directa con el software.

Portabilidad

- ✓ El subsistema de integración debe ser desarrollado en forma modular.
- ✓ El módulo de integración debe ser multiplataforma.

Restricciones en el diseño y la implementación

- ✓ Utilizar para la implementación el lenguaje de programación: JAVA.
- ✓ Utilizar como herramienta CASE: Visual Paradigm con soporte para UML.

Usabilidad

- ✓ El subsistema debe permitir a los sistemas externos poder acceder a la información de una manera segura.
- ✓ Podrá ser utilizado por cualquier usuario que posea conocimientos básicos de programación web.

Seguridad

- ✓ Registrar una cantidad finita de eventos anormales del sistema en LOGs, para poder desarrollar una auditoría.

- ✓ Garantizar el tratamiento adecuado de las excepciones, lanzando un tipo de excepción en dependencia de la situación anómala que ocurra; pudiendo manejarla en el cliente conociendo el error ocurrido.
- ✓ Proporcionar a los mecanismos de comunicación; seguridad a nivel de método y a nivel de peticiones web, utilizando como política de seguridad la autenticación.
- ✓ Se le proporcionará una seguridad basada en la autenticación.

Software

- ✓ Utilizar la Máquina Virtual de Java para el adecuado funcionamiento del sistema.

Hardware

- ✓ El sistema se desplegará en un servidor con 4gb de memoria RAM como requerimiento mínimo, procesador Core 2 Duo a 2.6 ghz o superior y debe contar además con un almacenamiento en disco de 500gb.

- ✓ Las aplicaciones clientes deberán contar como requerimiento mínimo con 512mb de RAM, procesador Pentium(R) 4 a 3.00.ghz.

Apariencia o Interfaz interna

- ✓ La implementación de las funcionalidades deben seguir los lineamientos de código establecidos por la Dirección del Proyecto, para garantizar la comprensión del código por desarrolladores en futuras versiones.

2.6 Modelo de Caso de Uso del Sistema

A continuación se enuncia el Modelo de Caso de Uso del Sistema de forma narrativa con la finalidad de describir, bajo la forma de acción y reacción el comportamiento del sistema desde la óptica del usuario.

2.6.1 Definición de Actores del Sistema Automatizar

Actor	Descripción
Sistema Externo	Representa al sistema externo que es el encargado de realizar todas las acciones en el sistema.

Tabla 1: Descripción de los Actores del Sistema.

2.6.2 Diagrama de Casos de Uso del Sistema a Automatizar

Los Diagramas de Casos de Uso se utilizan para modelar la vista de los Casos de Usos del sistema. Esto lleva a modelar el contexto del sistema, subsistema o clase, y el modelado de los requisitos para mostrar cómo responde el sistema a situaciones que se produzcan. Una vez confeccionado el modelo de dominio y definidos los requisitos funcionales, se obtuvo el siguiente Diagrama de Casos de Uso del Sistema, perteneciente al Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

Figura 2: Diagrama de Casos de Usos del Sistema.

2.7 Descripción de Casos de Uso

Una descripción detallada de casos de uso ayuda a entender cómo debe funcionar el sistema. A continuación se muestran las descripciones detalladas de los casos de uso arquitectónicamente significativos: Insertar Contenido, Particionar Fichero del Contenido y Guardar Fichero del Contenido; perteneciente al Diagrama de Casos de Usos del Sistema.

Caso de Uso:	Insertar Contenido.
Actores:	Sistema Externo.
Resumen:	Permite proveer contenidos por parte de los sistemas externos a la Plataforma de Gestión de Contenidos para Dispositivos Móviles.
Precondiciones:	El sistema externo debe estar autenticado en el sistema.
Referencias:	RF 11.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El sistema externo inserta los datos del contenido a insertar.	**2.** Ver Caso de Uso **Particionar Fichero del Contenido.** 3. Ver Caso de Uso **Guardar Fichero del Contenido.**
	4. El sistema muestra un mensaje" Él archivo se ha copiado correctamente".
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**

	4.1 El Sistema muestra un mensaje de error si la operación no pudo ser realizada.
Poscondiciones	Queda insertado el contenido en la base de datos.

Tabla 2: Descripción detallada del Caso de Uso Insertar Contenido.

Caso de Uso:	Particionar Fichero del Contenido.
Actores:	Sistema Externo.
Resumen:	Permite al módulo de integración particionar el Fichero del Contenido que se desea insertar.
Precondiciones:	El sistema externo debe estar autenticado en el sistema.
Referencias:	RF 15.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema particiona el fichero del contenido a insertar en arreglos de bytes.
	2. El sistema envía los segmentos de arreglos. Los cuales se recepcionan en la aplicación servidor.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema muestra un mensaje de error si la operación no pudo ser realizada.

Poscondiciones	Queda particionado el fichero del contenido en arreglos de byte.

Tabla 3: Descripción detallada del Caso de Uso Particionar Fichero del Contenido.

Caso de Uso:	Guardar Fichero del Contenido.
Actores:	Sistema Externo.
Resumen:	Permite al módulo de integración guardar el Fichero del Contenido que se desea insertar.
Precondiciones:	El sistema externo debe estar autenticado en el sistema.
Referencias:	RF 16.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema comprueba que todas las partes del arreglo de byte, hayan llegado sin problema a la plataforma.
	2. El sistema restaura el fichero de contenido en el directorio especificado.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El sistema muestra un mensaje de error notificando que parte del arreglo no se copió correctamente.
Poscondiciones	Queda restaurado el contenido y almacenado en el directorio especificado.

Tabla 4: Descripción detallada del Caso de Uso Guardar Fichero del Contenido.

En el ***Anexo1*** se describirán detalladamente los restantes casos de uso del subsistema definidos en el diagrama.

Conclusiones

Con la culminación de este capítulo se pretende alcanzar una mejor comprensión de lo que debe de hacer el sistema para dar solución al problema planteado. Con el listado de los requisitos funcionales y no funcionales se define que ha de hacer y que características ha de tener el producto informático para solventar la problemática y sea usable y confiable. El Modelo de Domino presentado permitió comprender mejor y gestionar la complejidad del mismo. Con el Modelo de Casos de Uso del Sistema se tiene una idea, bajo la forma de acciones y reacciones, del comportamiento del sistema desde el punto de vista del usuario.

CAPÍTULO 3: DISEÑO DEL SISTEMA

Introducción

El objetivo fundamental a tratar en el presente capítulo es la realización de un diseño consecuente que responda a los requerimientos planteados en el capítulo anterior. El diseño es un refinamiento que toma en cuenta los requerimientos no funcionales, por lo cual se centra en cómo el sistema cumple sus objetivos. Se presentan los diagramas de clases del diseño, así como los diagramas de secuencia y los modelos lógico y físico de datos. También se describen los patrones de diseños empleados.

3.1 Modelo de Diseño

El modelo de diseño es una abstracción de la implementación del sistema. Se utiliza para concebir y para documentar el diseño del sistema de software. Es un producto de trabajo integral y compuesto que abarca todas las clases de diseño, subsistema, paquetes, colaboraciones y las relaciones entre ellos. (13)

3.2.1 Diagrama de Interacción

El diagrama de interacción, representa la forma de cómo un Cliente (Actor) u Objeto (Clases) se comunican entre sí en petición a un evento. Esto implica recorrer toda la secuencia de llamadas, de donde se obtienen las responsabilidades claramente. (14)

Se elaboraron diagramas de secuencias para una mejor compresión de los eventos que tienen lugar por cada escenario de los diferentes casos de usos que se identificaron en el capítulo anterior. Los diagramas de interacción por cada caso de uso se encuentran en el ***Anexo II*** Diagramas de Secuencias por Casos de Uso.

3.2.2 Diagramas de Clases del Diseño

Una clase del diseño representa una abstracción de una o varias clases en la implementación del sistema, exactamente, a qué corresponde depende del lenguaje de implementación. Es una construcción similar en la implementación del sistema, es decir:

- El lenguaje utilizado para especificar una clase del diseño es lo mismo que el lenguaje de programación.

- Los métodos de una clase del diseño tienen correspondencia directa con el correspondiente método en la implementación de las clases.
- Una clase del diseño puede posponer el manejo de algunos requisitos para las siguientes actividades de implementación, indicándolos como requisitos de implementación de la clase.
- Una clase del diseño puede proporcionar interfaces si tiene sentido hacerlo en el lenguaje de programación.(15)

El subsistema propuesto (remote), se integra al Módulo Contenido (content); el cual posee varias clases de diseño e interfaces las cuales están fuertemente asociadas según funciones. Se realizaron diagramas de clases del diseño y un diagrama de paquetes; así como su organización teniendo en cuenta el patrón arquitectónico basado en capas.

Diagrama de Paquetes

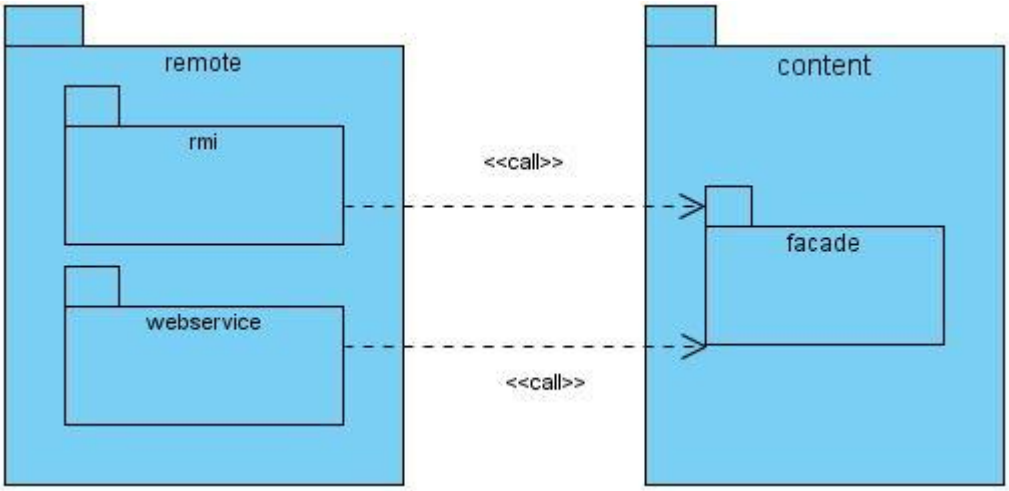

Figura 3: Diagrama de paquetes.

Diagramas de Clases

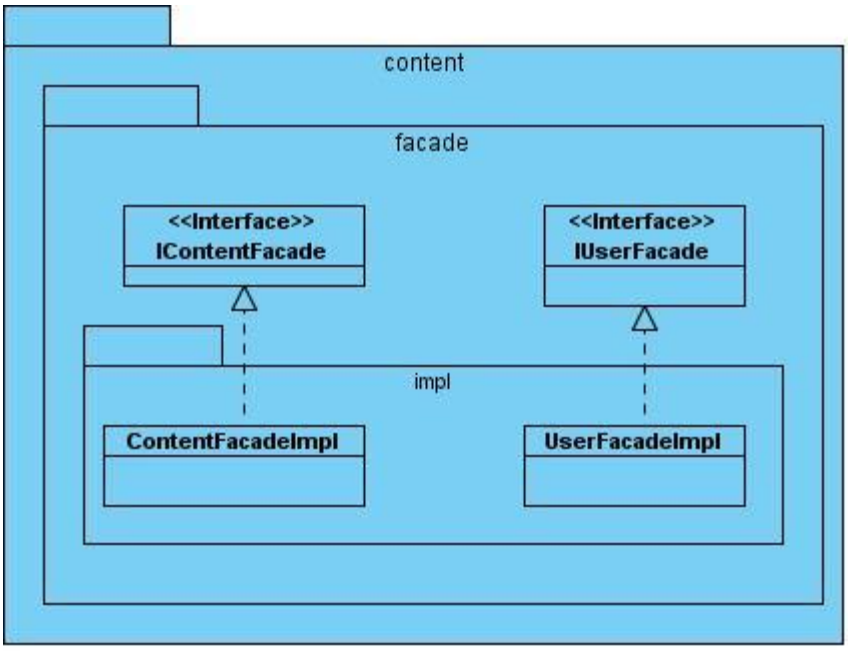

Figura 4: Diagrama de clases del diseño correspondiente al paquete content.

A continuación se representan los diagramas de clases del diseño de los paquetes rmi y webservice, con sus funcionalidades más importante referente a Insertar Contenido.

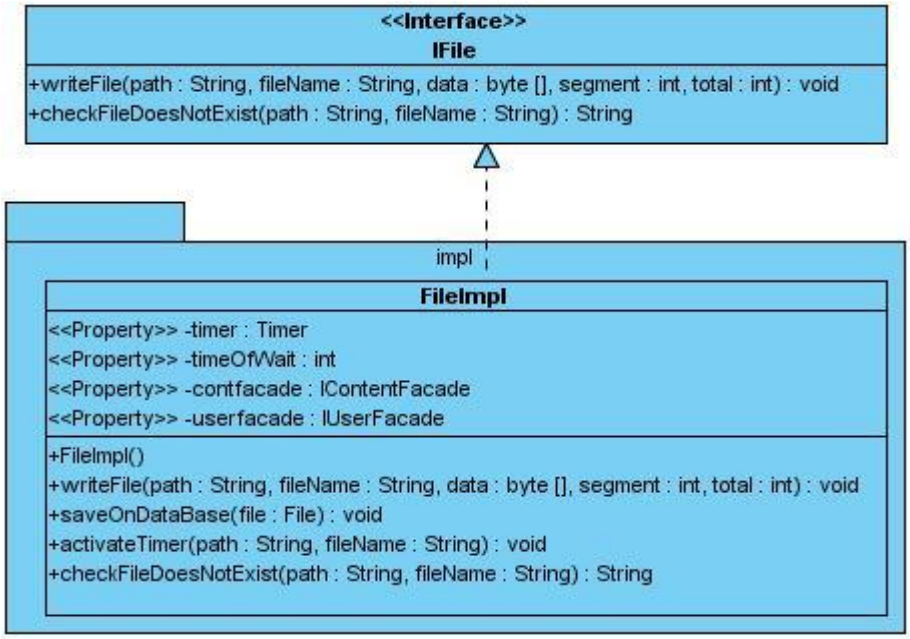

Figura 5: Diagrama de clases del diseño correspondiente al paquete rmi.

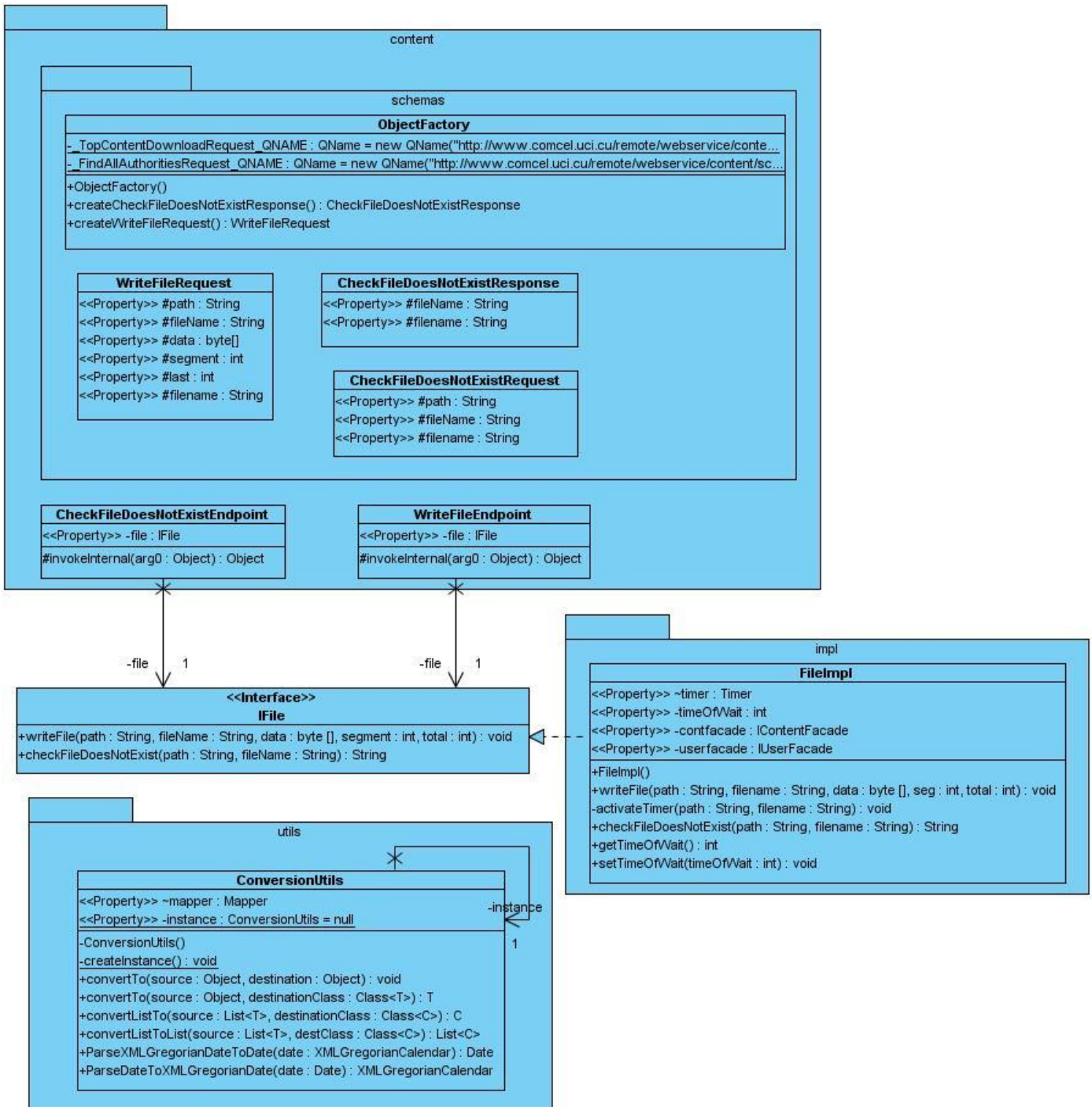

Figura 6: Diagrama de clases del diseño correspondiente al paquete webservice.

3.3 Descripción de la Arquitectura

3.3.1 Patrones Arquitectónicos

La Arquitectura en Capas define cómo organizar el modelo de diseño a través de capas, que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa solo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además, nos ayuda a identificar qué puede reutilizarse. Cuando se trabaja en capas, se pueden usar diferentes niveles o cantidad de capas (2 o más capas), en este caso se utilizó para el desarrollo del subsistema una Arquitectura en 2 Capas; donde la aplicación se divide en dos capas lógicas distintas cada una de ellas con un grupo de interfaces perfectamente definidas. La primera capa se denomina capa de servicios y la segunda capa de negocio. En la capa de servicio se encuentra todo lo relacionado con los servicios RMI y Web Service, y en la capa de negocio todo lo referente al acceso a las funcionalidades de las interfaces del negocio del Módulo de Contenido de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

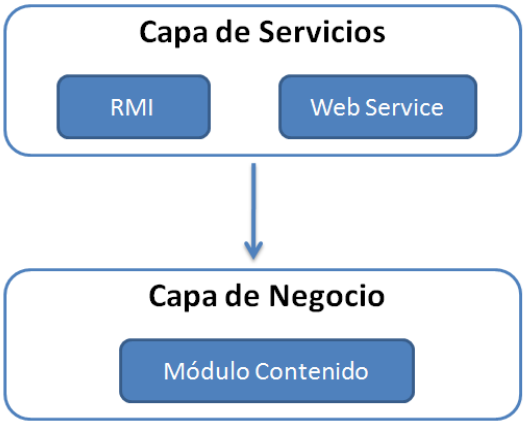

Figura 7: Arquitectura en 2 Capas.

3.3.2 Patrones de Diseño. Patrones GRASP

Los Patrones de Diseño representan una amplia colección de los principios generales basados en la experiencia que guía la creación de un software, uno de lo más conocidos es GRASP. Es un acrónimo que significa Patrones Generales de Software para Asignar Responsabilidades. El nombre se eligió para

indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos. (16)

Patrón Experto: Consiste en asignar una responsabilidad a la clase que cuenta con toda la información para cumplirla. Permite conservar el encapsulamiento y soporta un bajo acoplamiento logrando sistemas más robustos y de fácil mantenimiento. Brinda además un soporte a la alta cohesión definiendo clases más sencillas y más cohesivas.

```
package cu.uci.comcel.core.remote.webservice.impl;
import java.rmi.RemoteException;

public class MainThread  extends Thread{

    private Cola<ThreadExecution> cola;
    private ThreadExecution threadExecution;
    private IFile file;
    private boolean notrunning;
    private String Url;

    public MainThread(String Url)
    {
        cola= new Cola<ThreadExecution>();
        notrunning= true;
        threadExecution= null;
        this.Url= Url;
        ConnectionSettingsWS setting= new ConnectionSettingsWS();
        setting.setUrl(this.Url);
        file = FileImpl.Initialize(setting);
        start();
    }
```

Figura 8: Clase MainThread.

En la Figura 8 se ilustra uno ejemplo en el cual se evidencia el empleo del patrón experto en el subsistema, la clase MainThread (hilo principal), es la encarga del manejo de todos los hilos que se establecen para proveer los contenidos o consumir los servicios de la plataforma.

Patrón Creador: Consiste en asignar la responsabilidad de la creación de la instancia de una clase a aquella que agrega o contiene objetos de la misma, registra instancias de ella, utiliza específicamente sus objetos o tiene los datos de inicialización que le será transmitidos cuando sea creada su instancia. Brinda soporte a un bajo acoplamiento suponiendo menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización.

```
public void AddContent(String path, String name, String dest, String login, String password)
{
    try {

        ThreadExecution thread= new ThreadExecution(path, name, dest);
             cola.add(thread);

    } catch (Exception e) {
        System.out.println(e.getMessage());
    }

}
```

Figura 9: Método Adicionar de la Clase MainThread.

La clase MainThread es la encargada del manejo de los hilos que se establecen para proveer los contenidos a la plataforma, en la Figura 9 se ilustra el método Adicionar Contenido (“AddContent”), en el cual se crea una instancia del la clase ThreadExecution; por el cual se va a proveer un contenido, evidenciándose el uso del patrón creador en el subsistema.

Patrón Alta Cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas. Es un principio que debe estar presente durante el diseño, es un patrón evaluativo que se aplica al juzgar decisiones de diseño. Anteriormente se representó el uso del patrón experto, este patrón brinda un soporte a la alta cohesión permitiendo definir clases más sencillas y más cohesivas, por lo antes expuesto se pone de manifiesto el uso del patrón alta cohesión en el subsistema.

Patrón Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otra. Clases con fuerte acoplamiento son más difíciles de entender cuando están aisladas, son más difíciles de reutilizar porque se requiere la presencia de otras clases de las que dependen. El Bajo Acoplamiento y la Alta Cohesión constituyen un principio, un patrón evaluativo.

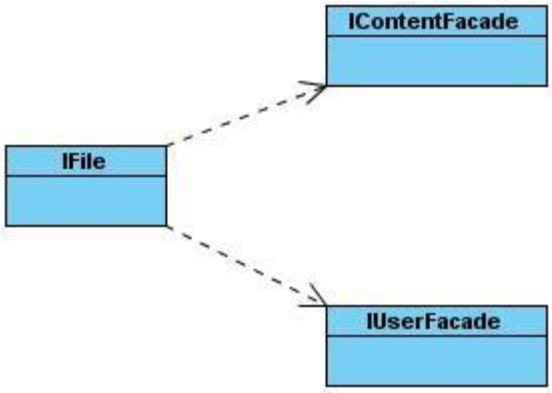

Figura 10: Relaciones entre clases.

La Figura 10 muestra la cantidad de conexiones máximas que existirán entre las clases de los subsistemas: módulo de integración y módulo de contenido, evidenciándose un bajo acoplamiento, en la cual se muestra la relación de la interfaz principal del módulo de integración con las interfaces de la fachada del módulo contenido.

Controlador: Es un objeto de interfaz no destinada al usuario, que se encarga de manejar un evento del sistema. Define además el método de su operación.

```
public class FileImpl  implements IFile {


    private Timer timer;
    private int timeOfWait;
    private IContentFacade contfacade;
    private IUserFacade userfacade;

    public Timer getTimer() {
        return timer;
    }
    public IContentFacade getContfacade() {
        return contfacade;
    }
    public void setContfacade(IContentFacade contFacade) {
        this.contfacade = contFacade;
    }
    public IUserFacade getUserfacade() {
        return userfacade;
    }
    public void setUserfacade(IUserFacade userFacade) {
        this.userfacade = userFacade;
    }
    public void setTimer(Timer timer) {
        this.timer = timer;
    }
```

Figura 11: Clase FileImpl.

En la Figura 11 se ilustra un ejemplo del empleo del patrón controlador en el subsistema. La clase FileImpl, implementa las funcionalidades de la interfaz principal IFile; las cuales hacen referencia a los servicios que brinda la plataforma a los sistemas externos. En esta clase se muestra el uso del patrón controlador pues se instancian objetos de las interfaces del negocios del módulo de contenido.

3.3.3 Patrones de Diseño. Patrones GOF

Facade: La idea principal es la de ocultar todo lo posible la complejidad de un sistema, el conjunto de clases o componentes que lo forman, de forma que solo se ofrezca un (o unos pocos) punto de entrada al sistema tapado por la fachada. (17). El patrón Facade se utiliza en el subsistema pues la interfaz del módulo de integración se relaciona con las interfaces del paquete Facade (Fachada), del módulo de contenido, lo anteriormente descrito se ilustra en la Figura 10.

3.4 Modelo de datos

El Modelo de Datos permite definir una estructura que brinda la posibilidad de almacenar datos y recuperar información. Describe las representaciones lógicas y físico de datos persistentes utilizados por el Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

3.4.1 Modelo Lógico de Datos

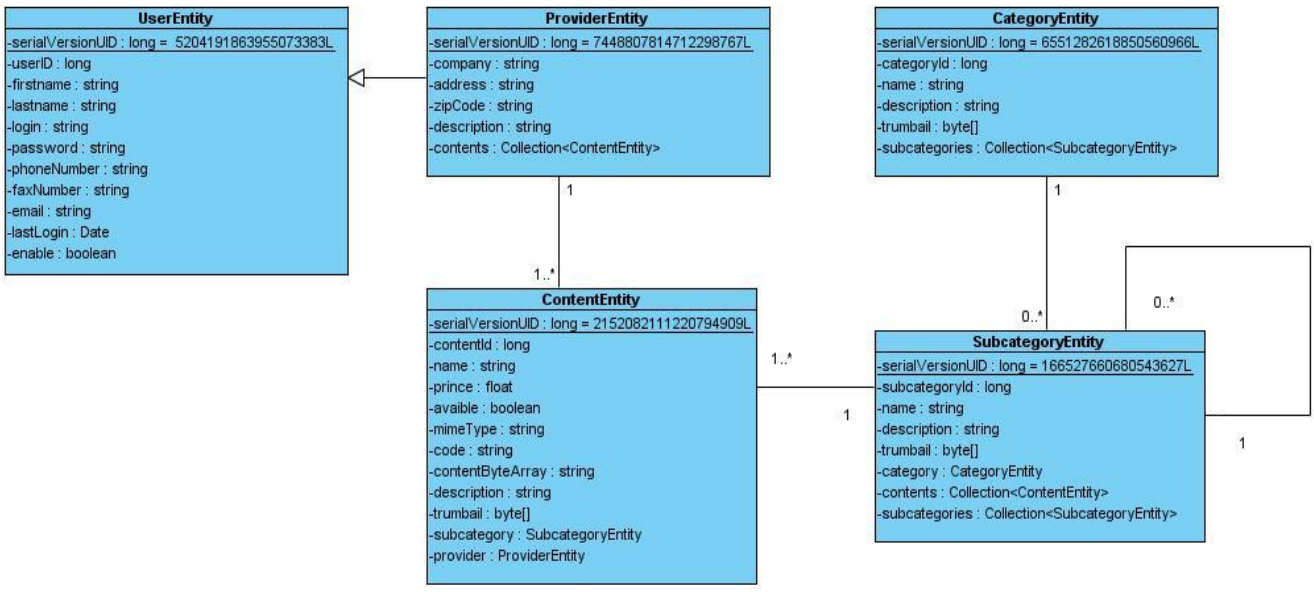

Figura 12: Diagrama de Clases Persistentes.

3.4.2 Modelo Físico de Datos

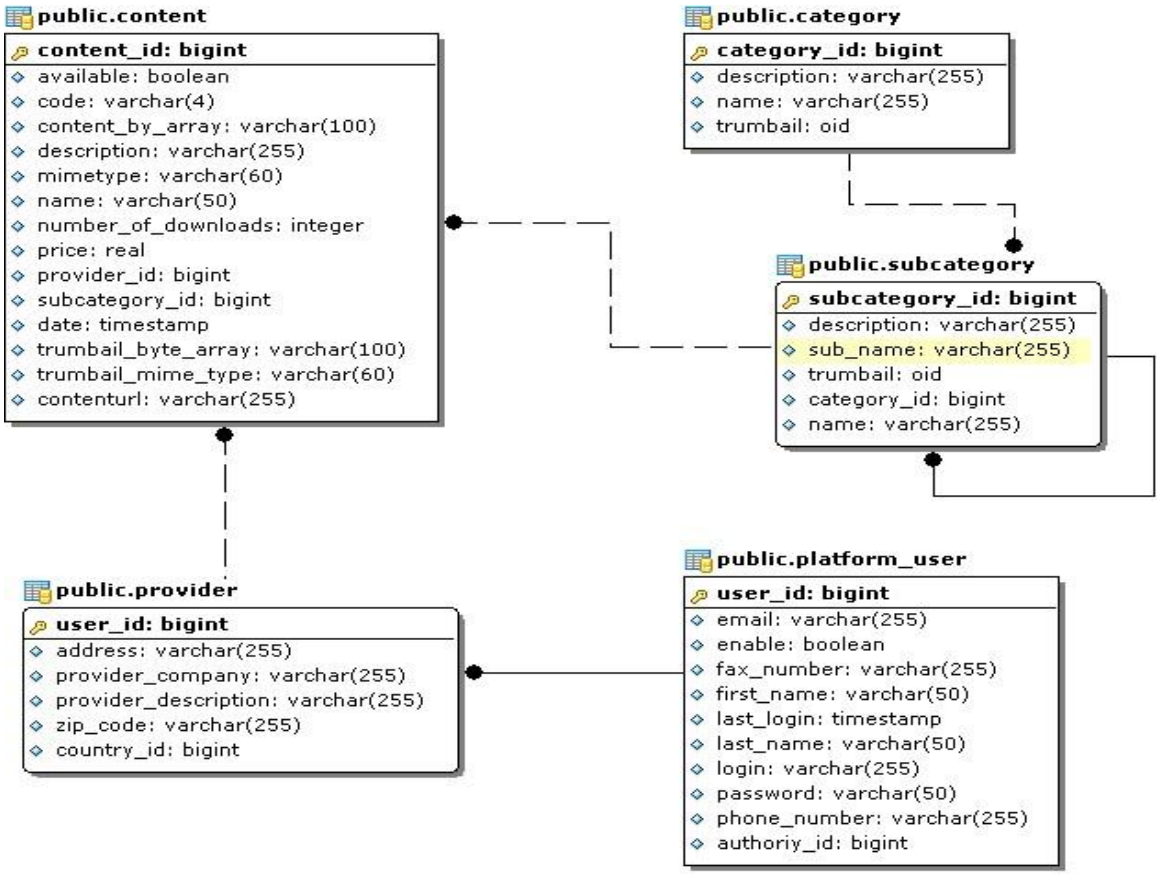

Figura 13: Modelo Físico de Datos.

3.5 Tratamiento de Errores

El Tratamiento de errores estará enfocado en detectar los posibles errores que pudieran surgir en el sistema que puedan llegar a provocar un mal funcionamiento y dañar el resultado esperado de una determinada operación, para esto se utilizarán librerías de clases que extiendan de la clase *Excepcion* y que permitan identificar el problema en todos los niveles de la aplicación, además estos errores se guardarán en LOGs para posibilitar una revisión detallada del comportamiento del sistema antes determinadas circunstancias. El principal objetivo del tratamiento de errores es identificar los principales errores que pudieran surgir durante el funcionamiento del módulo de integración y proporcionar su adecuado tratamiento.

3.6 Seguridad

Una de las principales características con que deben contar los productos informáticos es poder brindarle seguridad y confianza al cliente, es por esta razón que la "Plataforma de Gestión de Contenidos para Dispositivos Móviles" no está ajena a ello y utiliza para la parte de seguridad Spring Security que posibilita brindar autenticación y autorización al usuario cuando va a insertar un contenido o va a realizar una determinada operación en el sistema.

Conclusiones

En este capítulo se mostraron los diagramas de clases del diseño, con los cuales se pudo alcanzar un punto de vista del sistema en cuanto al entendimiento de las especificaciones de los requisitos reflejados en el capítulo anterior. Se expusieron los diagramas de secuencias. Se mostraron además los patrones arquitectónicos y de diseño; además los modelos de datos utilizados.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

Introducción

En este capítulo se presentan en términos de componentes los elementos anteriormente identificados en el Diseño. Se modela un diagrama de componentes, con los componentes principales que constituyen la plataforma, sirviendo de guía para luego presentar la situación física de los componentes lógicos desarrollados en el diagrama de despliegue.

4.1 Modelo de Implementación

El modelo de implementación representa la composición física de la implementación en términos de subsistemas de implementación, y elementos de implementación (directivos y archivos, incluyendo código fuente, datos y archivos ejecutables). (18)

4.1.1 Diagrama de Despliegue

Un diagrama de Despliegue muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue. (19)

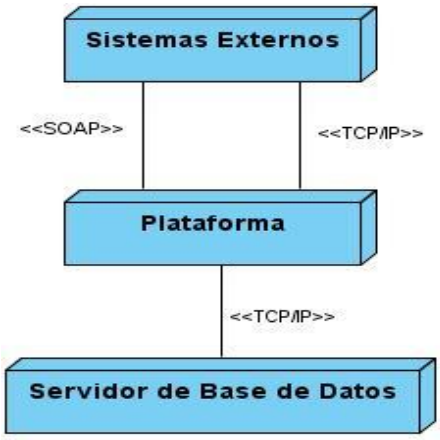

Figura 14: Diagrama de Despliegue.

4.1.2 Diagrama de Componentes

Los diagramas de componentes modelan la vista estática de un sistema. Se representa como un grafo de componentes software unidos por medio de relaciones de dependencia (compilación, ejecución), pudiendo mostrarse las interfaces que estos soporten. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. (20)

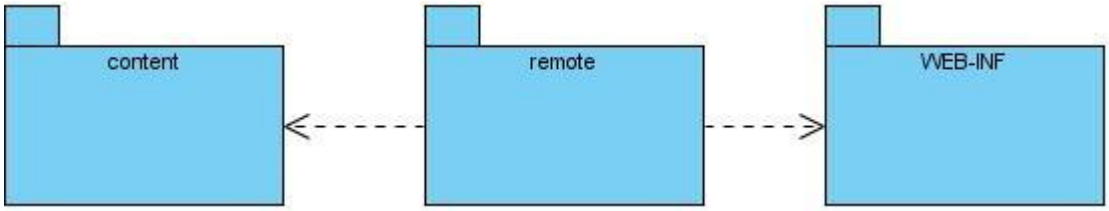

Figura 15: Diagrama de componentes.

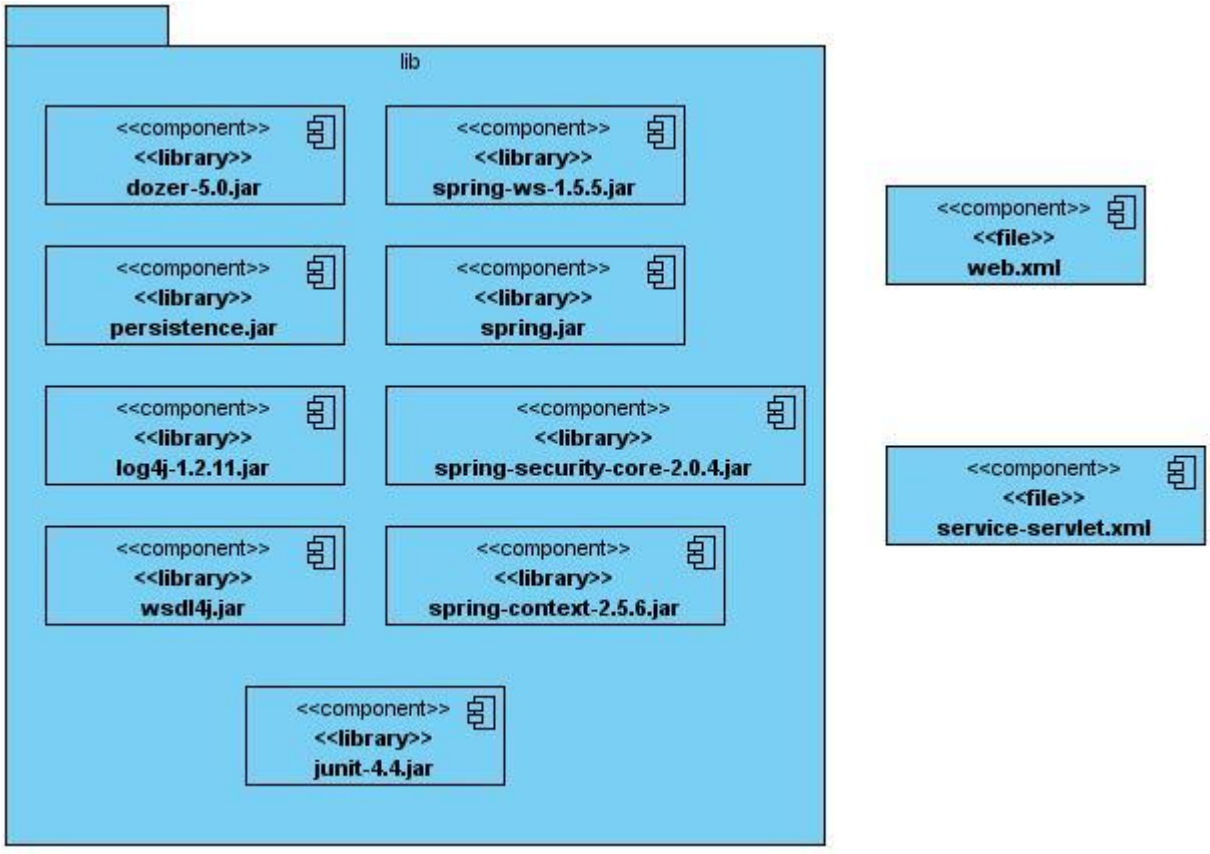

Figura 16: Diagrama de Componentes del paquete WEB-INF.

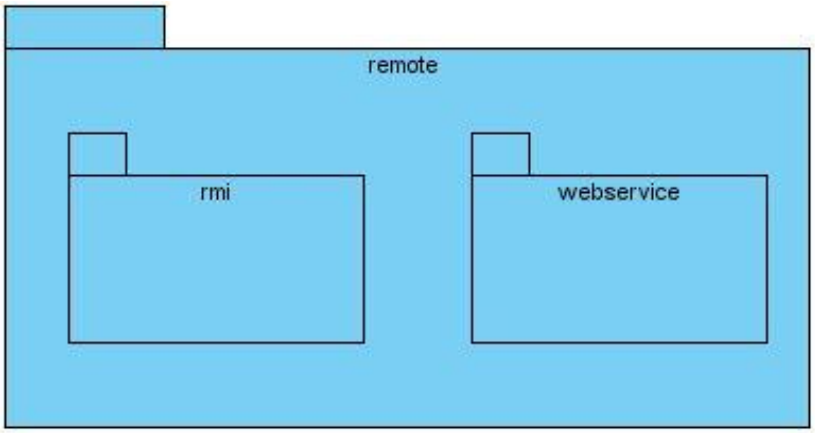

Figura 17: Diagrama de Componentes del paquete remote.

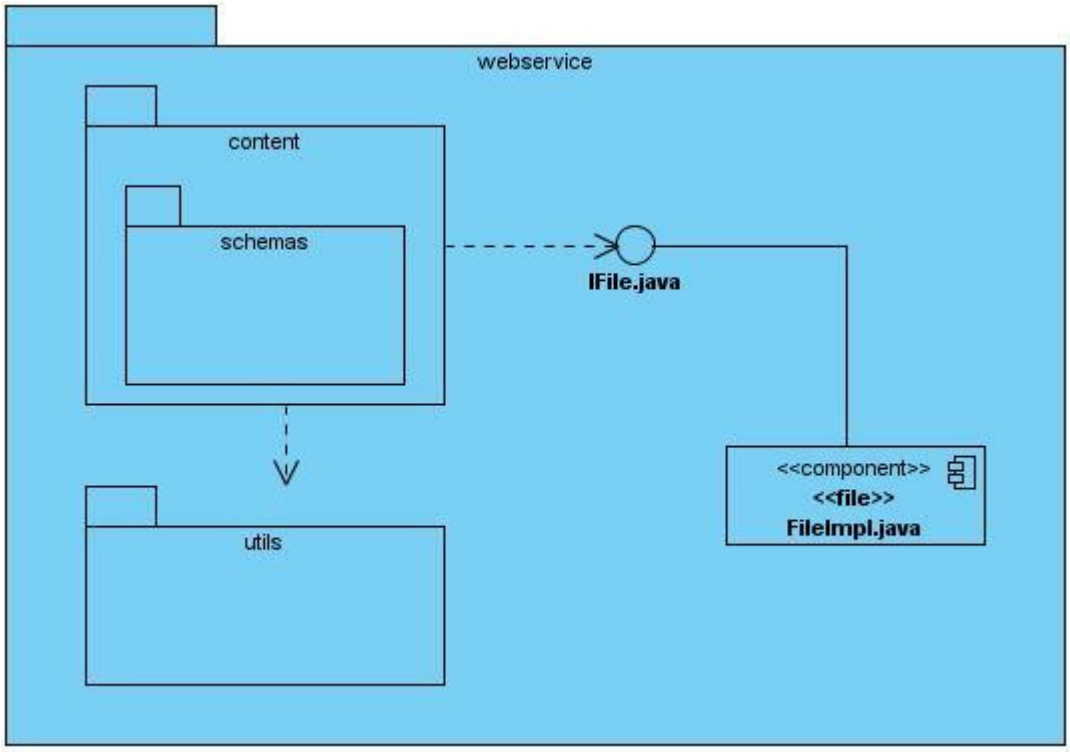

Figura 18: Diagrama de componentes del paquete webservice.

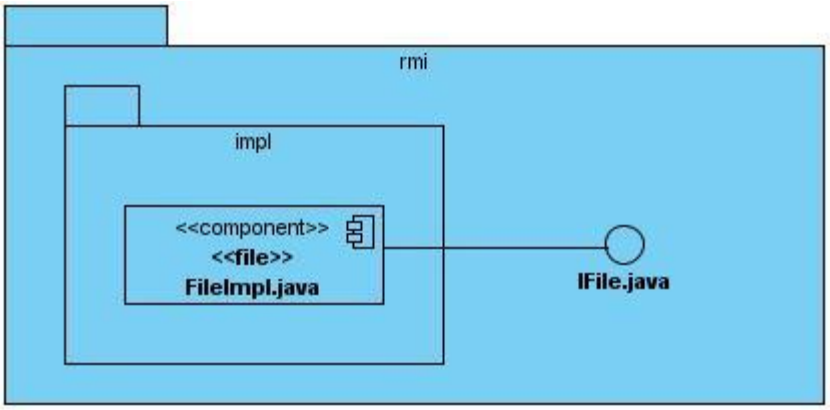

Figura 19: Diagrama de componentes del paquete rmi.

A continuación se representan los componentes más importantes presentes en el paquete webservice, modelado anteriormente.

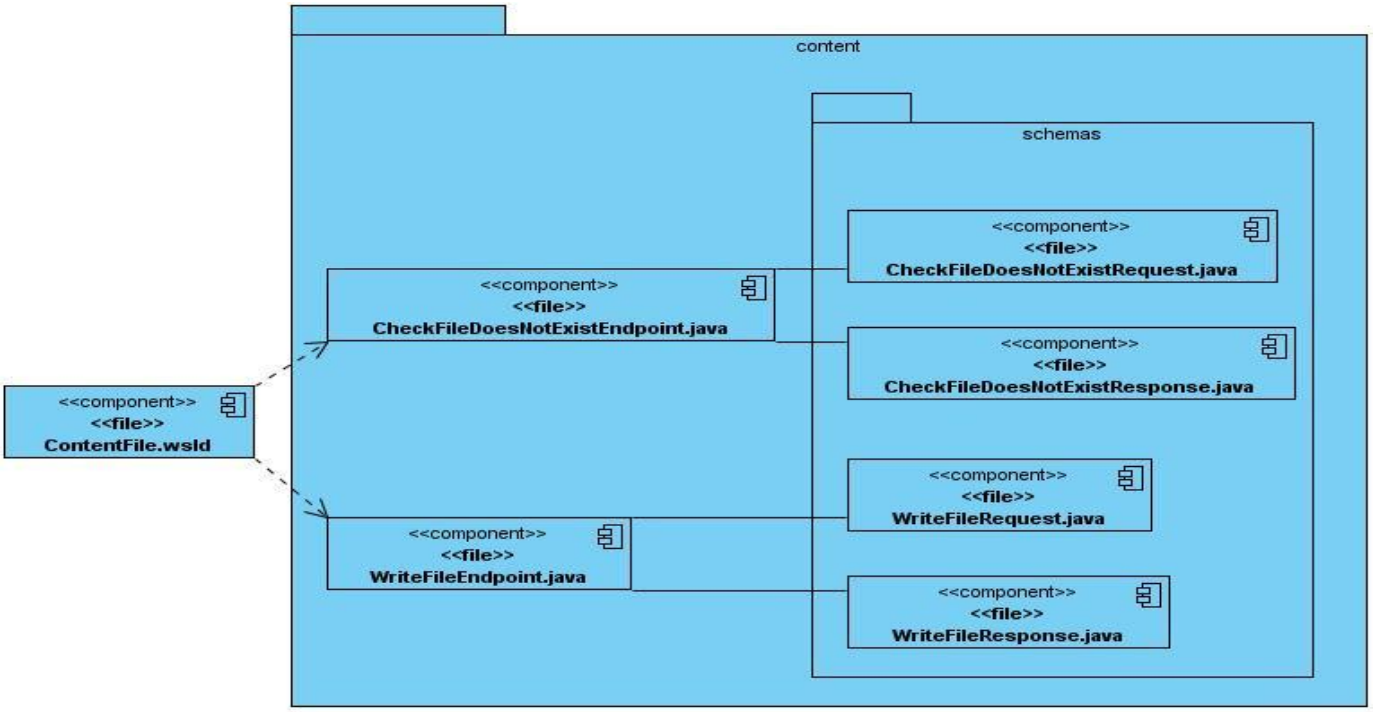

Figura 20: Diagrama de componentes del paquete webservice.

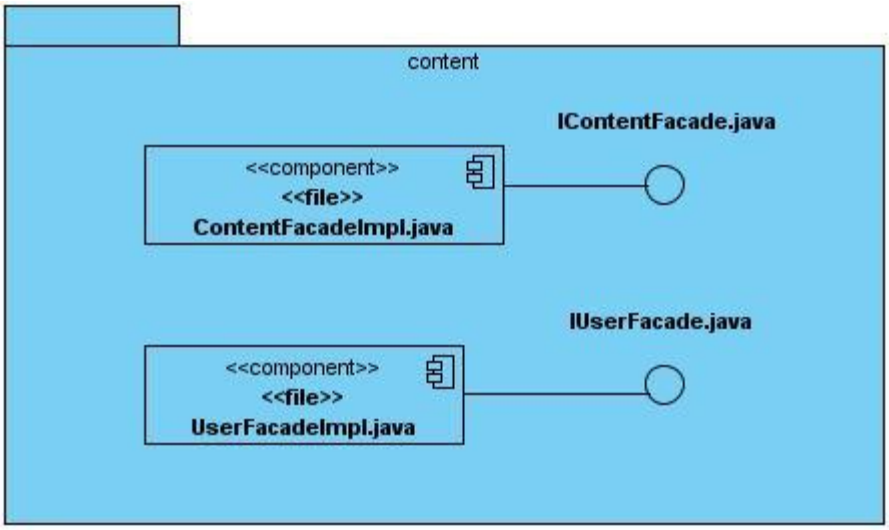

Figura 21: Diagrama de Componentes del paquete content.

4.2 Pruebas

Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto software. Se utilizan para identificar posibles fallos de implementación, calidad, o usabilidad de un programa. Éstas se integran dentro de las diferentes fases del ciclo de desarrollo del software en la Ingeniería de software. Para determinar el nivel de calidad se debe efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto a los requerimientos del sistema. El módulo de integración cuenta con 16 casos de usos del sistema a los cuales se le realizaron diferentes tipos de pruebas. Las pruebas unitarias y de integración se realizaron con la ayuda de JUnit; y para la prueba de funcionalidad se utilizo un modelo de prueba estático.

4.2.1 Tipos de Pruebas

A continuación se describen los tipos de prueba realizados al módulo de integración.

4.2.1.1 Pruebas Unitarias

Las pruebas unitarias es una forma de comprobar el correcto funcionamiento de un segmento (método) de código. Esto sirve para asegurar que cada uno de los segmentos funcione correctamente por separado. Al módulo de integración se le realizaron a cada una de sus funcionalidades pruebas unitarias, donde se comprobó el correcto funcionamiento; verificando que el sistema mostrara respuestas eficientes a peticiones realizadas por los sistemas externos y conociendo además el tiempo en que demoró la

aplicación en arrojar los resultados o en realizar una operación. En la Figura 17 se muestra el marco de trabajo de JUnit donde se realizaron pruebas unitarias a las funcionalidades del sistema. Se obtuvo como resultado 0 error en las pruebas realizadas.

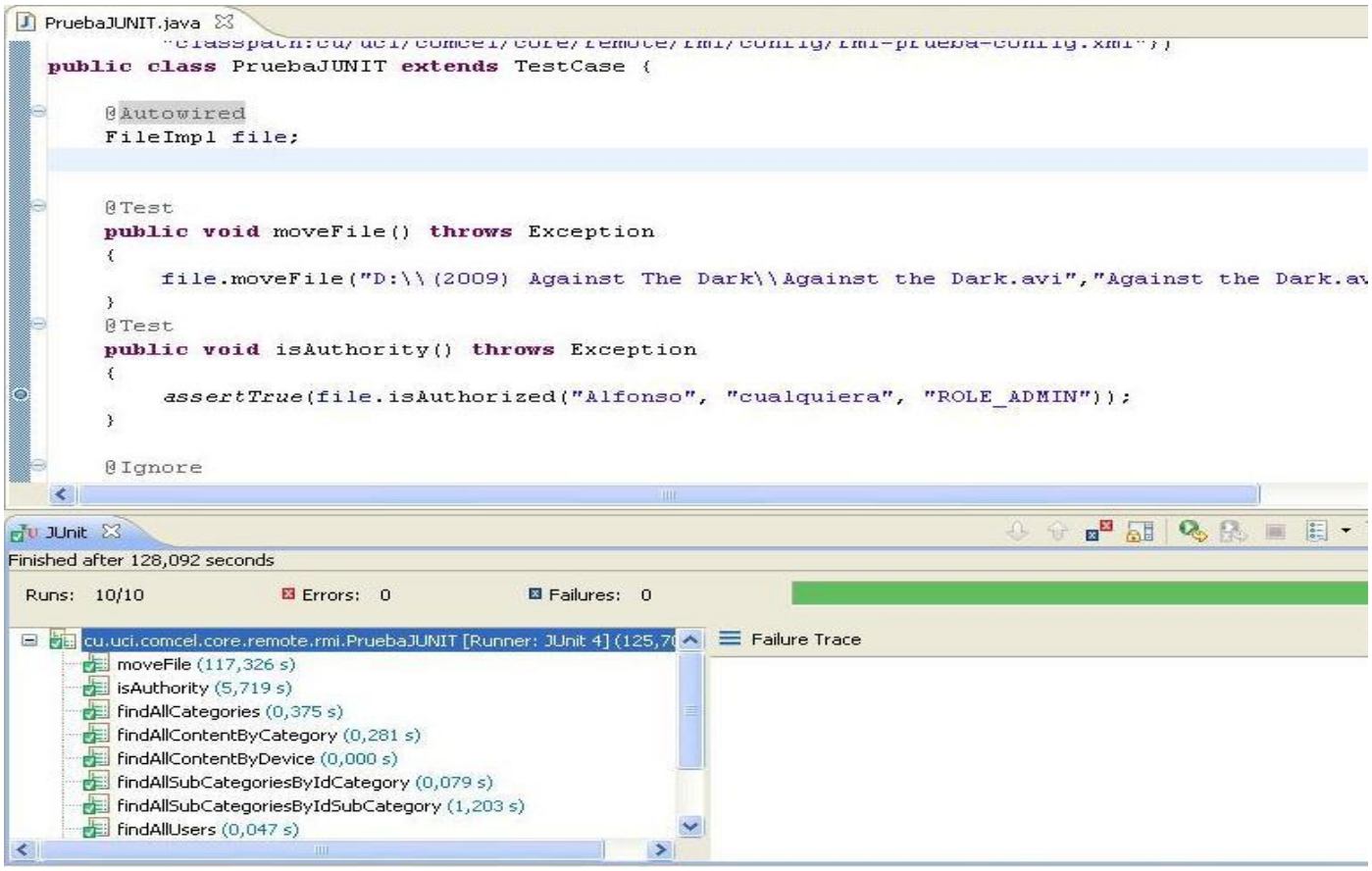

Figura 22: Marco de Trabajo de JUnit.

Las Pruebas Unitarias realizadas a las principales funcionalidades del módulo de integración se encuentran en el ***Anexo III.*** Luego, con las Pruebas de Integración, se podrá asegurar el correcto funcionamiento del módulo de integración.

4.2.1.2 Pruebas de Integración

Las pruebas de integración son las que se realizan en el ámbito del desarrollo de software, una vez comprobadas, el correcto funcionamiento de las pruebas unitarias. Esto se refiere a la prueba o pruebas

de todos los elementos unitarios que componen un proceso, hecho en conjunto, de una sola vez. Las Pruebas de Integración realizadas al módulo de integración se encuentran en el ***Anexo IV.***

4.2.1.3 Pruebas de Funcionalidad

Las pruebas de funcionalidad se basan en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. Se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con que cuenta el módulo de integración. Las mismas se realizaron mediante un modelo de prueba estático en el cual se creó o instancio un objeto de la clase principal "*MainThread*"; el cual permitió acceder a las funcionalidades del módulo de integración y comprobar los resultados, verificando las respuestas arrojadas en la consola. Estas pruebas permitieron verificar en cada iteración del desarrollo del módulo de integración, el correcto funcionamiento de las funcionalidades, ayudando a identificar 4 no conformidades a las cuales posteriormente se le dieron solución. Las clases pruebas realizadas al módulo de integración se encuentran en el ***Anexo V.***

Conclusiones

En este capítulo se describió la distribución del sistema en nodos mediante el diagrama de despliegue; se mostraron las organizaciones y las dependencias lógicas entre componentes a través del diagrama de componentes. Además de realizaron diferentes tipos de pruebas para el correcto funcionamiento del módulo de integración.

CAPÍTULO 5: ESTUDIO DE FACTIBILIDAD

Introducción

En este capítulo se muestra la factibilidad de la realización del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles, usando el método Puntos de Casos de Uso, el cálculo del esfuerzo y el tiempo de desarrollo, así como los beneficios y costos, mostrando de esta forma si es viable o no llevar a cabo la realización del módulo de integración de la plataforma.

5.1 Métodos de Estimación "Puntos por Casos de Uso"

El método de puntos por casos de uso permite realizar estimaciones del tiempo de desarrollo de un determinado proyecto informático mediante la asignación de *pesos* a un cierto número de factores que lo afectan, para posteriormente, contabilizar el tiempo total estimado para el proyecto a partir de estos factores.

5.1.1 Cálculo de Puntos de Casos de Uso sin ajustar

El cálculo de los Puntos de Casos de Uso sin ajustar constituye el primer paso y se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

Donde:
UUCP: Puntos de Casos de Uso sin ajustar.
UAW: Factor de Peso de los Actores sin ajustar.
UUCW: Factor de Peso de los Casos de Uso sin ajustar.

Para calcular el Factor de Peso de los Actores sin ajustar (UAW) se analiza la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos. Esta complejidad puede definirse de la siguiente manera:

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar	1

	mediante una interfaz de programación (API, Application Programming Interface).	
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3

Tabla 5: Determina el Factor de Peso de los Actores sin ajustar (UAW).

En el Módulo de Integración se determinó un actor simple.

UAW= ∑ (actores*Peso)= 1 * 1 = 1

El Factor de Peso de los Casos de Uso sin ajustar (UUCW) se calcula analizando la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde la cantidad de transacciones está dada a partir de la descripción de los Casos de Uso.

Tipo de Caso de Uso	Descripción	Factor de Peso
Simple	El Caso de Uso contiene de 1 a 3 transacciones.	5
Medio	El Caso de Uso contiene de 4 a 7 transacciones.	10
Complejo	El Caso de Uso contiene más de 8 transacciones.	15

Tabla 6: Determina el Factor de peso de los Casos de Uso sin ajustar (UUCW).

En el Módulo de Integración se determinaron 3 tipos de casos de uso simple, 12 tipos de casos de uso medio y 1 caso de uso complejo; quedando el UUCW:

UUCW= ∑ (CU * Peso)= 3 * 5 + 12 * 10 + 1 * 15 = 150

Donde:
CU: Cantidad de Casos de Uso presentes en el Módulo de Integración.
Peso: Factor de Peso asociado a un tipo de caso de uso.

Por tanto los Puntos de Casos de Uso sin ajustar (UUCP) resultan:
UUCP= UAW + UUCW
UUCP= 1 + 150= 151

5.1.2 Cálculo de Puntos de Casos de Uso ajustados

Una vez obtenidos los Puntos de Casos de Uso sin ajustar se procede al cálculo de los Puntos de Casos de uso ajustados mediante la siguiente ecuación:

UCP = UUCP * TCF * EF

Donde:
UCP: Puntos de Casos de Uso ajustados.
UUCP: Puntos de Casos de Uso sin ajustados.
TCF: Factor de Complejidad Técnica.
EF: Factor de Ambiente.

El Factor de Complejidad Técnica (TCF) es un coeficiente que se calcula cuantificando un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de estos factores se cuantifica con valores de 0 a 5.

Significado de los Valores:
0: No presente o sin influencia.
1: Influencia incidental o presencia incidental.
2: Influencia moderada o presencia moderada.
3: Influencia media o presencia media.
4: Influencia significativa o presencia significativa.
5: Fuerte influencia o fuerte presencia.

Factor	Descripción	Peso	Valor asociado
T1	Sistema distribuido.	2	0
T2	Objetivos de performance o tiempo de respuesta.	1	4
T3	Eficiencia del usuario final.	1	1
T4	Procesamiento interno complejo.	1	1
T5	El código debe ser reutilizable.	1	4
T6	Facilidad de instalación.	0.5	2
T7	Facilidad de uso.	0.5	3
T8	Portabilidad.	2	3
T9	Facilidad de cambio.	1	4
T10	Concurrencia.	1	4
T11	Incluye objetivos especiales de seguridad.	1	3
T12	Provee acceso directo a terceras partes.	1	3
T13	Se requieren facilidades especiales de entrenamiento a usuarios.	1	1

Tabla 7: Determina el Factor de Complejidad Técnica (TCF).

Calculando el factor de Complejidad Técnica (TCF):

TCF= 0.6 + 0.01 * ∑ (Peso$_i$ * Valor Asignado)

TCF= 0.6 + 0.01 * 33.5 = 0.935

El Factor de Ambiente (EF) se calcula similar al Factor de Complejidad Técnica, se cuantifican con valores de 0 a 5. Los factores que se cuantifican están relacionados con las habilidades y el entrenamiento del grupo involucrado en el desarrollo del sistema.

Factor	Descripción	Peso	Valor asociado
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	3
E2	Experiencia en la aplicación.	0.5	1
E3	Experiencia en la orientación a objetos.	1	5
E4	Capacidad del analista líder.	0.5	5
E5	Motivación.	1	5
E6	Estabilidad de los requerimientos.	2	3
E7	Personal part-time.	-1	2
E8	Dificultad del lenguaje de programación.	-1	3

Tabla 8: Determina el Factor Ambiente (EF).

Calculando el Factor Ambiente (EF):

EF= 1.4 - 0.03 * ∑ (Peso$_i$ * Valor Asignado$_i$)

EF= 1.4 – 0.03 * 18.5 = 0.845

Por tanto los Puntos de Casos de Uso ajustados (UUCP) resultan:

UCP= UUCP * TCF * EF

UCP= 151 * 0.905 * 0.845 = 115.47

5.1.3 Cálculo de Esfuerzo

Para realizar el cálculo del esfuerzo, el primer paso es definir el factor de conversión horas-hombre/Punto de Casos de Uso que se debe utilizar para calcular el esfuerzo. Para ello se contabilizan cuántos factores

de los que afectan al Factor de Ambiente están por debajo del valor medio (3), para los factores E1 a E6, luego se contabilizan cuántos factores de los que afectan al Factor de Ambiente están por encima del valor medio (3), para los factores E7 y E8.

- Si el total es 2 o menos, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.
- Si el total es 3 o 4, se utiliza el factor de conversión 28 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 28 horas-hombre.
- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto, ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

El Módulo de Integración utilizará el factor de conversión 20 horas-hombre/Punto de Casos de Uso.

El esfuerzo en horas- hombre viene dado por:

E= UCP * CF

Donde:
E: Esfuerzo estimado en horas-hombre.
UCP: Puntos de Casos de Uso ajustados.
CF: Factor de conversión.

Calculando el esfuerzo en horas- hombre se obtiene como resultado:
E= 115.47 * 20 = 2039.4

5.1.4 Distribución del Esfuerzo entre las diferentes actividades

Para una estimación más completa de la duración total del Módulo de Integración, se agrega el esfuerzo obtenido por las demás actividades. Para ello se plantea la distribución del esfuerzo entre las diferentes actividades:

Actividad	Porcentaje esfuerzo	Horas-Hombre
Análisis	10%	509.85

Diseño	20%	1019.7
Implementación	40%	2039.4
Pruebas	15%	764.76
Otras Actividades	15%	764.76
Total	100%	5098.5

Tabla 9: Esfuerzo entre las distintas actividades de un proyecto.

El Esfuerzo Total obtenido sería 5098.5 horas-hombre, si se estima teniendo en cuenta que un mes tiene 192 horas laborables, pues se trabajan 8 horas de trabajos diarios y 24 días laborables en el mes, se estima que para la elaboración del Módulo de Integración es necesario 26.55 mes-hombre.

5.1.5 Calcular el Costo de todo el proyecto

El costo del proyecto se estimará teniendo en cuenta la siguiente ecuación:

Costo= CHM * ET / CH

Donde:
CHM: Costo Hombre – Mes.
ET: Esfuerzo Total (mes-hombre).
CH: Cantidad de Hombres.

Si la cantidad de hombres es 2 y se tiene un Salario Promedio de mensual igual a $100.00

CHM= CH * Salario Promedio
CHM= 2 * 100 = 200

Entonces el Costo sería

Costo= CHM * ET / CH
Costo= 200 * 26.55 / 2

Costo= $2655

5.1.6 Calcular el Tiempo de desarrollo de todo el proyecto

Teniendo en cuenta que la cantidad de hombres para la realización del Módulo de Integración es 2 el tiempo total sería:

Tiempo = ET / CH

Tiempo= 26.55 / 2

Tiempo = 13.27 ≈ 13 meses

5.2 Análisis de costos y beneficios

Para el desarrollo del Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles se emplearon herramientas libres. El Módulo de Integración va dirigido especialmente a CUBACEL y su principal objetivo es proveer que sistemas externos puedan comunicarse e integrarse con la “Plataforma de Gestión de Contenidos para Dispositivos Móviles”, permitiendo que sistemas externos puedan consumir los servicios que ofrece la plataforma. Los medios tecnológicos utilizados son los que se invirtieron en la facultad por parte de la administración de la Universidad, por lo que el módulo no supone altos gastos en recursos. Por todo lo planteado se puede llegar a la conclusión que es totalmente factible el desarrollo del Módulo de Integración.

Conclusiones

En este capítulo se representó el estudio de factibilidad correspondiente al sistema propuesto, así como la estimación, el cálculo de esfuerzo, el costo y el tiempo del desarrollo del proyecto y los beneficios que representará el Módulo de Integración al ser implementado.

CONCLUSIONES GENERALES

Este Trabajo de Diploma culmina dando cumplimiento al Objetivo General planteado, el cual consiste en el desarrollo de un Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles.

Durante el desarrollo del Módulo de Integración, se realizó un profundo análisis de los principales mecanismos de comunicación, herramientas y tecnologías a utilizar. Se realizó además un proceso de investigación de la comunicación de aplicaciones con sistemas externos; canales de comunicación seguros para la transmisión de datos, teniendo en cuenta el tamaño, la integridad, y seguridad en el envío. Para un buen entendimiento del Módulo de Integración, se definió el diseño del mismo, el cual le da soporte a los requisitos del sistema. Se determinó la utilización de la Arquitectura en 2 Capas, lo que simplificó la comprensión y organización del desarrollo del Módulo de Integración. Se utilizaron además los Patrones de Diseño los cuales brindaron soluciones a problemas comunes. Una vez definido el cuerpo o base del subsistema, se realizó el Diagrama de Despliegue, donde se representaron los nodos que tenían capacidad de procesamiento, así como los protocolos que permiten la comunicación entre ellos. Se realizaron pruebas software las cuales permitieron verificar y revelar la calidad del subsistema, las mismas se realizaron con el fin de identificar posibles fallos de implementación, calidad e usabilidad. Para la realización de las pruebas automatizadas unitarias y de integración al Módulo de Integración, se utilizó el framework JUnit; y para las pruebas de funcionalidad se empleó un modelo de prueba estático. La realización de estas pruebas de software permitió verificar el correcto funcionamiento de las funcionalidades del subsistema. Se efectuó un análisis para la estimación del proyecto, con vista a la determinación de la viabilidad del mismo, donde se pudo concluir que el proyecto es viable.

Al finalizar este Proceso de Desarrollo de Software, se ha obtenido como resultado un Módulo de Integración capaz de permitir que sistemas externos puedan proveer contenidos a la “Plataforma de Gestión de Contenidos para Dispositivos Móviles” y consumir los servicios que ofrece la misma; representando esto un avance tecnológico en el desarrollo de la telefonía móvil en Cuba.

RECOMENDACIONES

Como resultado de este Trabajo de Diploma se obtuvo un Módulo de Integración de la Plataforma de Gestión de Contenidos para Dispositivos Móviles y se recomienda continuar el estudio de los mecanismos de comunicación entre aplicaciones en búsqueda de soluciones óptimas para versiones futuras del producto; tanto para optimizar los servicios ya implementados como la incorporación de nuevos servicios a la Plataforma de Gestión de Contenidos para Dispositivos Móviles. Se recomienda además la promoción del Módulo de Integración en busca de contratos con Instituciones.

REFERENCIAS BIBLIOGRÁFICAS

1. Estado del Arte: Servicios Web. [book auth.] Carlos Andres Morales Mochuca. Universidad Nacional de Colombia : s.n.

2. OMG WE SET THE STANDARD. [Online] 1997-2011. [Cited: Enero 20, 2011.]

3. Programación en Castellano. [Online] [Cited: Enero 16, 2011.] http://www.programacion.com/articulo/rmi_mano_a_mano_con_ssl:_construyendo_aplicaciones_distribuidas_seguras_79.

4. Apuntes. Ingeniería del Software. [Online] [Cited: Enero 26, 2011.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

5. **Marina Ramírez, Miguel Anguel.** *Análisis y Diseño siguiemdo RUP para los Terminales.* 2009.

6. **Carrero, Un proyecto de los hermanos.** Introducción a UML. Programación en Castellano. [Online] 1999-2011. [Cited: Octubre 30, 2010.] http://www.programacion.com/articulo/introduccion_a_uml_181.

7. Visual Paradigm [Online] 2010 [Cited: Octubre 10, 2010.] http://www.visual-paradigm.com/product/vpuml/.

8. Especialista Universitario en JAVA ENTERPRICE Universidad de Alicante. [Online] 2002-2010. [Cited: Octubre 11, 2010.]. http://www.jtech.ua.es/j2ee/2007-2008/jee.html.

9. Eclipse Platform Technical Overview. [Online] [Cited: Octubre 11, 2010.] http://www.eclipse.org/articles/Whitepaper-Platform-3.1/eclipse-platform-whitepaper.html.

10. **Walls, Craig y Breidenbach, Ryan.** *Spring in Accion.* s.l. : Greenwick : Manning Publications Co, 2008.
11. Entorno Virtual de Aprendizaje. [Online] [Cited: Enero 25, 2011.] http://eva.uci.cu/mod/resource/view.php?id=21011.

12. **Jacobson, Ivar, Grady Booch y James Rumbaugh.** *El proceso unificado de desarrollo de software.* 2000.

13. Ayuda del Rational. Artefacto: Modelo de diseño. [Online] [Cited: Marzo 6, 2011.].

14. Tutorial de UML- Diagramas de Interración. [Online] http://www.dcc.uchile.cl/~psalinas/uml/interaccion.html.

15. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady. 2000.** *El Proceso Unificado de Desarrollo de Software***.** Madrid: s.n., 2000.

16. **Visconti, Marcello y Astudillo, Hernán**. *Fundamentos de Ingeniería de Software*. **Walls, Craig y Breidenbach, Ryan. 2008**. *Spring in Action*. *Segunda edición*. Greenwick: Manning Publications Co., 2008.

17. Modelo Basado En Componentes Diseño De Sistemas. [Online] [Cited: Abril 10, 2011.] http://www.mitecnologico.com/Main/ModeloBasadoEnComponentesDise%F1oDeSistemas.

18. Entorno Vistual de Aprendizaje. [Online] [Cited: Abril 23, 2011.] http://eva.uci.cu/mod/resource/view.php?id=14094.

19. Guía de Usuario de Enterprice Architect 7.0. [Online] [Cited: Mayo 5, 2011.] http://www.sparxsystems.com.ar/download/ayuda/index.html?deploymentdiagram.html.

20. Entorno Vistual de Aprendizaje. [Online] [Cited: Mayo 10, 2011.] http://eva.uci.cu/mod/resource/view.php?id=14094.

BIBLIOGRAFÍA

1. BuenasTareas.com. [Online] http://www.buenastareas.com/ensayos/Cuando-Se-Ivento-El-Telefono/22157.html.
2. Etecsa. [Online] http://www.etecsa.cu.
3. WEB SERVICES. [Online] http://www.gxtechnical.com/gxdlsp/pub/genexus/internet/technicalpapers/web_services.htm.
4. **Seguel, Ricardo**. *Seguridad en Web Services*. Mayo, 2009.
5. **Saffirio, Mario.** ¿Qué son los Web Services? . [Online] 2009.
6. **Piñol, Carles Mateu.** *Desarrollo de Aplicaciones Web .* 2001.
7. **Xancatl, Picén.** *Servicios de Búsqueda en una arquitectura de componentes GIS***.**
8. **Aguero, Martín.** *Introducción a Spring Framework.* Buenos Aires,Argentina : s.n., 2007.
9. **Céspedes, Juan.** *Automatización de tareas. Diseño y Administración de Sistemas y Redes.* 2005-2006.
10. **Crespo, César.** *Tutorial de Hibernate.* Madrid : s.n., 2004.
11. **Voos, Javier Alfredo.** *Diseño de Aplicaciones utilizando la plataforma J2EE.* Facultad Regional Córdoba : s.n.
12. Especialista Universitario en JAVA ENTERPRICE Universidad de Alicante.
13. Programación en Castellano. [Online] 2011. http://www.programacion.com/.
14. *Spring Security Reference Documentation.* 2005-2007.
15. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady. 2000.** *El Proceso Unificado de Desarrollo de Software***.** Madrid: s.n., 2000.

16. **Visconti, Marcello y Astudillo, Hernán**. *Fundamentos de Ingeniería de Software*. **Walls, Craig y Breidenbach, Ryan. 2008**. *Spring in Action*. *Segunda edición*. Greenwick: Manning Publications Co., 2008.

17. **Sommerville, Ian.** *Ingeniería del Software.* Madrid : Perason Educación, 2005. ISBN/84-7829-047.

18. **James Rumbaugh, Ivan Jacobson, Grady Booch.** *El lenguaje unificado de modelado. Manual de Referencia.*

19. **Miguel Angel, Marina Ramírez.** *Análisis y Diseño siguiendo RUP para los Terminales.* 2009.

20. Modelo Basado En Componentes Diseño De Sistemas. http://www.mitecnologico.com/Main/ModeloBasadoEnComponentesDise%F1oDeSistemas.

21. **Pelaez, Juan.** Juan Pelaez. [Online] Mayo 29, 2009. [Cited: Diciembre 5, 2010.] http://geeks.ms/blogs/jkpelaez/archive/2009/05/29/arquitectura-basada-en-capas.aspx.

22. **Rod Johnson, Juergen Hoeller, Keith Donald, Colin Sampaleanu, Rob Harrop, Alef Arendsen, Thomas Risberg, Darren Davison, Dmitriy Kopylenko, Mark Pollack, Thierry Templier, Erwin Vervaet, Portia Tung, Ben Hale, Adrian Colyer, John Lewis, Costin Leau, Mark.** *Spring Java Application Framework.* 2004-2009.

23. **Galán Jiménez, Miguel Angel.** *Framework para el desarrollo del encadenamiento interdominio con QsO basado en PCE sobre MPLS.* 2008.

24. *Un Método de Ingeniería Inversa de Código Java hacia Diagramas de Secuencias de UML 2.0.* **Carlos Mario Zapata, Oscar Andres Ochoa, Camilo Vélez.** 9, Colombia : s.n., 2008. ISSN 1794-1237.

25. **Xerox.** *El Lenguaje de Programación Java.* 2001.

26. *JPA (Java Persistence API).* 2008.

27. *PRUEBAS UNITARIAS EN JAVA – JUNIT.* [Online] 2009. [Cited: Mayo 20, 2011.] http://elverdaderoblogdelaob.wordpress.com/2009/07/04/pruebas-unitarias-en-java-junit/.

GLOSARIO DE TÉRMINOS

A

AOP (Aspect-Oriented Programming): Es un paradigma de programación que tiene como objetivo aumentar la modularidad, permitiendo la separación de las preocupaciones de corte transversal, constituye una base para el desarrollo de software orientado a aspectos.

APIs (A*pplication Programming Interfaz*): Conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

C

CVS (*Concurrent Versions System*): Es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros que forman un proyecto y permite que distintos desarrolladores colaboren.

E

EJB (*Enterprise JavaBeans*): Son una de las APIs que forman parte del estándar de construcción de aplicaciones empresariales. J2EE de Sun Microsystems. Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son, precisamente, los EJB.

F

Framework: Microarquitectura que proporciona una plantilla incompleta para sistemas de un determinado dominio. Puede tratarse por ejemplo, de un subsistema construido para ser ampliado y reutilizado.

H

HTML (*HyperText Markup Language*): Es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

J

JSP (*JavaServer Pages*): Es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

O

OMG (*Object Management Group*): Es un consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como UML, XMI, CORBA.

S

SGML (*Standard Generalized Markup Language*): Consiste en un sistema para la organización y etiquetado de documentos. El lenguaje SGML sirve para especificar las reglas del etiquetado de los documentos y no impone en sí ningún conjunto de etiquetas en especial.

SMS (*Short Message Service*): Es un servicio disponible en los teléfonos móviles que permite el envío de mensajes cortos.

SOA (*Service Oriented Architecture*): Permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma bien definida de exposición e invocación de servicios, lo cual facilita la interacción entre diferentes sistemas propios o de terceros.

SOAP (*Simple Object Access Protocol*): Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

SQL (S*tructured Query Language*): Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas.

T

TCP (*Transmission Control Protocol*): Es uno de los protocolos de red en los que se basa Internet y que permiten la transmisión de los datos entre computadoras.

W

WSDL (*Web Services Description Language*): Es un formato XML que se utiliza para describir servicios web.

X

XML (*eXtensible Markup Language*): Es un metalenguaje extensible de etiquetas. Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes especificados (del mismo modo que HTML es a su vez un lenguaje definido por SGML).

ANEXOS

Anexo I: Descripción de Casos de Uso

CU Autenticar Usuario

Caso de Uso:	Autenticar Usuario.
Actores:	Sistema Externo.
Resumen:	Permite a un sistema externo autenticarse en la aplicación.
Referencia	RF 1
Precondiciones:	-

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El sistema externo introduce los datos de ingreso al sistema.	2. El sistema comprueba los datos con los almacenados en la base de datos.

Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1 El Sistema muestra un mensaje de error si la operación no pudo ser realizada.

Poscondiciones	-

CU Modificar Datos

Caso de Uso:	Modificar Datos.

Actores:	Sistema Externo.
Resumen:	Modificar los datos del sistema externo.
Precondiciones:	El sistema externo debe estar autenticado en el sistema.
Referencias:	RF 2.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El sistema externo introduce los datos a modificar.	2. El sistema modifica los datos en la base de datos.
	3. El Sistema muestra un mensaje de confirmación.

Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	3.1 El Sistema muestra un mensaje de error si la operación no pudo ser realizada.

Poscondiciones	Quedan gestionados los datos del sistema externo.

Anexo II: Diagramas de Secuencias

CU Autenticar Usuario

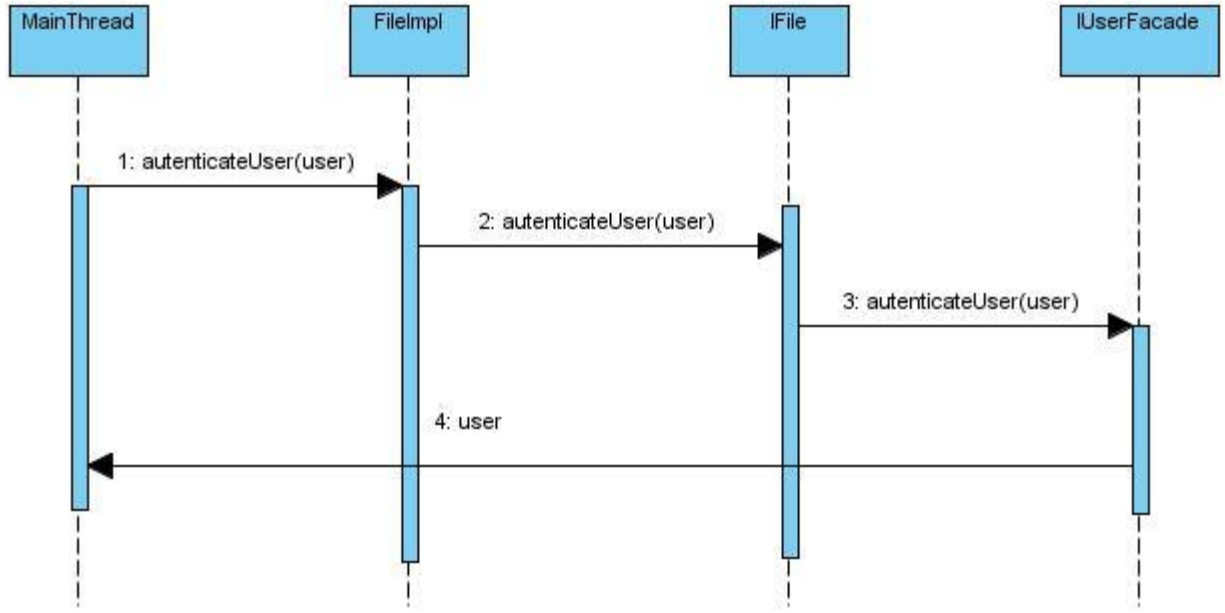

Figura 23: Diagrama de secuencias del Caso de Uso Autenticar Usuario (rmi).

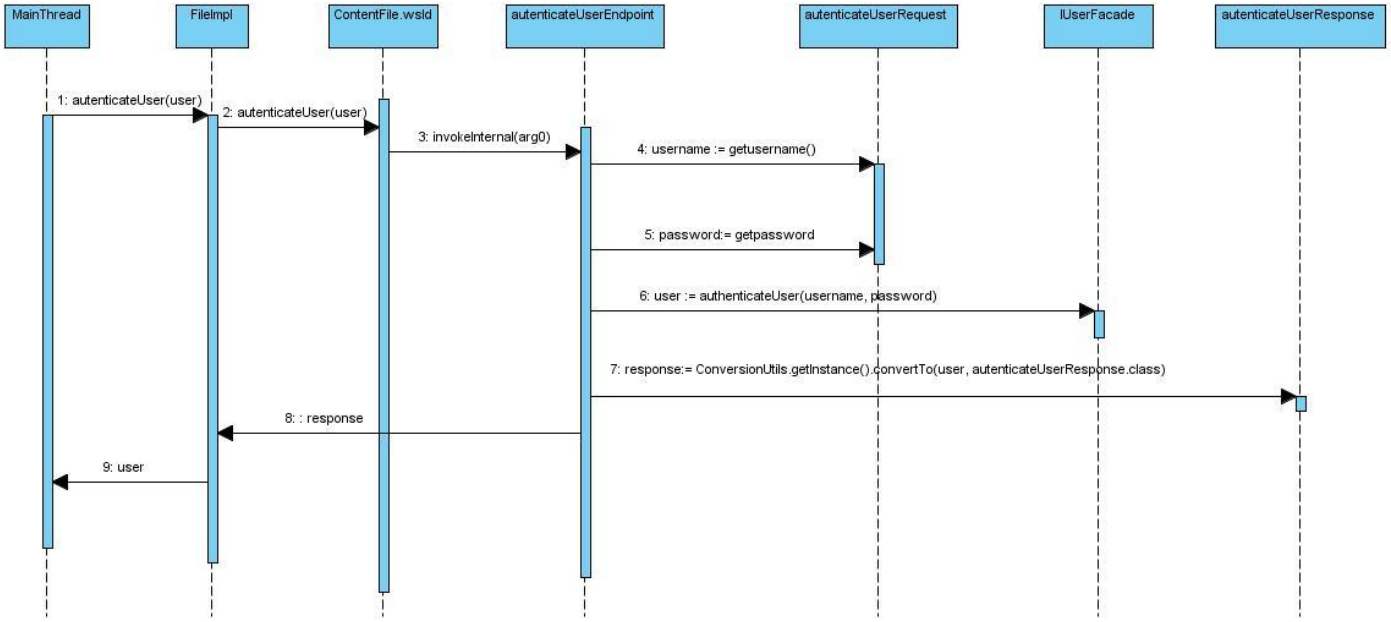

Figura 24: Diagrama de secuencias del Caso de Uso Autenticar Usuario (webservice).

CU Insertar Contenido

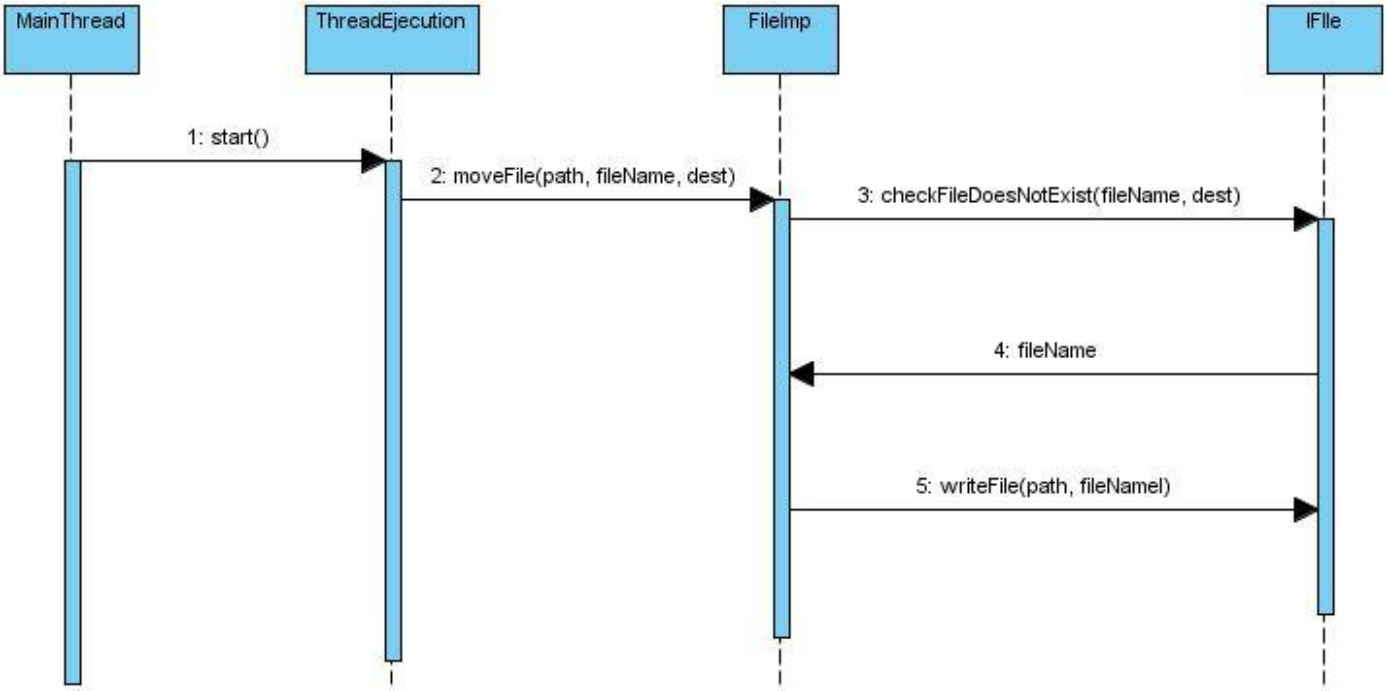

Figura 31: Diagrama de secuencias del Caso de Uso Insertar Contenido (rmi).

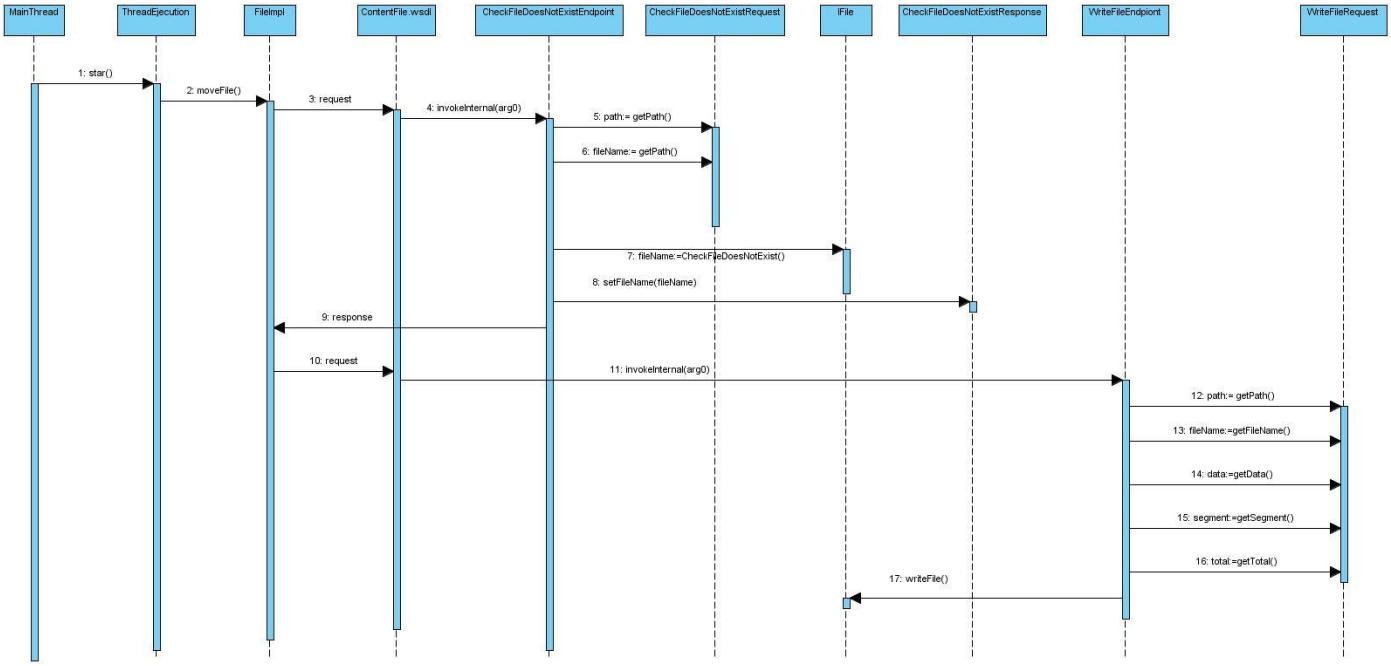

Figura 32: Diagrama de secuencias del Caso de Uso Insertar Contenido (webservice).

Anexo III: Pruebas Unitarias

CP Insertar Contenido

```
- <testrun name="PruebaJUNIT" project="PruebaRMIClient" tests="1" started="1" failures="0" errors="0" ignored="0">
  - <testsuite name="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="19.218">
      <testcase name="moveFile" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="19.218"/>
    </testsuite>
  </testrun>
```

Figura 49: Prueba Unitaria del Caso de Prueba Insertar Contenido (rmi).

```
- <testrun name="PruebaJUNIT (2)" project="ComcelWS-Client" tests="1" started="1" failures="0" errors="0" ignored="0">
  - <testsuite name="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="75.622">
      <testcase name="moveFile" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="75.622"/>
    </testsuite>
  </testrun>
```

Figura 50: Prueba Unitaria del Caso de Prueba Insertar Contenido (webservice).

Anexo IV: Pruebas de Integración

```
- <testrun name="PruebaJUNIT" project="ComcelRMI-Client" tests="10" started="10" failures="0" errors="0" ignored="0">
  - <testsuite name="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="125.701">
      <testcase name="moveFile" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="117.326"/>
      <testcase name="isAuthority" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="5.719"/>
      <testcase name="findAllCategories" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.375"/>
      <testcase name="findAllContentByCategory" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.281"/>
      <testcase name="findAllContentByDevice" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.0"/>
      <testcase name="findAllSubCategoriesByIdCategory" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.079"/>
      <testcase name="findAllSubCategoriesByIdSubCategory" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="1.203"/>
      <testcase name="findAllUsers" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.047"/>
      <testcase name="findContentByCode" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.671"/>
      <testcase name="findContentById" classname="cu.uci.comcel.core.remote.rmi.PruebaJUNIT" time="0.0"/>
    </testsuite>
  </testrun>
```

Figura 59: Prueba de Integración (rmi).

```
− <testrun name="PruebaJUNIT (1)" project="Comcel-WS-Client" tests="10" started="10" failures="0" errors="0" ignored="0">
  − <testsuite name="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="137.954">
      <testcase name="moveFile" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="134.048"/>
      <testcase name="isAuthority" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.547"/>
      <testcase name="findAllCategories" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.906"/>
      <testcase name="findAllContentByCategory" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.61"/>
      <testcase name="findAllContentByDevice" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.0"/>
      <testcase name="findAllSubCategoriesByIdCategory" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.14"/>
      <testcase name="findAllSubCategoriesByIdSubCategory" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="1.406"/>
      <testcase name="findAllUsers" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.032"/>
      <testcase name="findContentByCode" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.172"/>
      <testcase name="findContentById" classname="cu.uci.comcel.core.remote.webservice.PruebaJUNIT" time="0.093"/>
    </testsuite>
  </testrun>
```


Figura 60: Prueba de Integración (webservice).

Anexo V: Pruebas de Funcionalidad

```
import cu.uci.comcel.core.remote.rmi.impl.MainThread;

public class Test  {


    private static MainThread mainThread= new MainThread();

    public static void main(String[] args) throws Exception {

        System.out.println(" Begining service of copy by RMI");

        mainThread.AddContent("D:\\(2009) Against The Dark\\Against the Dark.avi","Against the Dark.avi",
                "C:\\RMI","Alfonso","cualquiera");
        mainThread.AddContent("D:\\Harold y Rumar (comedia)\\Harold y Rumar (comedia).mpg",
                "Harold y Rumar (comedia).mpg","C:\\RMI","Alfonso","cualquiera");
        mainThread.AddContent("D:\\MOV03024.mpg","MOV03024.mpg","C:\\","nicko","nicko");
        mainThread.AddContent("D:\\MOV03024.mpg","MOV03024.mpg","C:\\","Alfonso","cualquiera");
        System.out.println(mainThread.getUserByLogin("Alfonso"));
        System.out.println(mainThread.findAllAuthorities());
        System.out.println("All file is addiction at cola for to copy");
```

```
Console
Test [Java Application] C:\Archivos de programa\Java\jre6\bin\javaw.exe (04/06/2011 15:13:58)
Sending segment number:83
Sending segment number:84
Sending segment number:85
Sending segment number:86
Sending segment number:87
Sending segment number:88
Sending segment number:89
```

Figura 61: Prueba de Funcionalidad (rmi).

```java
public class Test {

    private static MainThread hilos= new MainThread("http://10.31.18.231:8080/comcel/service/");

    public static void main(String[] args) throws Exception {
        System.out.println(" Begining service of copy by WebService");
        hilos.Add_Content("D:\\Yanni.mpg","Yanni.mpg","C:\\WebService",
                "Alfonso","cualquiera");
        System.out.println(hilos.findAllAuthorities().get(1).getAuthority());
        System.out.println(hilos.findAllCategories(0, 2).get(1).getCategoryId());
        System.out.println(hilos.findSubcategoryById(102).getDescription());
        System.out.println(hilos.findAllUsers(0, 2).isEmpty());
        System.out.println("All file is addiction at cola for to copy");
    }
}
```

Console

<terminated> Test (1) [Java Application] C:\Archivos de programa\Java\jre6\bin\javaw.exe (04/06/2011 15:25:10)

```
 Begining service of copy by WebService
Begining copy of file Yanni.mpg by Thread-2
ROLE_PROVIDER
11415
Música Loca
```

Figura 62: Prueba de Funcionalidad (webservice).