Universidad de las Ciencias Informáticas

Facultad 2

Título: Aplicación de Administración para Proveedores de la Plataforma de Entrega de Contenidos para Móviles.

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

Autores: Giselle María Capote Chávez.
Yolanda Mérida Aguirre.

Tutor: Ing. Ismaila López Sotolongo.

Co-Tutor: Ing. Miguel Ángel Torres Pérez.

Co-Tutor: Deivis Ricardo Álvarez Mendoza.

La Habana, Cuba

"Año 54 de la Revolución"

Junio, 2012

Pensamiento:

¨La paz viene como necesaria consecuencia del trabajo: pero el trabajo no se alimenta cuando no puede tener la esperanza de realizar y mejorar sus productos.¨

José Martí.

Declaración de Autoría:

Declaración de autoría

Declaramos ser las únicas autoras de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Giselle María Capote Chávez

_____

Firma del autor

Yolanda Mérida Aguirre

_____

Firma del autor

Ing. Miguel Ángel Torres Pérez

_____

Firma del co - tutor

Ing. Deivis Ricardo Álvarez Mendoza.

_____

Firma del co - tutor

Ing. Ismaila López Sotolongo

_____

Firma del tutor

Dedicatoria:

A Dios por darme la oportunidad y la dicha de la vida, al brindarme los medios necesarios para continuar mi formación como profesional, sin él no hubiera podido.

A ti Madre, lo más grande que tengo en este mundo:

Mi madre fue la mujer más bella que jamás conocí. Todo lo que soy, se lo debo a ella .Te atribuyo todos mis éxitos en esta vida, a la enseñanza moral, intelectual y física que recibí de ti. Por haberme educado y soportar mis errores. Gracias por tus consejos, por el amor que siempre me has brindado, por cultivar e inculcar ese sabio don de la responsabilidad en mí.

¡Gracias por darme la vida!

¡Te quiero mucho!

Yolanda.

Dedicatoria:

Le dedico este trabajo a mis padres que son mi razón de ser, por darme la vida y guiarme por el camino correcto, sin su ayuda no hubiera podido llegar a lo que soy hoy. Los adoro.

A mi hermanita y mi sobrinita, las dos son mi vida, las amo con el corazón.

A mi familia por ser tan unida y ser el mejor regalo que me ha dado la vida.

Gracias a todos por confiar en mí.

Giselle.

Agradecimientos:

Mis agradecimientos son para:

Mi mamá: *Por estar en todo momento para mí, por su apoyo y amor incondicional, por todos sus esfuerzos, por hacerme una persona de bien, por estar orgullosa de lo que soy, por apoyarme siempre en todas las decisiones que he tomado y por sobre todo haber tenido mucha paciencia en todos aquellos duros momentos que pasamos al inicio de la carrera.*

A mi papá Roberto: *Por darme el mejor ejemplo todos estos años que has estado a mi lado, por brindarme todo el amor y cariño que necesitaba de un padre y apoyar a mi mamá en todos esos momentos que se refieran a mí.*

A mi papá Raúl: *Por estar en todos los momentos en los que mi mamá y yo lo necesitábamos, por haber hecho posible mi existencia en este mundo, por el cariño y respeto que me has brindado.*

A mi tata: *Mi hermanita querida, gracias por apoyarme en todo momento, por estar para mí en los momentos cuando te necesitaba.*

Agradecimientos:

Familia: *A mi tía Moraima que ha estado a mi lado en todo momento comportándose como la segunda madre que es para mí, a mi abuela que la quiero mucho, al resto de mis tías, mis primos.*

A Alfredo: *Por haber estado en los momentos difíciles y apoyarme cuando más lo necesitaba.*

A todas esas personas que creyeron en mí, a todos los profesores que he tenido durante mi carrera estudiantil, dándome ejemplos dignos de superación y entrega, a mis amigos de años: Osvaldo , a Dilonyx (Kuki) mi segunda hermanita, a Jorge(tito) persona de admirar, Giselle la mejor compañera de tesis que pude tener, Yaneysi ejemplo de una gran mujer, Yami, Karina, Yuliani, a mi cotutor Miguel que me apoyo mucho y lo admiro de todo corazón, al profe Deivis que nos brindó sus conocimientos, a mi tutora, Joan, a los de tribunal , porque en gran parte gracias a ustedes, hoy puedo ver alcanzada mi meta. Agradezco a nuestro Comandante en Jefe Fidel Castro, por hacer posible la oportunidad de convertirnos en hombres de ciencia y realizar nuestros sueños más preciados.

Yolanda.

Agradecimientos:

Agradecer primeramente a mis padres: *Por darme la vida, por estar en todo momento junto mí, por brindarme su apoyo y amor incondicional, por todo su esfuerzo, por hacerme una persona de bien, por estar orgullosos de mí, por ser los mejores padres del mundo.*

A mi hermanita *por apoyarme y ser mi confidente.* ***A mi niña chiquitica*** *a quien adoro y espero ser un ejemplo a seguir para ella.*

A mi familia*: En especial mis tías Mary y Cachy, gracias por poder contar con ustedes y ser mis segundas mamás. A mis abuelitas, mis hermanitos Luismy y Yarenis, a mi hermano mayor Hanoy, a mis demás primos y tíos. Agradecer a mi tía Nena, Magdalena y Nilda gracias por contar con su apoyo durante mi carrera. A mi tío Omarito por ser mi segundo papá y a mi tía Daymara, gracias a los dos por apoyarme, darme buenos consejos y guiarme por el buen camino. A mis abuelitos Tiburcio y Herminio porque sé están orgullosos de la persona que me he convertido, los quiero mucho estén donde estén. En fin a toda mi familia gracias por confiar en mí.*

A Carlos*: Porque durante los 4 años que estuvimos juntos estuvo siempre presente para mí en los buenos y malos momentos, gracias por creer y confiar en mí, por tu ayuda, por hacerme feliz, por cuidarme, por tu apoyo y por apoyarme cuando más lo necesitaba.*

Agradecimientos:

A mis mejores amigos de Cienfuegos: *Indira, Héctor y Edel, gracias por estar junto a mí y apoyarme en los buenos y malos momentos, los quiero mucho, los tres son un ejemplo a seguir.*

A mis mejores amigos de la Universidad: *Entre ellos citar a Cuca y Yoli mis mejores amigas y hermanitas aquí en la escuela, gracias a las dos por darme buenos consejos, escucharme y estar conmigo siempre. A mi mejor amigo Ernesto gracias por apoyarme y estar siempre presente para mí, te quiero mucho. Agradecer también a otros muy buenos amigos entre ellos a Albey, Yasmani, Alexeis, Yankiel, Wilder, Yudi, Yamila, Karina, Yuliani, Yaneysi, Ivette, en fin a todos los que me acompañaron durante mi carrera.*

A todos los profes que hicieron posible mi formación como Ingeniera en Ciencias Informáticas, mis compañeros del proyecto, a todas las personas que de una forma u otra hicieron posible la realización de este trabajo, especialmente a mis cotutores Miguel Angel y Deivis, agradecer también a la tutora Ismaila, al oponente Adonys y al tribunal Osmany, Erick y Ramon Alfonso.

A la Revolución, a la Universidad de las Ciencias Informáticas, al comandante en Jefe Fidel Castro Ruz, por ser el creador de esta universidad real que una vez fue su sueño y a todas las personas que me quieren. A todos muchas gracias.

Giselle.

Resumen:

En la actualidad el uso de la telefonía celular se ha incrementado considerablemente a medida que han pasado los años. El desarrollo de la telefonía móvil ha provocado que los usuarios sean más estrictos en cuanto a la variedad y gusto de los contenidos que necesitan.

En Cuba, el aumento de la telefonía móvil ha provocado que la Empresa de Telecomunicaciones de Cuba S.A (ETECSA), principalmente la Unidad de Negocios Móvil de ETECSA, Cubacel, se encuentre en la necesidad de contar con una plataforma que permita a los proveedores de contenidos administrar dichos contenidos. Como parte del desarrollo de la plataforma se hace necesario implementar un mecanismo que permita a los proveedores de contenidos poder interactuar con la misma de una forma sencilla y amigable.

Esta plataforma está pensada para brindar servicios tales como la gestión de los contenidos, ya sea insertar, eliminar, modificar y buscar contenidos, así como ofrecer diferentes reportes a los proveedores, tales como conocer de los contenidos que publicó, cuales fueron un éxito para los usuarios móviles, si fallaron o si fueron cancelados por los administradores. Además podrá saber que contenidos fueron más descargados, la facturación de su ganancia, los últimos descargados, entre otros servicios.

Con el desarrollo de este trabajo de diploma con título “Aplicación de administración para proveedores de la plataforma de entrega de contenidos para móviles.” se presenta una propuesta de solución informática que facilita el acceso de los proveedores a cada una de las funcionalidades que provee dicha plataforma. En el mismo se abordan los fundamentos teóricos, así como la descripción de la metodología de desarrollo, las herramientas y tecnologías seleccionadas para el desarrollo del sistema. Se específica la arquitectura sobre la cual se basa el desarrollo y los diagramas que ilustran los elementos del diseño e implementación de la aplicación. El resultado final del trabajo es un sistema que provee una interfaz de usuario a los proveedores para cada una de las funcionalidades que brinda la plataforma.

Palabras Claves: Contenidos, ETECSA, Cubacel, telefonía móvil, proveedores, plataforma, servicios, reportes, facturación.

Índice de Contenidos:

Índice de Contenidos:

Índice de Contenidos:

Índice de Contenidos:

Introducción

Las tecnologías inalámbricas se han acrecentado y desarrollado en estos últimos años en gran medida, en especial la telefonía celular. A partir del siglo XXI, los teléfonos móviles han adquirido funcionalidades que van más allá de llamar o enviar mensajes de texto.

El avance de la tecnología ha hecho que estos dispositivos incorporen funciones que antes parecían futuristas. En el mundo existen empresas proveedoras de servicios como Volantis (1) y BeeWeb (2). Estas empresas proveedoras de servicios para móviles, no solo brindan el servicio, sino que gestionan información relacionada a estos, por ejemplo: datos de los proveedores de contenidos, el costo de un contenido, cantidad de suscripciones o permisos sobre un servicio, entre otras funcionalidades.

La Empresa de Telecomunicaciones de Cuba S.A., ETECSA, es la encargada de operar la telefonía celular en Cuba, esta empresa tiene una alta responsabilidad en el desarrollo socio-económico del país, y en especial en la informatización de la sociedad, ya que garantiza la infraestructura tecnológica para una efectiva conectividad. Como parte de su crecimiento, Cubacel promueve el desarrollo de sus propias herramientas para brindar sus servicios. Una de estas herramientas es la plataforma de servicios para telefonía móvil, COMCEL.

El Proyecto COMCEL perteneciente al Centro de Telemática de la Facultad 2 de la Universidad de las Ciencias Informáticas (UCI) desarrolla una plataforma de entrega de contenidos para móviles, esta cuenta con una aplicación de administración que permite un conjunto de actividades a los usuarios según su rol, dígase proveedores de contenidos o administradores. Actualmente la plataforma brinda servicios como el estado del tiempo, noticias, búsqueda en páginas amarillas y descargas de contenidos; además está orientada a servicios, y pensada para desplegarse de manera distribuida.

Se encuentra disponible un solo servidor para realizar el despliegue de la plataforma. Debido a la carencia de servidores, los cambios en las fuentes de información de los servicios, así como en la estructura de la información y el sistema de almacenamiento de los contenidos provocaron la insatisfacción de las necesidades del cliente para la administración de la plataforma.

Introducción:

Dada la situación antes expuesta surge el siguiente **problema a resolver**: ¿Cómo adecuar la administración de los contenidos por los proveedores en la plataforma de descarga de contenidos para móviles según los nuevos requerimientos definidos por ETECSA?

Según el problema identificado anteriormente el **objeto de estudio** se enmarca en los sistemas de gestión de contenidos para telefonía móvil, y el **campo de acción** en la administración de los servicios y contenidos para teléfonos móviles en la Plataforma de gestión de contenidos para dispositivos móviles.

El **objetivo general** de la investigación es desarrollar un sistema de administración que permita la interacción de los Proveedores de contenidos con la plataforma de entrega de contenidos para móviles.

Con el fin de alcanzar el objetivo planteado se definen los siguientes **objetivos específicos:**

- ✓ Desarrollar un módulo de administración de usuario.
- ✓ Desarrollar un módulo de administración de contenidos.
- ✓ Asignar derechos digitales por contenidos específicos.

Para el desarrollo del trabajo se han formulado las siguientes **preguntas de investigación:**

1. ¿Cómo desarrollar aplicaciones web adaptables a las funcionalidades existentes en la plataforma Comcel?
2. ¿Qué mecanismos de seguridad pueden utilizarse en aplicaciones web?
3. ¿Cómo desarrollar aplicaciones web con funcionalidades necesarias para adoptar mecanismos de seguridad?
4. ¿Cómo generar reportes en formato PDF en un sistema web?
5. ¿Qué librerías utilizar para generar reportes en formato PDF?
6. ¿Cómo asegurar la protección de los derechos digitales de contenidos de los proveedores?

Introducción:

Con el propósito de cumplir con todos los objetivos planteados se definen las **siguientes tareas de investigación:**

1. Caracterizar aplicaciones de administración para proveedores en las plataformas de descargas de contenidos para móviles existentes en la actualidad.
2. Evaluar y seleccionar las herramientas, y metodologías de desarrollo de software más adecuadas para la construcción del sistema.
3. Redefinir la arquitectura de la aplicación.
4. Definir las librerías necesarias para generar reportes en formato PDF.
5. Definir estándares DRM a aplicar a los contenidos.
6. Realizar el diseño del sistema.
7. Implementar funcionalidades de la aplicación.
8. Realizar pruebas al sistema.

Con el cumplimiento de estas tareas se pretende la implementación de una aplicación para la gestión de los servicios que brinda la Plataforma de Gestión de Contenidos para Dispositivos Móviles, la misma permitirá que Cubacel cuente con una herramienta que gestione los servicios que proporcionan los proveedores.

Métodos Científicos:

Los métodos científicos utilizados en el siguiente trabajo son los siguientes:

- **Métodos Teóricos:**
  - ✓ Analítico–Sintético: Se utiliza durante el proceso de revisión bibliográfica para conocer las principales técnicas, tecnologías, metodologías de desarrollo de software con el fin de lograr la confección de la aplicación. Se puede comprobar la utilización de este método en el Capítulo 1.

- ✓ Análisis Histórico – Lógico: A través de este método se establece la necesaria correspondencia entre los elementos de los métodos lógico e histórico, proyectando el análisis de la evolución histórica de los fenómenos con la proyección lógica de su comportamiento futuro. Permite realizar un estudio de los antecedentes y tendencias actuales de los proveedores de contenidos existentes. Se puede comprobar la utilización de este método en el Capítulo 1.

- ✓ Inductivo-Deductivo: La inducción expresa el movimiento de lo particular a lo general, o sea se llega a generalizaciones partiendo del análisis de casos particulares, mientras la deducción expresa el movimiento de lo general a lo particular. Se puede comprobar este método en la utilización de las bibliografías.

❖ **Métodos empíricos:**

- ✓ Observación: Todo proceso científico comienza con la **observación** directa o indirecta de los fenómenos que ocurren o existen. Se puede comprobar la utilización de este método en el Capítulo 1.

El presente documento consta de 4 capítulos, estructurados de la siguiente forma:

Capítulo 1: Fundamentación Teórica.

En este capítulo se analizan aspectos teóricos que serán necesarios investigar para la correcta realización del trabajo. Se describen las metodologías de desarrollo de software, herramientas a utilizar y características de los sistemas de gestión y administración de servicios para telefonía móvil. Se definen las herramientas, tecnologías y metodología a usar para la construcción del sistema.

Capítulo 2: Características del Sistema.

Se realiza una descripción detallada de los casos de uso que guiarán el desarrollo del software, donde una vez identificados y clasificados los requerimientos del sistema es necesario identificar los actores. Se procede a la realización del caso de uso, así como las descripciones de cada uno de estos.

Capítulo 3: Diseño del Sistema.

En este capítulo se realizará la descripción de la arquitectura base del sistema a partir de las tecnologías seleccionadas para la construcción del mismo, de los procesos que se llevan a cabo en la disciplina de diseño, obteniendo como resultado los artefactos más importantes para modelar el sistema, teniendo en cuenta los patrones de diseño que aportan soluciones concretas a problemas específicos, para lograr un diseño eficaz en el software orientado a objetos.

Capítulo 4: Implementación y Prueba.

El presente capítulo se enfoca en desarrollar los artefactos correspondientes a la implementación tomando como entrada los resultados obtenidos en la etapa de diseño. Se representa el diagrama de componentes que detalla la forma en que está estructurado el sistema, reflejando la transformación de los elementos del modelo del diseño en términos de componentes, así como las dependencias entre ellos. Además se diseñan las pruebas a realizarle al sistema, los casos de pruebas para comprobar el correcto funcionamiento de las principales funcionalidades de la aplicación y se valida el sistema obtenido a través del empleo de pruebas de caja negra.

Capítulo 1: Fundamentación Teórica

Introducción:

En este capítulo se enuncian los principales conceptos teóricos que constituyen la base de la investigación realizada. Se analizan las características fundamentales de las tecnologías, metodología y herramientas que se utilizarán en el desarrollo del sistema. Se realiza un estudio del arte de algunas soluciones existentes en el mundo para la gestión de contenidos para dispositivos móviles.

1.1. Conceptos Fundamentales:

1.1.1. Contenidos:

Un contenido es la información sustantiva o material creativa que se ve en contraste con su forma real o potencial de la presentación. El contenido es cualquier texto, imagen o sonido que los usuarios se encuentran en una interfaz de software.

Contenido de la web es el contenido textual, visual o auditivo que se encuentra como parte de la experiencia del usuario en los sitios web. (3)

1.1.2. Gestión de Contenidos para Móviles:

En la actualidad existe gran demanda de contenidos para celulares: ringtones[1], fondos, animaciones, videos, programas y juegos, lo que provoca un incremento en las empresas encargadas de facilitar aplicaciones para la descarga de contenidos.

Los sistemas de gestión de contenidos móviles (MCMS) son un tipo de sistema de gestión de contenidos (CMS) capaces de almacenar, entregar contenido y servicios para dispositivos móviles, como teléfonos móviles, teléfonos inteligentes. La entrega de contenido móvil que presentan restricciones únicas y específicas, incluidas las capacidades de dispositivos muy variables, tamaños de la pantalla, anchos de

[1] Ringtones: Fichero con formato de sonido usado para indicar cuando se recibe una llamada o mensaje en el teléfono móvil.

banda inalámbrico limitado, poca capacidad de almacenamiento y los procesadores de dispositivos relativamente débiles ocasionó que la demanda de gestión de contenidos móviles se hiciera omnipresente y sofisticada. (4)

La tecnología MCMS se centró inicialmente en las empresas consumidoras (B2C) de mercado móvil con tonos, juegos, mensajes de texto, noticias y otros contenidos relacionados. Desde entonces, los sistemas móviles de gestión de contenidos también han tenido gran demanda en el negocio para empresas (B2B) y empresas que establecen relación comercial entre ellas y sus propios empleados (B2E), permitiéndoles proporcionar información más oportuna y funcionalidad a los socios comerciales y empleados móviles de una manera más eficiente.

1.1.3. Proveedores:

Son proveedores aquellos que proveen o abastecen, o sea que entregan bienes o servicios a otros.

1.1.4. Proveedores de Contenidos:

Se estará refiriendo a aquellas empresas que proveen a los usuarios móviles diferentes contenidos, por ejemplo: música, imágenes, videos, entre otros.

1.1.5. Administradores:

Son aquellos usuarios que tienen control total de los contenidos y usuarios presentes en la plataforma.

1.1.6. Interfaz de Usuario:

Una interfaz se define como el entorno o herramienta que permite una conexión física y funcional entre dos aparatos (5). Las interfaces de usuarios han evolucionado logrando ser más amigables, intuitivas y prácticas; en los inicios se trabajaba con consolas a modo de texto. En este tipo de interfaces los usuarios necesitan conocer los comandos a introducir para ser procesados y obtener los resultados. La interfaz de consola tiene una limitada capacidad expresiva, las primeras solo contaban con un tipo de texto sin colores y una cantidad de caracteres limitado. Aunque con el desarrollo de la informática se han ido enriqueciendo las posibilidades de estas terminales, las ventajas expresivas que brinda la interfaz gráfica son muy superiores.

Una interfaz web cuenta con cuatro características fundamentales, las mismas son: contenido, tecnología, aspectos visuales y económicos. El contenido es a través del cual se informa a los usuarios; la tecnología es quien ofrece la funcionalidad, interactividad y el dinamismo del sistema; los temas visuales influyen en el impacto visual de la aplicación hacia la audiencia; y las implicaciones económicas en la factibilidad de la construcción del sistema. (6)

1.1.7. Aplicación Web:

Una aplicación web es un software basado en tecnologías. El término también puede significar una aplicación de software que se aloja en un entorno de navegador controlada o codificada en un navegador con el apoyo del lenguaje (como JavaScript, junto con un navegador de lenguaje de marcas como HTML) y dependen de un navegador web común para hacer que la aplicación sea ejecutable.

Las aplicaciones web son basadas en tecnologías y estándares de W3C[2] que provee recursos específicos tales como contenidos y servicios a través de una interfaz de usuario a la que puede accederse utilizando un navegador web.

Las aplicaciones web utilizan documentos web escritos en un formato estándar como HTML y JavaScript, que son apoyados por una variedad de navegadores web. Las aplicaciones web pueden ser considerados como una variante específica de software cliente-servidor donde se descarga el software del cliente a la máquina cliente al visitar la página web correspondiente, utilizando los métodos convencionales, tales como HTTP.

Beneficios:

- ✓ Se tiene mayor control de datos y mejor seguridad en las diferentes secciones del website.
- ✓ No se requieren complicadas combinaciones de Hardware/Software para utilizar estas aplicaciones. Solo un computador con un buen navegador Web.
- ✓ Las aplicaciones web son fáciles de usar y no requieren conocimientos avanzados de computación.

[2] World Wide Web Consortium: Consorcio internacional de compañías y organizaciones involucradas en el desarrollo de Internet y en especial de la WWW. Su propósito es desarrollar estándares y “poner orden” en Internet.

- ✓ Tiene alta disponibilidad, ya que se pueden realizar consultas en cualquier parte del mundo donde tenga acceso a Internet y a cualquier hora. (7)

1.2. Empresas Proveedoras de Servicios para Móviles:

Existen empresas que se han dedicado a la gestión de contenidos para los dispositivos móviles. Algunas de las que se pueden mencionar “Content Delivery Platform”, de la empresa Volantis, y “Mobile Web & Content Delivery Platform” creado por la empresa BeeWeb.

- ❖ **Content Delivery Platform:**

Volantis Content Delivery Platform (V-CDP) proporciona una plataforma de software integrada para la entrega de contenido variado, innovador y personalizado, dentro de una experiencia de usuario sin fisuras y convincente, para el mayor número de abonados móviles. Permite a los proveedores de servicios de valor añadido la gestión y entrega de contenidos multimedia a escala general. Además presenta un completo juego de herramientas para el desarrollo, gestión y entrega de los contenidos. (1)

- ❖ **Mobile Web & Content Delivery Platform:**

Mobile Web & Content Delivery Platform (MWCDP) surge como resultado de la experiencia y el grado de madurez alcanzado por BeeWeb en el suministro de soluciones para la administración de contenidos. La plataforma se basa en un sistema de centralización de funcionalidades y permite que los contenidos se puedan adaptar para cualquiera de los canales de distribución soportados. (2)

En Cuba, a pesar de no contar con un desarrollo de punta en la gestión de contenidos para móviles, se están desarrollando algunos proyectos para responder al crecimiento de la telefonía celular.

Con todo este crecimiento las demandas de contenidos se han acelerado a la par de que los usuarios se hacen más exigentes y variados. Se hace necesario entonces desarrollar soluciones para la gestión de contenidos. Como referencia de desarrollos de plataformas de este tipo en el país se encuentra Procyon Soluciones[3] que les ofrece a los usuarios de teléfonos móviles servicios basados en mensajes de texto o SMS.

[3] Procyon Soluciones: División de Telecomunicaciones de DESOFT S.A y Cubacel.

Al pasar los años ha aumentado en gran medida la posibilidad de que haya más usuarios con dispositivos móviles, por ejemplo antes del 2008 la cantidad de usuarios con celulares era de 331270 que representa un 7%, mientras que se espera para el 2015 un aumento considerable de 2400000 que representa un 52%, superando al 100% lo que hasta el momento se había obtenido en años anteriores. (8)

Por ello se hace necesaria esta plataforma ya que constituirá para Cuba sustituir importaciones de diferentes componentes que proveen las mismas funcionalidades a un costo elevado, además con la puesta en marcha de este sistema la empresa Cubacel permitirá ofrecer mejoras en los servicios que brinda.

1.3. Tecnologías y Herramientas:

1.3.1. Lenguaje de programación:

Java:

Ofrece todas las funcionalidades de un lenguaje potente, trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo.

Java se ha construido con extensas capacidades de interconexión TCP/IP, realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo. Cuando se usa Java para crear un navegador, se combinan las características del lenguaje con protecciones de sentido común aplicadas al propio navegador.

Más allá de la portabilidad básica por ser de arquitectura independiente, Java implementa otros estándares de portabilidad para facilitar el desarrollo. Se beneficia todo lo posible de la tecnología orientada a objetos. (9)

C#:

C# (siglas en inglés “C Sharp” y en español “C Almohadilla”) es un lenguaje de propósito general diseñado por Microsoft para su plataforma .NET. C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo. (10)

La realidad es que hoy día C# es más fuerte en plataformas basadas en Windows, mientras que Java es más fuerte en una gran diversidad de plataformas menos populares (celulares, tarjetas inteligentes, etc.)

1.3.2. Servidor de Base de Datos:

PostgreSQL 8.4:

PostgreSQL es un sistema de base de datos objeto-relacional (ORDBMS), basado en POSTGRES, Versión 4.2, desarrollado en la Universidad de California en Berkeley Departamento de Informática. POSTGRES fue pionero en muchos conceptos que sólo estuvo disponible en algunos sistemas de bases de datos comerciales mucho más tarde.

PostgreSQL es un descendiente de código abierto del código original de Berkeley. Es compatible con una gran parte del estándar SQL y ofrece muchas características modernas:

- consultas complejas
- las claves externas
- dispara
- puntos de vista
- integridad de las transacciones
- multiversión control de concurrencia

Además, PostgreSQL puede ser ampliado por el usuario en muchos aspectos, por ejemplo mediante la adición de nuevos

- los tipos de datos
- funciones
- los operadores
- las funciones de agregado

- métodos de índice
- lenguas de procedimiento

Y a causa de la licencia liberal, PostgreSQL puede ser utilizado, modificado y distribuido por cualquiera de forma gratuita para cualquier propósito, ya sea privado, comercial o académico. (11)

MySQL 5.0:

Es un sistema de gestión de bases de datos relacional, creada por la empresa sueca MySQL AB. MySQL.

La siguiente lista describe algunas de las características más importantes del software de base de datos MySQL.

- Interioridades y portabilidad
- Escrito en C y en C++
- Probado con un amplio rango de compiladores diferentes
- APIs disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl.
- Uso completo de multi-threaded mediante threads del kernel. Pueden usarse fácilmente multiple CPUs si están disponibles
- Proporciona sistemas de almacenamiento transaccionales y no transaccionales
- Relativamente sencillo de añadir otro sistema de almacenamiento. Esto es útil si desea añadir una interfaz SQL para una base de datos propia
- Un sistema de reserva de memoria muy rápido basado en threads
- Joins muy rápidos usando un multi-join de un paso optimizado
- Tablas hash en memoria, que son usadas como tablas temporales
- Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible. Normalmente no hay reserva de memoria tras toda la inicialización para consultas. (12)

1.3.3. Servidor de Aplicación:

Apache Tomcat (también llamado Jakarta Tomcat) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP).

Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software Licence. Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad.

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java. (13)

1.3.4. Framework de Acceso a Datos:

Hibernate es una herramienta de Mapeo Objeto-Relacional (ORM) para la plataforma Java, consiste en la técnica de realizar la transición de una representación de los datos de un modelo relacional a un modelo orientado a objetos y viceversa, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones. Hibernate es software libre, distribuido bajo los términos de la licencia GNU LGPL.

Hibernate no solo realiza esta transformación sino que nos proporciona capacidades para la obtención y almacenamiento de datos de la base de datos que nos reducen el tiempo de desarrollo.

Hibernate busca solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en una aplicación: el usado en la memoria de la computadora (orientación a objetos) y el usado en las bases de datos (modelo relacional). Para lograr esto permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen. Con esta información Hibernate le permite a la aplicación manipular los datos de la base operando sobre objetos, con todas las características de la POO. Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL. Hibernate genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución.

Hibernate está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente. También tiene la funcionalidad de crear la base de datos a partir de la información disponible.

Hibernate para Java puede ser utilizado en aplicaciones Java independientes o en aplicaciones Java EE, mediante el componente Hibernate Annotations que implementa el estándar JPA, que es parte de esta plataforma.(14)

1.3.5. Plataforma de Desarrollo:

Java 2 Platform Enterprise Edition (J2EE):

En la plataforma Java 2, los componentes web proporcionan la capacidad para extender la dinámica de un servidor web. Los componentes web son los servlets Java o bien, las páginas JSP, o los puntos finales de servicios web. La interacción entre un cliente web y una aplicación web se ilustra en la Figura 1. El cliente envía una petición HTTP al servidor web. Un servidor web que implementa la tecnología Java Servet y JavaServer Pages convierte la solicitud en un HTTPServletRequest objeto. Este objeto se entrega a un componente de web, que pueden interactuar con los componentes JavaBeans o una base de datos para generar contenido dinámico. El componente web puede generar un HTTPServletResponse o puede pasar la petición a otro componente web. Finalmente, un componente web genera un HTTPServletResponse objeto. El servidor web se convierte en objeto de una respuesta HTTP y lo devuelve al cliente.

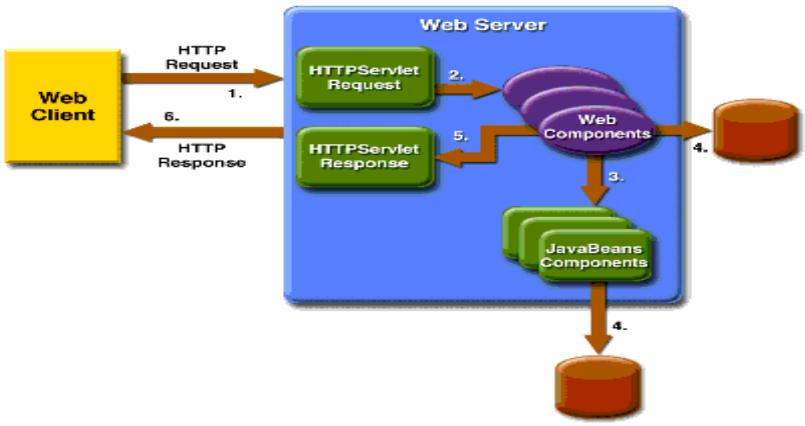

Figura 1: Aplicación Java Web Tratamiento de la Petición.

Java 2 Platform Enterprise Edition (J2EE) define el estándar para el desarrollo de aplicaciones empresariales de varios niveles. La plataforma J2EE simplifica las aplicaciones empresariales basándolas en componentes estandarizados y modulares, proporcionando un conjunto completo de servicios a esos componentes, y manejando muchos detalles de comportamiento de las aplicaciones de forma automática, sin necesidad de programación compleja. La plataforma J2EE añade soporte completo para componentes Enterprise JavaBeans, Java Servlets API, JavaServer Pages y la tecnología XML.

En el cliente de nivel, J2EE es compatible con HTML, así como Java Applets o aplicaciones. Se basa en JavaServer Pages y Servlet de código para crear los datos con formato HTML o de otro tipo para el cliente. Su estructura está basada en J2SE y un conjunto de sus APIs, a las que J2EE les aporta un conjunto de componentes, contenedores (containers) y las APIs para los servicios de transacciones, mensajería, servicio de correo, y conectores de recursos externos. (15).

Plataforma .NET:

.NET es un modelo de programación que soporta múltiples idiomas, ofrece servicios como la gestión del ciclo de vida, se basa en los principios de diseño orientado a objetos.

J2EE y .NET tienen conceptos de máquinas virtuales y justo a tiempo de compilación, J2EE tiene una máquina virtual Java (JVM) que facilita la elección de la plataforma y las instalaciones .NET a Common Language Runtime (CLR) que facilita la elección de los lenguajes de programación, el marco conceptual y la arquitectura. (16)

1.3.6. Tecnología del Lado del Servidor:

Spring Framework:

Spring Framework (también conocido simplemente como Spring) es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java, aunque se puede decir que también hay una versión para la plataforma .NET, Spring .NET.

A pesar de que Spring Framework no obliga a usar un modelo de programación en particular, se ha popularizado en la comunidad de programadores en Java al ser considerado como una alternativa y sustituto del modelo de Enterprise JavaBean. Por su diseño el framework ofrece mucha libertad a los

desarrolladores en Java y soluciones muy bien documentadas y fáciles de usar para las prácticas comunes en la industria.

Mientras que las características fundamentales de este framework pueden emplearse en cualquier aplicación hecha en Java, existen muchas extensiones y mejoras para construir aplicaciones basadas en web por encima de la plataforma empresarial de Java (Java Enterprise Platform).

En una aplicación web, basta con un contenedor de servlets como Tomcat o Jetty. Spring no solo se puede usar para crear aplicaciones web, se puede usar para cualquier aplicación java, aunque su uso habitual sea en entornos web, nada impide utilizarlo para cualquier tipo de aplicación. (17)

Spring tiene 20 módulos colocados en grupos, estos grupos se muestran en la siguiente imagen:

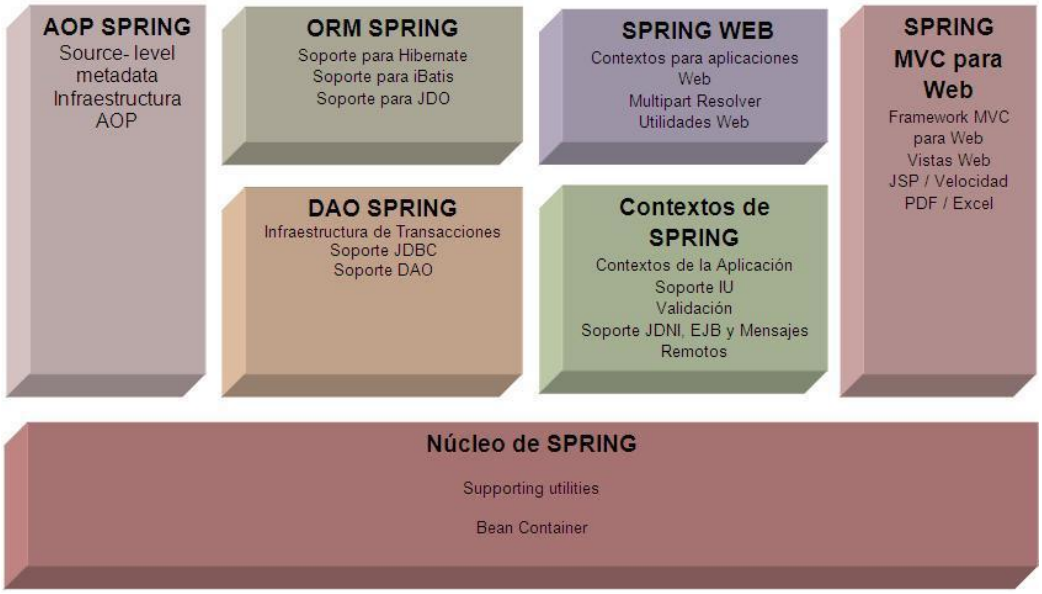

Figura 2: Spring Framework Runtime.

Una descripción breve sobre cada una de las partes en las que se divide este Framework es la siguiente:

- ✓ **APPLICATION CONTEXT:** Encargado de extender ciertas funcionalidades ofrecidas por el Core (internacionalización, eventos de ciclo de vida de la aplicación y validación). Aporta servicios empresariales como e-mail, acceso JNDI, integración con EJB, remoting y scheduling.

- ✓ **AOP:** Módulo basado en la API AOP (programación orientada a aspectos), de tal forma que pretende independizar conceptos presentes a lo largo de toda la aplicación en módulos diferenciados facilitando el desarrollo.
- ✓ **DAO:** Módulo que crea una capa de abstracción para trabajar con JDBC manteniendo el código de acceso a datos claro y simple. Aporta una jerarquía de excepciones para evitar examinar los códigos de error de cada fabricante de BBDD para interpretar el error. Usa el módulo AOP para la gestión de transacciones de los objetos en la aplicación.
- ✓ **ORM:** Módulo que permite la integración con motores de persistencia como Hibernate, iBatis y JPA.
- ✓ **WEB:** Módulo que proporciona características de integración orientadas a la Web como subida de ficheros, mapeo de parámetros del formulario en objetos de negocio, etc. Es el módulo encargado de integrarse con Struts.
- ✓ **MVC:** Módulo que incorpora un Framework Modelo-Vista-Controlador para construir aplicaciones Web. Pese a que puede ser integrado con Struts, el patrón hace uso de IoC (Inversión de Control) para separar la lógica de control de los objetos de negocio.

1.3.7. Tecnología del Lado del Cliente:

JQuery no es más que un conjunto de librerías JavaScript que permite simplificar la manera de interactuar con los documentos HTML, permitiendo manejar eventos, desarrollar animaciones, y agregar interacción con la tecnología AJAX a nuestras páginas web. Está diseñada para cambiar la forma de escribir el código JavaScript escribiendo menos y haciendo más.

Beneficios:

- Reducción de código.
- Reutilización de código.
- Código más conciso y de fácil lectura.
- Facilidad en manejo de arrays.

- Facilidad para operaciones Ajax.
- Compatibilidad Cross browser (multi-navegador)
- Fácil manejo de eventos y animaciones

1.3.8. IDE de Desarrollo:

Eclipse:

Eclipse 3.4 es una plataforma de programación diseñada para la integración de IDEs, presenta una arquitectura abierta y basada en plugins, lo que le permite integrar diversos lenguajes sobre un mismo IDE e introducir otras aplicaciones accesorias. Soporta a los proyectos durante todo el ciclo de vida de desarrollo e incluye soporte de modelado. Eclipse es soportado en los principales sistemas operativos: Linux, Windows, Solaris 8, entre otros. (18)

1.3.9. Metodología de Desarrollo de Software:

Proceso Unificado de Desarrollo (Rational Unified Process, RUP):

El Proceso Unificado de Desarrollo, es un marco de trabajo extensible. RUP es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado (UML), constituye una metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Posee una forma disciplinada de asignar tareas y responsabilidades que definen quién hace qué, cuándo y cómo. Su objetivo es garantizar que la producción de software sea de alta calidad y que satisfaga las necesidades de sus usuarios finales, dentro de un calendario y con el presupuesto previsible. Se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. Se utiliza como **Lenguaje de Modelado UML** (Unified Modeling Language) con el que se puede especificar, construir, visualizar y documentar los artefactos de un sistema de software orientado a objetos (OO). Un artefacto es una información que es utilizada o producida mediante un proceso de desarrollo de software.

RUP establece actividades y criterios que conducen a un sistema desde su máximo nivel de abstracción (la idea en la cabeza del cliente), hasta su nivel más concreto (un programa ejecutándose en las instalaciones del cliente). UML ofrece la notación gráfica necesaria a RUP para representar los sucesivos modelos que se obtienen en este proceso. (19) (20)

Programación Extrema (Extreme Programming, XP):

La Programación Extrema o Extreme Programming (XP) es un enfoque de la ingeniería de software. Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Esta se puede aplicar de manera dinámica durante el ciclo de vida del software. Los valores originales de la programación extrema son: simplicidad, comunicación, retroalimentación (feedback) y coraje. Un quinto valor, respeto, fue añadido en la segunda edición de Extreme Programming Explained. (21)

1.3.10. Herramienta Case:

Visual Paradigm:

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. 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.

Rational Rose:

Es una herramienta basada en el Lenguaje Unificado de Modelación (UML), que permite crear los diagramas que se van generando durante el proceso de Ingeniería en el Desarrollo del Software.

Rational permite completar una gran parte de las disciplinas (flujos fundamentales) de RUP tales como:

- ✓ Captura de requisitos (parcialmente).
- ✓ Análisis y diseño (completamente).
- ✓ Implementación (como ayuda).
- ✓ Control de cambios y gestión de configuración (parcialmente).

1.3.11. Herramienta SVN:

Subversion (SVN**)** es una herramienta de control de versiones Open Source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Utiliza el concepto de revisión para guardar los cambios producidos en el repositorio. Entre dos revisiones sólo guarda el conjunto de modificaciones (delta), optimizando así al máximo el uso de espacio en disco.

SVN permite al usuario crear, copiar y borrar carpetas con la misma flexibilidad con la que lo haría si estuviese en su disco duro local. Dada su flexibilidad, es necesaria la aplicación de buenas prácticas para llevar a cabo una correcta gestión de las versiones del software generado. El objetivo de este artículo es guiar al desarrollador para que sea capaz de tomar la mejor decisión en cada etapa del ciclo de vida de su proyecto.

1.3.12. Herramienta para generar PDF:

iReport 4.0 es un diseñador visual de código libre para JasperReports escrito en Java. Es un programa que ayuda a los usuarios y desarrolladores que usan la librería JasperReports para diseñar reportes visualmente. A través de una interfaz agradable y simple de usar, iReport provee las funciones más importantes para crear reportes amenos en poco tiempo. iReport puede ayudar a la gente que no conoce la sintaxis XML para generar reportes de JasperReports.

Funcionamiento:

iReport provee a los usuarios de JasperReports una interfaz visual para construir reportes, generar archivos “jasper” y “print” de prueba. iReport nació como una herramienta de desarrollo, pero puede utilizarse como una herramienta de oficina para adquirir datos almacenados en una base de datos, sin pasar a través de alguna otra aplicación

Las características más relevantes de iReport son:

- 100% escrito en Java y además Open Source y gratuito.
- Maneja el 98% de las etiquetas de JasperReports

- Permite diseñar con sus propias herramientas: rectángulos, líneas, elipses, campos de los textfields, cartas, y subreportes.
- Soporta internacionalización nativamente.
- Browser de la estructura del documento.
- Recopilador y exportador integrados.
- Soporta JDBC.
- Soporta JavaBeans como orígenes de datos (éstos deben implementar la interfaz JRDataSource).
- Incluye Wizard's (asistentes) para crear automáticamente informes.
- Tiene asistentes para generar los subreportes.
- Tiene asistentes para las plantillas.
- Facilidad de instalación.

Para generar documentos PDF desde código Java utilizaremos una librería llamada JasperReports, con esta librería podemos crear documentos PDF con texto, tablas, gráficos, crear conexiones con base de datos para los reportes, etc

JasperReports es la mejor herramienta de código libre en Java para generar reportes. Puede entregar ricas presentaciones o diseños en la pantalla, para la impresora o para archivos en formato PDF, HTML, RTF, XLS, CSV y XML. Está completamente escrita en Java y se puede utilizar en una gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones Web, para generar contenido dinámico. (22)

1.3.13. Mecanismo de Seguridad:

Spring Security es un subproyecto del framework Spring, que permite gestionar completamente la seguridad de nuestras aplicaciones Java, y cuyas ventajas principales son las siguientes:

- Es capaz de gestionar seguridad en varios niveles: URLs que se solicitan al servidor, acceso a métodos y clases Java, y acceso a instancias concretas de las clases.

- Permite separar la lógica de nuestras aplicaciones del control de la seguridad, utilizando filtros para las peticiones al servidor de aplicaciones o aspectos para la seguridad en clases y métodos.
- La configuración de la seguridad es portable de un servidor a otro, ya que se encuentra dentro del WAR o el EAR de nuestras aplicaciones.
- Soporta muchos modelos de identificación de los usuarios (HTTP BASIC, HTTP Digest, basada en formulario, LDAP, OpenID, JAAS y muchos más). Además podemos ampliar estos mecanismos implementando nuestras propias clases que extiendan el modelo de Spring Security.
- Spring Security nos ofrece servicios de identificación y acceso a los recursos de nuestras aplicaciones de una forma potente y sencilla, sin tener que escribir ni una sola línea de código.
- Podemos asignar los permisos directamente a los usuarios o a grupos de usuarios, o incluso mezclar ambas formas.
- Podemos añadir fácilmente seguridad en nuestras aplicaciones existentes o modificar la seguridad de una aplicación rápidamente, ya que Spring Security separa claramente la seguridad de una aplicación de su lógica. (23)

1.4. Conclusiones del Capítulo:

Durante este capítulo se realizó el estudio sobre las principales herramientas a utilizar en el desarrollo del trabajo de diploma. Como lenguaje de programación se propone Java, además, con el fin de permitir la gestión de diferentes proveedores, así como los permisos asignados a cada uno de ellos se propone como servidor de base de datos a Postgres y como servidor de aplicación Apache Tomcat. Como framework de acceso a datos se propone Hibernate con su estándar JPA ya que es una herramienta de Mapeo Objeto Relacional para la plataforma Java. Con el fin de desarrollar una aplicación potente y eficiente, la plataforma de desarrollo seleccionada fue Java2EE. Como tecnología del lado del servidor se propone Spring Framework y del lado del cliente JQuery. Como IDE de desarrollo se seleccionó a Eclipse y a RUP como metodología de desarrollo de software con el fin de asegurar un software con calidad, utilizando como lenguaje de modelado UML pues ofrece la notación gráfica a RUP para representar los sucesivos modelos que se obtienen en este proceso. Como herramienta case debido a la gran plataforma

de seguridad que ofrecen a los sistemas que las usan se seleccionó Visual Paradigm, la cual permite una rápida construcción de aplicaciones con eficacia y rapidez para aspectos claves de todo el proceso de desarrollo del sistema. Como herramienta SVN se escogió Subversión porque recuerda los cambios que se le hayan hecho a sus ficheros y directorios permitiendo así recuperar versiones antiguas de sus datos. Para diseñar reportes visualmente se utilizó iReport pues provee las funciones más importantes para crear reportes amenos en poco tiempo haciendo uso de la librería JasperReports. Con el fin de gestionar completamente la seguridad de la aplicación se utilizó como mecanismo de seguridad a Spring Security. Esta gama de componentes que incluyen todas o la mayoría de los requisitos necesarios para el desarrollo de los sistemas, fueron seleccionadas con exactitud en torno a las necesidades de los desarrolladores de sistemas para la automatización de procesos incluyendo el análisis, diseño e implementación.

Capítulo 2: Características del Sistema

Introducción:

En el presente capítulo se plantean los objetivos estratégicos de Cubacel así como los procesos que serán objetos de automatización. Además se describen todos los conceptos del dominio relacionados con la propuesta, así como los requisitos funcionales y no funcionales. Se realiza una descripción detallada de los casos de uso que guiarán el desarrollo del software, así como de los actores del sistema. Se describe la arquitectura base del sistema a partir de las tecnologías seleccionadas para la construcción del mismo. Se modelan y describen los diagramas que representan las funcionalidades del sistema, aplicando los patrones de arquitectura y diseño seleccionados.

2.1 Objetivos Estratégicos de la Organización:

Cubacel provee de soluciones y servicios a sus clientes dentro del mundo de la telefonía móvil, prestando especial atención al empleo de las nuevas tecnologías de acceso a contenidos, nuevos modelos de dispositivos móviles, así como nuevos formatos multimedia. Constantemente actualiza su infraestructura tecnológica garantizando capacidad inmediata de respuesta a los distintitos modelos de negocios que puedan presentar operadores, clientes y proveedores de contenidos. Actualmente el principal objetivo de la entidad es adentrarse en el mercado de las aplicaciones y servicios para dispositivos móviles.

2.2 Objeto de Automatización:

Serán objeto de automatización en esta aplicación los procesos de gestión de contenidos en el que tiene lugar el servicio de buscar, además de la gestión de reporte y facturación los cuales van a permitir la generación de pdf, imprimir, así como mostrar gráfico de porcentaje.

2.3 Modelo de Dominio:

En el Modelo de Dominio se muestran los principales conceptos a utilizar en el desarrollo de la aplicación. Esto ayuda a los usuarios, clientes, desarrolladores e interesados en general, a utilizar un vocabulario

común para poder entender el contexto en que se ubica el sistema, para capturar correctamente los requisitos y poder construir una aplicación robusta que brinde los beneficios que de ella se esperan.

La figura muestra los principales conceptos y sus relaciones, presentes en el modelo de dominio.

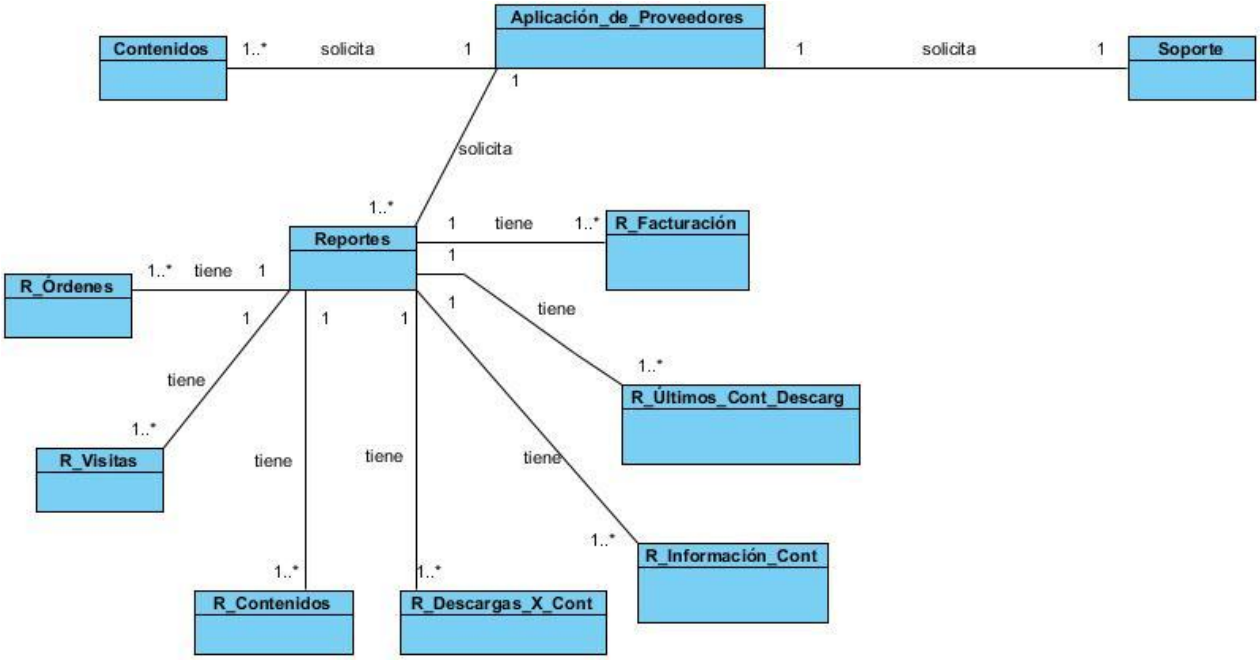

Figura 3: Modelo de Dominio.

2.3.1 Conceptos del Modelo de Dominio:

A continuación se realiza una descripción de los conceptos representados en el Modelo del Dominio, los cuales simbolizan objetos del mundo real que se encuentran dentro del dominio del negocio, con la necesidad de comprender las relaciones que se establecen entre las diferentes clases.

Aplicación de Proveedores: Representa el subsistema encargado de la comunicación entre los proveedores y la plataforma.

Contenidos: Representa los contenidos que serán generados.

Reportes: Representa los reportes que serán generados.

Soporte: Representa la comunicación entre los proveedores y los administradores a través de preguntas y respuestas.

R_Órdenes: Representa los reportes de los diferentes tipos de órdenes.

R_Visitas: Representa los reportes de las visitas realizadas a los diferentes contenidos solicitados por los proveedores.

R_Contenidos: Representa los reportes de los diferentes tipos de contenidos.

R_Descargas_x_Cont: Representa los reportes de la cantidad de descargas que se le han realizado a los diferentes contenidos que fueron solicitados por los proveedores.

R_Información_Cont: Representa los reportes sobre los diferentes contenidos que fueron solicitados por los proveedores.

R_Últimos_Cont_Descarg: Representa los reportes de los últimos contenidos más descargados de los diferentes contenidos que fueron solicitados por los proveedores.

R_Facturación: Representa los reportes de la facturación de los contenidos que fueron solicitados por los proveedores.

2.4 Descripción del Sistema a Desarrollar:

El sistema a desarrollar tiene como objetivo fundamental proporcionar una aplicación de administración que permita a los proveedores de la plataforma la gestión de sus contenidos, ya sea imágenes, videos, música, juegos, temas de dispositivos móviles y sus derechos digitales, así como la generación de reportes y servicio de soporte, todo ello accesible desde una interfaz de usuario web. Debe permitir el acceso a las personas autorizadas desde cualquier sitio físico, posibilitar el acceso a información estadística y de facturación tanto histórica como en tiempo real, garantizando la seguridad de la información que maneja.

2.4.1 Requerimientos del Sistema:

La IEEE Standard Glossary of Software Engineering Terminology define un requerimiento como la condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente. (24)

Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos. Los requisitos se pueden clasificar en: funcionales y no funcionales.

Requisitos funcionales:

Son capacidades o condiciones que el sistema debe cumplir. Los requisitos funcionales se mantienen invariables sin importar con que propiedades o cualidades se relacionen.

RF1 Autenticar Usuario:

Permitirá autenticar a los proveedores registrados.

- Usuario.
- Contraseña.

RF2 Adicionar Contenido:

Permitirá adicionar un nuevo contenido al sistema.

- Nombre del contenido.
- Categoría.
- Subcategoría.
- Lista de celulares compatibles con el contenido.
- Imagen (logo).
- Descripción.
- Especificación de derechos digitales.
- Precio del contenido.

RF3 Modificar Contenido:

Permitirá modificar un contenido existente en el sistema.

- Categoría.
- Nombre del contenido.
- Subcategorías.

RF4 Eliminar Contenido:

Permitirá eliminar un contenido existente en el sistema.

- Categoría.
- Nombre del contenido.
- Subcategorías.

RF5 Buscar Contenido:

Permitirá buscar contenidos por diferentes parámetros.

- Categoría.
- Nombre del contenido.
- Subcategorías.

RF6 Órdenes Fallidas:

Permitirá reportar los contenidos que han sido interrumpidos o cuando al usuario no le haya llegado una notificación de la descarga que realizó.

- Rango de Fechas.

RF7 Órdenes Exitosas:

Permitirá reportar los contenidos que han sido exitosos.

- Rango de Fechas.

RF8 Órdenes Canceladas:

Permitirá reportar los contenidos que han sido cancelados, estos son aquellos que los usuarios han cancelado su descarga o los cancelados durante la descarga del contenido.

- Rango de Fechas.

RF9 Libro de Visitas:

Permitirá buscar la cantidad de visitas que se ha realizado a un determinado contenido.

- Rango de Fechas.
- Nombre del contenido.

RF10 Contenidos Más Descargados:

Permitirá buscar los contenidos más descargados.

- Rango de Fechas.

RF11 Contenidos más Descargados por Categoría:

Permitirá buscar los contenidos más descargados atendiendo a una categoría específica.

- Rango de Fechas.
- Categoría

RF12 Cantidad de Descarga por Contenido:

Permitirá mostrar la cantidad de veces que se ha descargado un contenido determinado, así como otros datos.

- Nombre del contenido.

RF13 Últimos Contenidos Descargados:

Permitirá mostrar cierta cantidad de los últimos contenidos descargados atendiendo a la petición del usuario.

- Cantidad de elementos.

RF14 Facturación:

Permitirá facturar los reportes de los usuarios a partir de los contenidos descargados.

- Rango de Fechas.

RF15 Soporte:

Permitirá a los proveedores y administradores comunicarse mediante una vía de preguntas y respuestas.

RF16 Generar PDF:

Permitirá generar gráficos PDF con el fin de documentar los datos requeridos.

Requisitos no funcionales:

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

Existen múltiples categorías para clasificar a los requisitos no funcionales, las mismas son las siguientes:

- ✓ Restricciones en el diseño y la implementación.
- ✓ Requisitos de apariencia o interfaz externa.
- ✓ Requisitos de Seguridad.
- ✓ Requisitos de Usabilidad.
- ✓ Requisitos de Soporte.

La especificación de los requisitos debe ser precisa, completa y clara, además de especificada por escrito como todo contrato o acuerdo entre dos partes.

1. **Usabilidad:**

1.1 Formato de Fecha utilizadas en el sistema:

En la interfaz visual se utilizará un formato de Fecha, para efectos de almacenamiento se utilizará el formato estándar según el SGBD. El formato disponible para la interfaz visual es:

Month/Day/Year Ej. 05/10/2012.

1.2 Mensajes de Textos en la Interfaz:

Tanto los mensajes para interactuar con los usuarios, así como los mensajes de error, deberán ser en idioma castellano y tener una apariencia estándar. Los mensajes de error deberán ser lo suficientemente informativos para dar a conocer la severidad del error. Los colores utilizados deben ajustarse a los estándares definidos por ETECSA.

1.3 Permitir listas de valores en la interfaz:

El sistema deberá facilitar la entrada de datos a los usuarios, presentando listas de valores que permitan escoger datos descriptivos y no aislados.

1.4 Entrenamiento y navegación de los usuarios en el sistema:

La Aplicación de Administración de Proveedores está diseñada para un público objetivo, por lo que la información que se maneja es conocimiento de todos los usuarios que accederán a la interfaz. Gracias a esto, no es necesario un entrenamiento exhaustivo, además se utilizará un lenguaje claro y sencillo.

El usuario deberá saber en todo momento en que parte del sitio se encuentra navegando, permitiéndole acceder siempre a la opción de retornar a la página principal.

2. Fiabilidad:

2.1 Tiempo de disponibilidad del sistema:

El sistema debe estar disponible 24 horas los 7 días de la semana, dependiendo esto de factores externos al sistema.

3. Eficiencia:

3.1 Tiempo de respuesta por transacciones:

Los tiempos de respuesta serán de 15 segundos.

3.2 Rendimiento:

El 95% de las acciones deben de realizarse en menos de 15 segundos.

4. Soporte:

La UCI debe brindar soporte tanto a administradores como a clientes que necesiten un breve entrenamiento para operar el sistema, en dependencia de las demandas del comprador, puede ser mantenimiento, entrenamiento de personal, integración con otro software u operar directamente el sistema. Se cuenta con un sistema de soporte mediante el cual los usuarios pueden obtener toda la

información que necesitan de manera instantánea contactando a través de un sistema de chat con los administradores o personal encargado.

5. **Restricciones de Diseño:**

La aplicación contará con un diseño en el cuál el logo va a ser el especificado por ETECSA, en este caso es CUBACELWAP, además será especificado el tipo de aplicación, en este caso es la Aplicación de Administración de Proveedores: CUBACELWAP PROVEEDORES.

6. **Seguridad**

La información privada contenida en la Plataforma de Gestión de Contenidos para Dispositivos Móviles y manejada por los usuarios debe ser protegida de acceso no autorizado haciendo uso de mecanismos de autenticación como el uso de usuario y contraseña. Para que se realice cualquier operación el usuario debe estar previamente registrado.

El sistema en caso de ocurrir algún error debe recuperarse si es posible, realizando un correcto manejo de las excepciones y registrando en trazas los eventos que ocasionan funcionamientos indebidos para un posterior análisis.

2.5 Modelo del Sistema:

Una vez identificados y clasificados los requerimientos del sistema en necesario identificar los actores y caso de usos. Los casos de usos son priorizados y detallados, realizando la descripción de cada uno ellos.

2.5.1 Actores del Sistema:

Actor del sistema	Descripción
Proveedor	Es el encargado de gestionar el contenido

Tabla 1: Actores del Sistema.

2.5.2 Diagrama de Casos de Uso:

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar. Su ventaja principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente. (25).

Una vez descritos los actores se realizan el proceso de identificación de los casos de usos apoyados en los requisitos del sistema.

En la siguiente figura se muestra el diagrama de casos de uso del sistema para representar el flujo de eventos definidos por el actor.

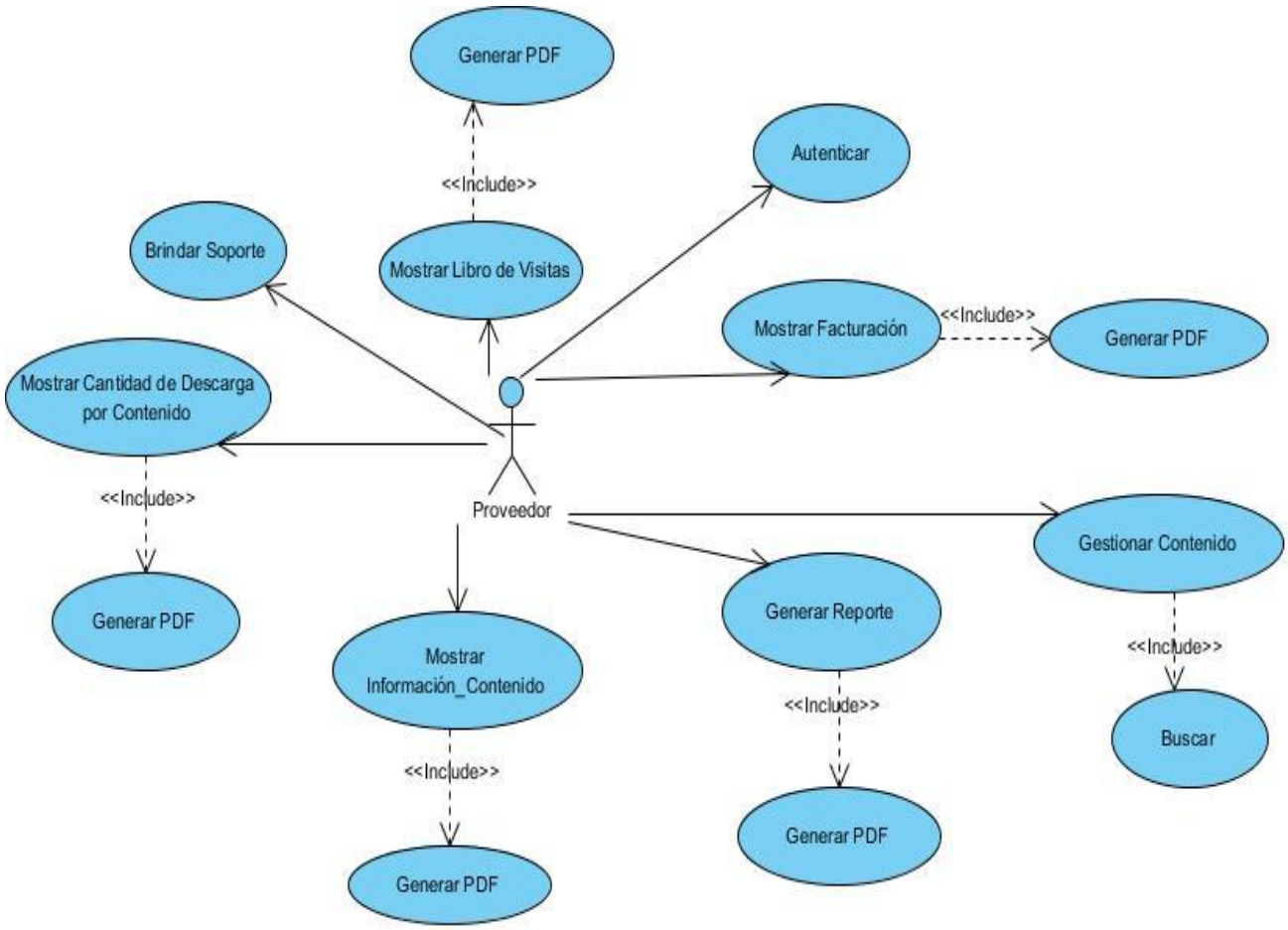

Figura 4: Diagrama de Caso de Uso del Sistema.

2.5.3 Casos de Uso del Sistema:

A continuación se expone la descripción del caso de uso Autenticar Usuario para lograr entender los procesos globales de la aplicación.

CU Autenticar Usuario.

Objetivo	Acceder a la sección de proveedor para realizar las funcionalidades a las que tiene permiso.
Actor	Proveedor.
Resumen	El proveedor entra los datos correspondientes, una vez dentro este podrá realizar los cambios correspondientes y mantener actualizados sus contenidos.
Complejidad	Baja.
Prioridad	Crítico.
Precondiciones	El proveedor debe estar registrado en la base de datos.
Referencias:	RF1.
Flujo de eventos	
Flujo básico Autenticar Usuario	
Actor	**Sistema**
1. Entra los datos correspondientes: • Usuario. • Contraseña.	
	2. Se conecta con la base de datos y valida que los datos sean correctos.
	3. Muestra la página principal.

Flujos alternos	
Actor	**Sistema**
	2.1 Muestra un mensaje (“Credenciales incorrectas, intente de Nuevo”).
Poscondiciones	

Tabla 2: Autenticar Usuario.

2.6 Conclusiones del Capítulo:

En este capítulo se definió el objeto de estudio de la organización, así como el objeto de automatización. Se realizó una descripción del sistema a desarrollar donde se definieron los requisitos funcionales y las cualidades que el producto debe tener, recogidas en los requisitos no funcionales. Se identificaron y describieron los actores del sistema y casos de usos estableciendo su relación en el diagrama de caso de uso.

Capítulo 3: Diseño del Sistema

Introducción:

El objetivo principal de este capítulo es realizar la descripción de los procesos que se llevan a cabo en la disciplina de diseño, obteniendo como resultado los artefactos más importantes para modelar el sistema, teniendo en cuenta los patrones de diseño que aportan soluciones concretas a problemas específicos, para lograr un diseño eficaz en el software orientado a objetos.

3.1 Descripción de la Arquitectura:

En los inicios de la informática, la programación se consideraba un arte y se desarrollaba como tal, con el tiempo se han ido descubriendo y desarrollando formas y guías generales en base a las cuales se puedan resolver los problemas. A estas, se les ha denominado Arquitectura de Software, porque, a semejanza de los planos de un edificio o construcción, estas indican la estructura, funcionamiento e interacción entre las partes del software. La Arquitectura de Software tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad. (26) (27)

3.1.1 Estilos Arquitectónicos:

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante composición de los estilos fundamentales. En tal sentido, los estilos conjugan elementos o “componentes” como son los conectores, configuraciones y restricciones. Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que restringe los roles/rasgos de los elementos arquitectónicos y las relaciones permitidas entre esos elementos dentro de la arquitectura que se conforma a ese estilo. (28)

La descripción de un estilo se puede formular en lenguaje natural o en diagramas, pero lo mejor es hacerlo en un lenguaje de descripción arquitectónica o en lenguajes formales de especificación.

Entre los estilos arquitectónicos más difundidos y usados se encuentran:

- ✓ **Sistemas de flujo de datos:**

- Secuencial en lote.
- Tubos y filtros.

- ✓ **Sistemas centrados en los datos:**

- Arquitecturas basadas en pizarras o repositorios.

- ✓ **Sistemas de llamada y retorno:**

- Modelo – Vista – Controlador (MVC).
- Arquitecturas en Capas.
- Arquitecturas Orientadas a Objetos.
- Arquitecturas Basadas en Componentes.

- ✓ **Peer to Peer:**

- Procesos comunicativos.
- Arquitecturas Basadas en Eventos.
- Arquitecturas Orientadas a Servicios.
- Arquitecturas Basadas en Recursos.

- ✓ **Máquinas virtuales:**

- Intérpretes.
- Sistemas basados en reglas.

3.1.2 Arquitectura Cliente/Servidor:

La arquitectura Cliente/Servidor agrupa conjuntos de elementos que efectúan procesos distribuidos y cómputo cooperativo.

Cliente:

Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

Servidor:

Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LAN o WAN, para proveer de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc.

Características del Modelo Cliente/Servidor:

En el modelo Cliente/Servidor se pueden encontrar las siguientes características:

1. El Cliente y el Servidor pueden actuar como una sola entidad y también pueden actuar como entidades separadas, realizando actividades o tareas independientes.

2. Las funciones del Cliente y el Servidor pueden estar en plataformas separadas, o en la misma plataforma.

3. Un servidor da servicio a múltiples clientes en forma concurrente.

4. Cada plataforma puede ser escalable independientemente. Los cambios realizados en las plataformas de los Clientes o de los Servidores, ya sean por actualización o por reemplazo tecnológico, se realizan de una manera transparente para el usuario final.

5. La interrelación entre el hardware y el software están basados en una infraestructura poderosa, de tal forma que el acceso a los recursos de la red no muestra la complejidad de los diferentes tipos de formatos de datos y de los protocolos.

6. Un sistema de servidores realiza múltiples funciones al mismo tiempo que presenta una imagen de un solo sistema a las estaciones Clientes. Esto se logra combinando los recursos de cómputo que se encuentran físicamente separados en un solo sistema lógico, proporcionando de esta manera el servicio más efectivo para el usuario final. (29)

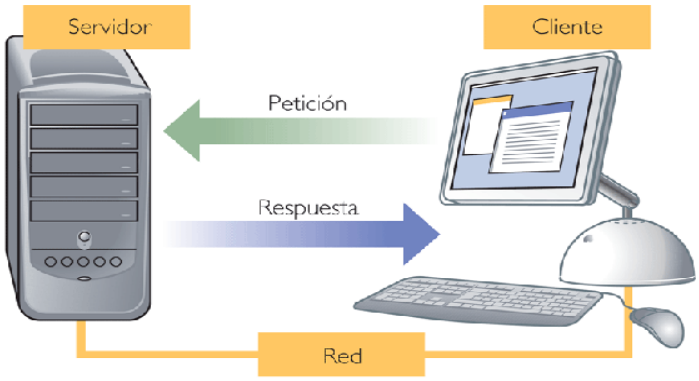

Figura 5: Arquitectura Cliente/Servidor.

3.1.3 Arquitectura en Capas:

La estrategia tradicional de utilizar aplicaciones compactas causa gran cantidad de problemas de integración en sistemas software complejos como pueden ser los sistemas de gestión de una empresa o los sistemas de información integrados consistentes en más de una aplicación. Estas aplicaciones suelen encontrarse con importantes problemas de escalabilidad, disponibilidad, seguridad e integración.

Para solventar estos problemas se ha generalizado la división de las aplicaciones en capas que normalmente serán tres: una capa que servirá para gestionar los datos y el acceso a la base de datos, una capa para centralizar la lógica de negocio (modelo) y por último una interfaz gráfica que facilite al usuario el uso del sistema. (30)

Figura 6: Arquitectura en Capas.

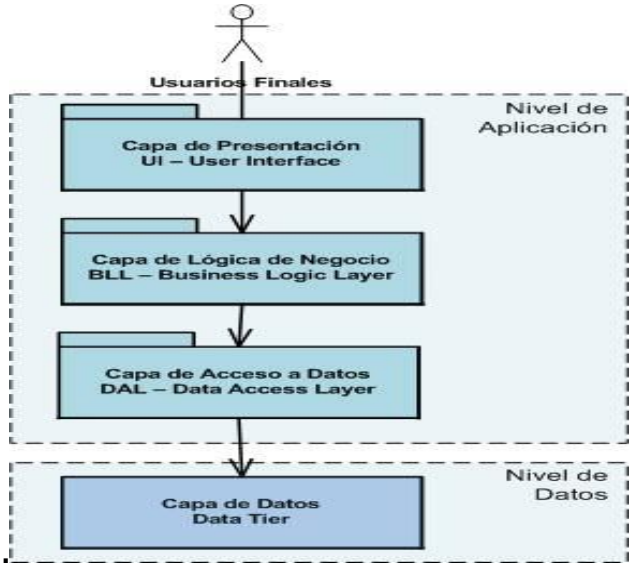

Figura 7: Arquitectura en tres Capas.

Capa de presentación: En esta se encuentran los servicios orientados al usuario, lo que les permite la interacción con el sistema. Esta capa contiene los elementos que garantizan la interacción del usuario con el negocio del sistema.

Capa de negocio: Es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

Capa de acceso a datos: Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Modelo – Vista – Controlador:

Modelo-Vista-Controlador (Model-View-Controller) es un patrón de arquitectura de las aplicaciones software, separa la lógica de negocio de la interfaz de usuario lo que facilita la evolución por separado de ambos aspectos e incrementa reutilización y flexibilidad. (31)

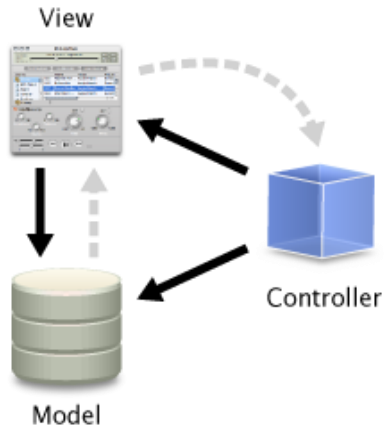

Figura 8: Modelo- Vista-Controlador.

Un modelo puede tener diversas vistas, cada una con su correspondiente controlador. Un ejemplo clásico es el de la información de una base de datos, que se puede presentar de diversas formas: diagrama de pastel, de barras, tabular, etc. A continuación se muestra una explicación de cada componente:

1. El **modelo** es el responsable de:

   - Acceder a la capa de almacenamiento de datos.
   - Define las reglas de negocio (la funcionalidad del sistema).
   - Lleva un registro de las vistas y controladores del sistema.
   - Si estamos ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

2. El **controlador** es responsable de:

   - Recibir los eventos de entrada (un clic, un cambio en un campo de texto, etc.).
   - Contiene reglas de gestión de eventos.

3. Las **vistas** son responsables de:

   - Recibir datos del modelo y las muestras al usuario.
   - Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
   - Pueden dar servicios para que sean invocados por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).

3.1.4 Diseño de la Arquitectura del Sistema:

En el sistema de administración para los proveedores de la plataforma COMCEL, las reglas fundamentales del negocio serán manejadas del lado servidor, dejando en el lado cliente las reglas de presentación y validación de la información. Con este diseño se propone que los usuarios accedan a través de un navegador web a las funcionalidades de un sistema centralizado.

Se propone un diseño por capas, encapsulando y delimitando las responsabilidades de cada una de estas en el sistema. En la siguiente figura se representa la arquitectura del sistema:

Figura 9: Arquitectura del Sistema.

Acceso a Datos: En la capa de accesos a datos del sistema se encuentran los objetos de acceso a datos, DAO, que se encargan de intercambiar información con la capa de negocio. Un DAO encapsula la lógica del acceso a datos separando la petición de los mismos, de la lógica de negocio.

Presentación: Es la encargada de interactuar con el usuario, en la misma se encuentran las vistas y controladores. Las vistas son ficheros JSP[4] generadas con HTML[5] y el TAGLIB[6], estándar de JSP. La capa de presentación interactúa con la capa de negocio a través de la arquitectura que propone Spring MVC[7].

[4] JSP: Java Server Pages.

[5] HTML: Hypertext Markup Language (lenguaje de marcado de hipertexto).

[6] TAGLIB: Bibliotecas desarrolladas para ser integradas y utilizadas en las páginas JSP de una aplicación J2EE.

[7] Spring MVC: SPRING Modelo-Vista-Controlador.

Lógica del Negocio: En esta capa se encuentran los managers que contienen las reglas o lógica de la aplicación. Los managers interactúan con los objetos de acceso a datos para consultar, insertar, modificar o eliminar información. Además de las fachadas, la cual responde a las solicitudes que realizan los controladores. Este responde a eventos, usualmente acciones del usuario, e invoca peticiones a la capa de acceso a datos y, probablemente, a la vista.

Entidades del Negocio o Dominio: Son las entidades que modelan el problema, es la abstracción de la situación real. Se representan de forma vertical ya que todas las capas interactúan con ellas, las consultan, las crean, las modifican y las eliminan.

Seguridad: Están presentes las funcionalidades que garantizan el nivel de acceso de los usuarios al sistema, verificando en cada petición que el usuario tenga permiso para ejecutarla. Para implementar la seguridad se utiliza Spring Security. Se representa de forma vertical ya que todas las capas interactúan con ella.

Excepciones: Significa que todas las solicitudes de un determinado sitio pasarán por un solo punto de entrada. Como resultado, todas las excepciones burbujearán eventualmente hacia arriba hasta el Controller, permitiendo al desarrollador manejarlos en un solo lugar. Se representa de forma vertical ya que todas las capas interactúan con ella.

3.2 Modelo de Diseño:

El modelo de diseño es una realización del diseño del sistema. Es un modelo físico y concreto. Se centra en cómo los requisitos funcionales y no funcionales tienen impacto en el sistema a desarrollar. Dentro de sus propósitos están: crear una entrada apropiada y un punto de partida para la implementación, descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo. Adquirir comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnología de distribución y concurrencia. Esto se representa por colaboraciones en el modelo de diseño y denota realización de caso de uso del diseño. (32)

3.2.1 Diagrama de Clases del Diseño:

Un diagrama de Clases del Diseño es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Una clase es una descripción de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y semántica. (32)

El diagrama general de las clases del diseño representa la interacción entre las clases de las diferentes capas que integran el sistema.

A continuación se muestra el diagrama de clases del diseño del CU Autenticar Usuario.

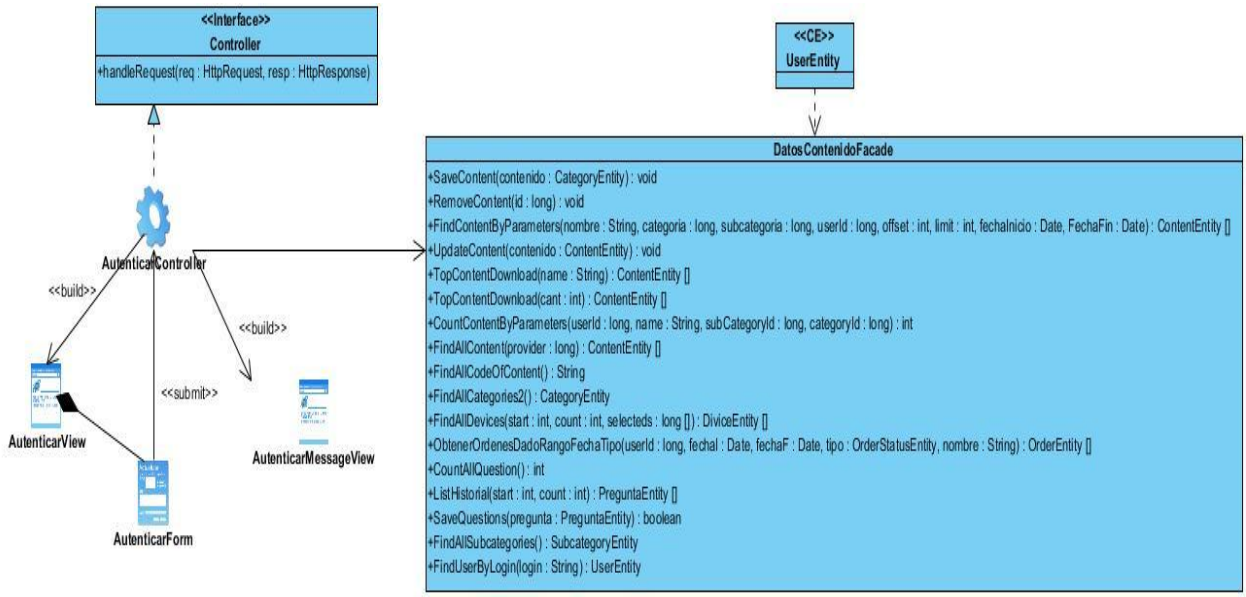

Figura 10: Diagrama de Clases del Diseño del CU Autenticar Usuario.

La creación de los diagramas de interacción requiere la aplicación de los principios para la asignación de responsabilidades y el uso de los principios y patrones de diseño. Es por ello que en el epígrafe siguiente se explican los patrones de diseño utilizados en el desarrollo del sistema.

3.3 Patrones de Diseño:

Los patrones de diseño son un conjunto de prácticas de óptimo diseño que se utilizan para abordar problemas recurrentes en la programación orientada a objetos.
Una arquitectura orientada a objetos bien estructurada está compuesta por varios patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles.

Los patrones de diseño se dividen en tres grupos principales:

- **Creacionales:** solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación.
- **Estructurales:** solucionan problemas de composición (agregación) de clases y objetos.
- **De Comportamiento:** soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

3.3.1 Patrones GRASP:

GRASP es un acrónimo que significa General Responsibility Asignment Software Patterns (patrones generales de software para asignar responsabilidades). Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

Una de las actividades más complicadas en Orientación de Objetos consiste en elegir las clases adecuadas y decidir cómo estas clases deben interactuar. Las responsabilidades están relacionadas con las obligaciones de un objeto en cuanto a su comportamiento.

- **Experto:** La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada.

- **Creador:** El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar la clase responsable de crear una nueva instancia de determinada clase.

- **Controlador:** Un Controlador es un objeto que se encarga de manejar un evento del sistema. Este no realiza mucho trabajo por sí mismo, sino que delega en otros objetos, coordina y controla el trabajo que se necesita hacer.

- **Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y recurre a ellas. El Bajo Acoplamiento soporta un diseño de clases más independientes, que reducen el impacto de cambios, y permite que sean más reutilizables.

- **Alta Cohesión:** La cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos, y auto-identificable. Una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo.

3.3.2 Patrones GoF:

✓ **Patrones creacionales:**

- **Singleton** (instancia única): garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

✓ **Patrones estructurales:**

- **Facade** (Fachada): Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema.

✓ **Patrones de comportamiento:**

- **Iterator** (Iterador): Permite realizar recorridos sobre objetos compuestos independientemente de la implementación de estos.

- ✓ **Patrones de interacción:**

Los patrones de interacción buscan la reutilización de interfaces eficaces y un manejo óptimo de los recursos de las páginas web, haciendo más eficaz el consumo de tiempo en el diseño del sitio web y permitiendo a los programadores novatos adquirir más experiencia.

3.3.3 DAO:

Muchas aplicaciones de la plataforma J2EE en el mundo real necesitan utilizar datos persistentes en algún momento. El Objeto de Acceso a Datos (DAO) implementa el mecanismo de acceso requerido para trabajar con la fuente de datos. Esta fuente de datos puede ser un almacenamiento persistente como una RDMBS, un servicio externo como un intercambio B2B, un repositorio LDAP, o un servicio de negocios al que se accede mediante CORBA Internet Inter-ORB Protocolo (IIOP) o sockets de bajo nivel. (33)

El problema que viene a resolver este patrón es el de contar con diversas fuentes de datos (base de datos, archivos, servicios externos, etc.). Este patrón surge de la necesidad de gestionar una diversidad de fuentes de datos, aunque su uso se extiende al problema de encapsular no sólo la fuente de datos, sino además ocultar la forma de acceder a los datos. Se trata de que el software cliente se centre en los datos que necesita y se olvide de cómo se realiza el acceso a los datos o de cuál es la fuente de almacenamiento. El DAO tiene una interfaz común, sea cual sea el modo y fuente de acceso a datos.

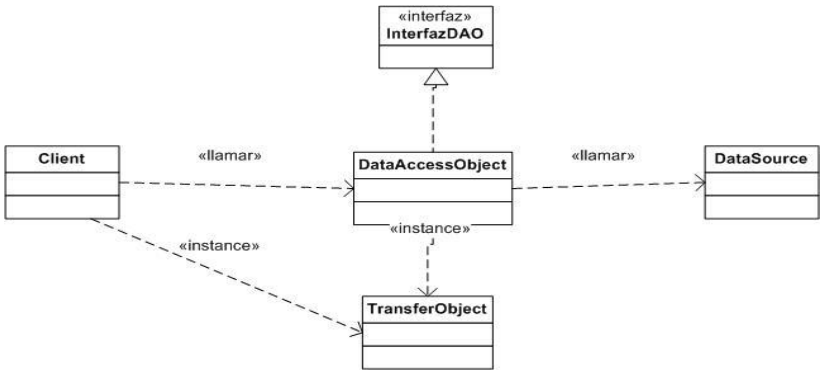

Figura 11: Diagrama de clases que representa las relaciones para el patrón DAO.

3.3.4 Inyección de Dependencia:

Spring se basa en IoC (Principio de Inversión de Dependencia, Inversion of Control) que consiste en:

1. Un contenedor que maneja objetos.
2. El contenedor generalmente controla la creación de estos objetos.
3. El contenedor resuelve dependencias entre los objetos que contiene.

Spring proporciona un contenedor que maneja todo lo que se hace con los objetos del IoC. Debido a la naturaleza del IoC, el contenedor más o menos ha definido el ciclo de vida de los objetos. Y, finalmente, el contenedor resuelve las dependencias entre los servicios que él controla. (34)

Para el desarrollo de la solución se aplicaron diferentes patrones de diseño, con el objetivo de facilitar el mantenimiento del software y contribuir a la realización de un producto reutilizable y escalable.

Entre los diferentes patrones se pueden mencionar los referidos a asignación de responsabilidades GRASP (del inglés: General Responsibility Assignment Software Patterns), estos fueron tomados en cuenta para el diseño de clases y métodos de cada una de ellas. El patrón DAO se utilizó con el fin de contar con diversas fuentes de datos (base de datos, archivos, servicios externos) y crear una capa de abstracción de estos con respecto a la lógica de negocio. La Inyección de Dependencia fue otro patrón utilizado mediante el cual cada una de las clases de la aplicación es independiente comunicándose únicamente a través de un interface y eliminando la responsabilidad de instanciar las referencias a otras clases de las cuales dependen, siéndoles inyectadas por el contexto de la aplicación.

Dentro de los creacionales se utilizó el patrón Singleton el cual se aplicó para que las clases de lógica del negocio de las cuales dependen otras clases como las controladoras, pudieran ser creadas como instancias únicas, que pudieran ser reutilizado sin necesidad de ser instanciadas nuevamente. Como patrón estructural se utilizó el Facade (Fachada) con el fin de agrupar varias funcionalidades que pertenecen a un subsistema que interactúa con una gran cantidad de clases que se encargan de la lógica del negocio sobre las entidades del dominio u otras fuentes de datos de la aplicación. Dentro de los de comportamiento al Iterator para iterar sobre listas de Entidades del Negocio.

3.4 Diagramas de Interacción:

El objetivo de esta técnica es describir el comportamiento dinámico del sistema de información mediante el paso de mensajes entre los objetos del mismo. Además representa un medio para verificar la coherencia del sistema mediante la validación con el modelo de clases.

Un diagrama de interacción describe en detalle un determinado escenario de un caso de uso. En él se muestra la interacción entre el conjunto de objetos que cooperan en la realización de dicho escenario. Los elementos que componen los diagramas de interacción son los objetos y los mensajes.

Hay dos tipos de diagramas de interacción: diagramas de secuencia y diagramas de colaboración. Ambos tipos de diagramas tratan la misma información pero cada uno hace énfasis en un aspecto particular en cuanto a la forma de mostrarla. (35)

A continuación se muestra el diagrama de secuencia del CU Autenticar Usuario.

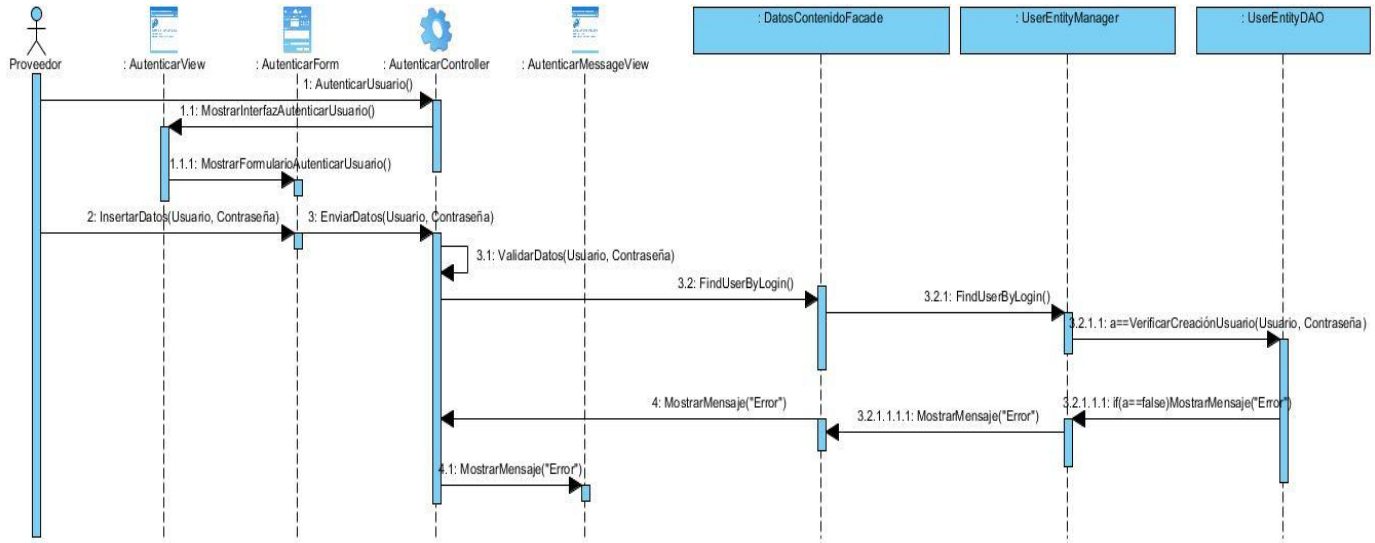

Figura 12: Diagrama de Secuencia CU Autenticar Usuario.

3.5 Conclusiones:

Se definió una arquitectura base flexible y adaptada a las funcionalidades del sistema. Se estudiaron y aplicaron patrones de diseño para asignar responsabilidades a las clases, necesarios para la implementación. Además, se ilustraron los diagramas de clases del sistema, se describió la interacción entre objetos a través de los diagrama de secuencias.

Capítulo 4: Implementación y Prueba

Introducción:

A partir de los resultados obtenidos del Diseño, en el presente capítulo se realiza el modelo de implementación. El propósito es implementar las clases, exponer la distribución del sistema en nodos y los protocolos de comunicación entre cada uno de ellos. En este capítulo se diseñan los casos de prueba que permitirán evaluar y valorar la calidad del producto.

4.1 Modelo de Implementación:

4.1.1 Diagrama de Despliegue:

Un diagrama de despliegue modela la topología del hardware sobre el que se ejecuta un sistema. Muestra la configuración de los nodos que participan en la ejecución y los protocolos de comunicación. Un nodo es un elemento físico que existe en tiempo de ejecución. Representa un recurso computacional que, por lo general, tiene memoria y capacidad de almacenamiento. Además, representa el despliegue físico de un componente.

A continuación se muestra el diagrama de despliegue del sistema:

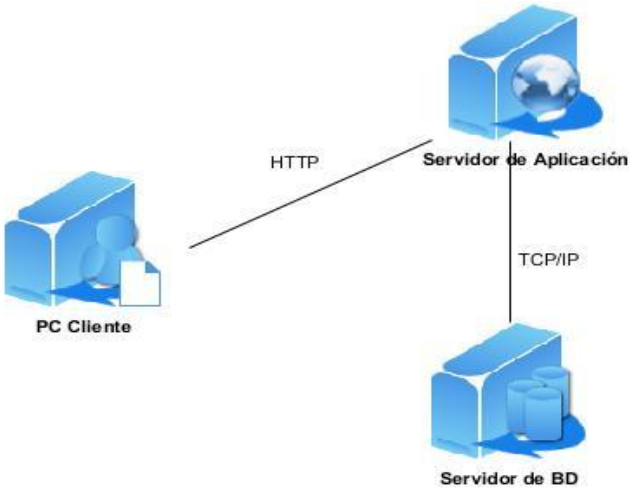

Figura 13: Diagrama de Despliegue.

4.2 Modelo de Prueba:

Las pruebas de software son un conjunto de herramientas y técnicas que evalúan el desempeño de un programa. En todo proceso de desarrollo de aplicaciones es indispensable la presencia de un proceso de pruebas de software que garantice el buen funcionamiento y la calidad del producto final. Según Pressman:

"Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación".

Las pruebas de software permiten conocer hasta qué punto las funciones del software operan de acuerdo con las especificaciones y requisitos del cliente. Estas involucran las operaciones del sistema evaluando los resultados bajo condiciones específicas, lo que hace que la realización de pruebas a los programas constituya un factor de vital importancia. El objetivo de las pruebas, expresado de forma sencilla, es encontrar el mayor número posible de errores en un período de tiempo moderado y con la mínima cantidad de esfuerzo. Un proceso de prueba completo debe garantizar además que los defectos encontrados sean corregidos antes de entregar el software al cliente.

4.2.2 Pruebas Funcionales:

Las pruebas funcionales son aplicadas a un sistema para verificar los dominios de entrada y salida del programa y así descubrir errores en la funcionalidad y comportamiento del mismo. A la aplicación se le realizarán pruebas de caja negra que se centran en los requisitos funcionales del software. Estas pruebas se llevan a cabo sobre la interfaz del software mediante un conjunto de datos de entrada que ejercitan completamente todos los requisitos funcionales de un programa. Examina aspectos del modelo principalmente del sistema sin tener mucho en cuenta la estructura interna del software. La herramienta que se utilizará para la ejecución de estas pruebas serán los casos de prueba.

4.2.3 Diseño de Casos de Prueba:

Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto.
Se utilizarán para la aplicación las pruebas de caja negra, ya que están diseñadas para verificar los requerimientos del usuario. A continuación se muestra el caso de prueba realizado a partir de la descripción del caso de uso Autenticar Usuario.

- **CP Autenticar Usuario:**

Descripción General: El proveedor entra los datos correspondientes, una vez dentro este podrá realizar los cambios correspondientes y mantener actualizados sus contenidos.

Condiciones de Ejecución: El proveedor debe estar registrado en la base de datos.

SC Autenticar Usuario:

Escenario	Descripción	Usuario	Contraseña	Respuesta del sistema	Flujo central
EC 1.1 Autenticación satisfactoria.	Se accede a la aplicación.	V cubarte	V cubarte123	Muestra la página principal.	1-El usuario entra a la página “Autenticar”. 2-Entra los datos solicitados. 3-Da clic en el botón “Entrar”.
EC 1.2 Campos Vacíos	El usuario deja datos sin llenar.	-	V cubarte123	Se muestra un mensaje de error “Credenciales incorrectas, intente de Nuevo”.	1-El usuario entra a la página “Autenticar”. 2-Entra los datos solicitados. 3-Da clic en el botón “Entrar”.
		V cubarte	-	Se muestra un mensaje de error “Credenciales incorrectas,	

				intente de Nuevo”.	
		-	-	Se muestra un mensaje de error “Credenciales incorrectas, intente de Nuevo”.	
EC 1.3 Datos Incorrectos	El usuario introduce valores no permitidos en los datos	I cub@rte	V cubarte123	Muestra un mensaje de error “Credenciales incorrectas, intente de Nuevo”.	1-El usuario entra a la página “Autenticar”. 2-Entra los datos solicitados. 3-Da clic en el botón “Entrar”.

Tabla 3: CP Autenticar Usuario SC Autenticar Usuario.

Descripción de las Variables:

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Usuario	Campo de texto	No	No puede tener números ni caracteres especiales.
2	Contraseña	Campo de Texto	No	Acepta cualquier valor.

Tabla 4: CP Autenticar Usuario. Descripción de las Variables.

Una vez finalizado el proceso de pruebas se arribó a las siguientes conclusiones:

Fueron realizadas tres iteraciones arrojando para una primera iteración 24 no conformidades de tipo Validación y 2 no conformidades del tipo Opciones que no funcionan, en una segunda iteración fueron encontradas 2 no conformidades del tipo Validación y 1 no conformidad del tipo Opciones que no funcionan mientras que para una tercera iteración no se encontraron no conformidades. A continuación se muestra una imagen que aprueba lo anteriormente descrito.

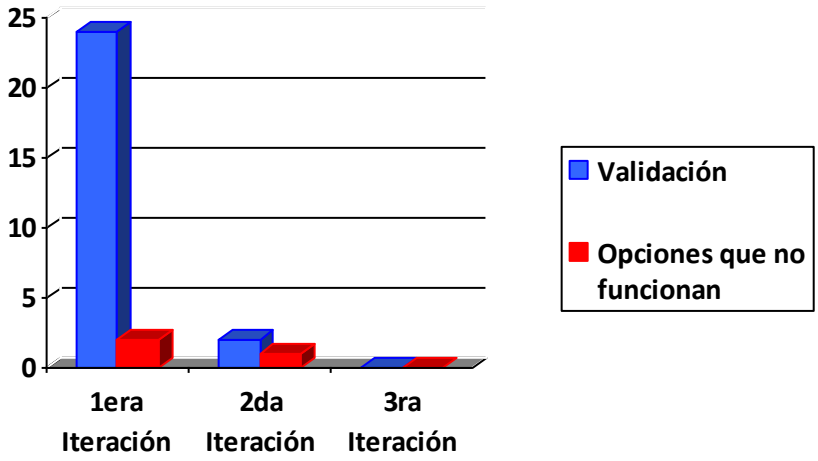

Figura 14: Gráfico No Conformidades detectadas.

4.3 Conclusiones del Capítulo:

En este capítulo se describió la distribución del sistema en nodos, especificados en el diagrama de despliegue. Además se puede apreciar el desarrollo de los casos de pruebas realizados a algunas funcionalidades de la aplicación, mostrando que dichas funcionalidades no presentan anomalías en su funcionamiento ya que realizan un correcto tratamiento de los datos que manejan. Por lo que se concluye que el resultado de las pruebas realizadas ha sido satisfactorio.

Conclusiones Generales:

La presente investigación permitirá dar una solución a la problemática planteada. La aplicación de administración para proveedores de la plataforma de entrega de contenidos para móviles permitió que los proveedores pudieran gestionar sus contenidos, así como obtener reportes de los mismos, de esta manera el proveedor se convierte en el responsable de la creación de los contenidos que se ofrecen. Esto influye en el aumento de las ganancias que se generan puesto que se satisface las necesidades del usuario al brindarle lo que le interesa.

El país no cuenta con una plataforma de Gestión de Contenidos para Dispositivos Móviles, por lo que la aplicación permitirá que Cubacel cuente con una herramienta que gestione los servicios que proporcionan los proveedores y ayudará al crecimiento de la telefonía celular ya que ofrecerá nuevos servicios a la población. El mero hecho de utilizar la misma sitúa a ETECSA a altura de determinadas empresas que se dedican a negocios similares que proveen las mismas funcionalidades a un costo elevado.

En este trabajo se realizó el estudio de las herramientas y tecnologías de desarrollo que se utilizaron en la confección de la solución. Finalmente se escogieron las que resultaron más adecuadas para el desarrollo del sistema. Se desarrolló la aplicación Web sobre la plataforma J2EE, específicamente con la tecnología Spring Frameworks, con los subproyectos Spring Web Service y Spring Security. Se utilizó la metodología RUP, se concibió la arquitectura del sistema y se generaron los demás artefactos pertenecientes al diseño. Durante el flujo de trabajo de implementación se realizó el diagrama de despliegue, lo que facilitó la comprensión de la comunicación entre los componentes y la distribución física de estos para la implementación del sistema. Además, se diseñaron casos de prueba para evaluar y valorar la calidad del sistema, logrando así una revisión final de las especificaciones del diseño y de la codificación. Como resultado de este trabajo se desarrolló un sistema que provee una interfaz de usuario a los proveedores para cada una de las funcionalidades que brinda la plataforma COMCEL que cumple con los requisitos establecidos inicialmente. Por todo lo anterior expuesto se concluye que los objetivos trazados fueron cumplidos.

Recomendaciones:

Se recomienda:

- ✓ Aumentar la seguridad del sistema utilizando el mecanismo de Certificados Digitales.
- ✓ Permitir mostrar reportes gráficos.
- ✓ Aplicar una estrategia de marketing para captar la atención de sus clientes, dar promoción a sus productos y servicios, logrando el incremento de los afiliados y el aumento de los ingresos por descargas de los contenidos a la plataforma.

Referencias Bibliográficas:

1. Volantis Content Delivery Platform. [En línea] http://www.volantis.com/content-delivery-platform.
2. Mobile Web & Content Delivery Platform. BeeWeb technologies.[En línea] http://www.beeweeb.com/mwt/index.php/products/mobile-web-content-delivery-platform/.
3. Content Management Bible, Wiley Publishing. Ilustraciones de Antoine+Manuel enero de 2005, actualizado en noviembre de 2009. [En línea] Boiko, B (2002).
4. De gestión de contenidos para redes inalámbricas - Insight Research Informe. [En línea] 2008-2013.
5. Interfaz de Usuario. . [En línea] http://www.fismat.umich.mx/~crivera/tesis/node6.html.
6. Pirámide del Diseño Web . [En línea] 02 de 12 de 2009. http://www.rena.edu.ve/cuartaEtapa/Informatica/Tema14.html.
7. Ventajas y Beneficios de una Aplicación Web. [En línea] Esencia Humana, 2008. http://www.esenciahumana.com.mx/Servicios/AplicacionesWeb/VentajasBeneficiosAplicaciones.html.
8. Cuba reporta explosivo avance de la telefonía celular. AméricaEconomía . El sitio de los negocios globales de América Latina. . [En línea] http://www.americaeconomia.com/negocios-industrias/cuba-reporta-explosivo-avance-de-la-telefonia-celular.
9. Dave Thomas, David Heinemeier Hansson. Agile Web Development with Rails, Second Edition.The Pragmatic Bookself. [En línea] Addison Wesley Professional. http://upcommons.upc.edu/pfc/bitstream/2099.1/5542/2/Memoria.pdf.
10. El lenguaje de programación C#, José Antonio González Seco, El lenguaje de programacion Csharp.pdf.
11. The PostgreSQL Global Development Group. PostgreSQL. [En línea] http://www.postgresql.org/docs/8.4/static/intro-whatis.html.
12. MySQL 5.0 Reference Manual. MySQL. [En línea] http://es.scribd.com/doc/10037315/MySQL-50-Reference-Manual-Espanol#outer_page_23.

Referencias Bibliográficas:

13. Claudio Ferreira, Jorge Ocampos y Evelyn Cuenca. Servidor de Aplicaciones Apache Tomcat. [En línea] 26 de Mayo de 2011. http://mistock.lcompras.biz/sysoper/1362-sysop2011-servidor-de-aplicaciones-apache-tomcat.
14. Martin, Cesar Crespo. Soporte a desarrollo informático. [En línea] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernate.
15. Investigación de la Plataforma J2EE y su Aplicación Práctica. [En línea] JUAN MANUEL BARRIOS NUÑEZ. , 2003.
16. Plataforma .NET. [En línea] http://blogs.msdn.com/b/mohammadakif/archive/2005/11/15/493161.aspx.
17. Spring Framework. [En línea] http://www.genbetadev.com/java-j2ee/spring-framework-introduccion
18. Eclipse. [En línea] http://www.webmaster-mexico.com/book/export/html/30.
19. Booch Grady, Jacobson Ivar, Rumbaugh James. El Proceso Unificado de Desarrollo de Software. Capítulo 1 Páginas 1-12. [En línea] Addison Wesley , 2000.
20. Rational Unified Process. [En línea] http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf.
21. Extreme Programming: A gentle introduction [En línea] http://www.extremeprogramming.org.
22. Roger Armando Contreras Corrales. Introducción a JasperReports. [En línea] http://es.scribd.com/doc/37388195/Manual-de-Ireport.
23. Enrique Viñe Lerma. Spring Security. [En línea] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=utilizaciondegruposenspringsecurity.
24. Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform] 6.0). [En línea] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.
25. Diagrama de Casos de Uso. [En línea] Jesús Cáceres Tello. http://www2.uah.es/jcaceres/capsulas/DiagramaCasosDeUso.pdf.
26. Arquitectura de software. [En línea] http://es.wikipedia.org/wiki/Arquitectura_de_software.

Referencias Bibliográficas:

27. Arquitectura - IEEE 1471-2000. [En línea]
28. Estilos Arquitectónicos. Definición. [En línea] http://www.buenastareas.com/ensayos/Estilos-Arquitect%C3%B3nicos/1019429.html.
29. Application Architecture Guide 2.0, J.D. Meier, Alex Homer, David Hill, Jason Taylor, Prashant Bansode, Lonnie Wall, Rob Boucher Jr, Akshay Bogawat. [En línea] Microsoft Corporation., 2008
30. Arquitectura en Capas ~ DNA,Un camino hacia los procesos distribuidos. [En línea] José Enrique González Cornejo, 25 de 03 de 2001 http://www.docirs.cl/arquitectura_tres_capas.htm.
31. Estructura de las Aplicaciones Orientadas a Objetos,El patrón Modelo-Vista-Controlador (MVC),. [En línea] 16 de 01 de 2012. http://www.fdi.ucm.es/profesor/jpavon/poo/2.14.MVC.pdf.
32. Daniel Riesco. UML Diagrama de Clases y de Objetos. [En línea] 12 de 04 de 2010. http://sel.unsl.edu.ar/licenciatura/ingsoft2/UML-DiagramaClaseObjeto.pdf.
33. Catálogo de Patrones de Diseño J2EE. Y II: Capas de Negocio y de Integración. [En línea] http://www.programacion.com/java/tutorial/patrones2/8/.
34. Spring_tutorial_v0.271.pd . [En línea] 29 de 12 de 2004.
35. Técnicas y Prácticas,Metodología MÉTRICA Versión 3. [En línea] http://es.scribd.com/doc/37200680/14/Diagrama-de-Paquetes.

Bibliografía:

- ✓ Larman Craig. “UML y Patrones”. Prentice Hall Hispanoamericana 1999.
- ✓ Jacobson, I.; Booch, G. y Rumbaugh, J.; “El Proceso Unificado de Desarrollo de software”, Addison-Wesley, 2000.
- ✓ Hernández León Rolando A., Coello González Sayda. El Paradigma Cuantitativo de la Investigación Científica. EDUNIV 2002.
- ✓ Patrones de diseño software,Ramiro Lago (Abril 2007), http://www.proactiva-calidad.com/java/patrones/index.html#algunos_patrones, 16 enero 2012.
- ✓ UML y Patrones. Introducción al análisis y diseño orientado a objetos - Larman - Prentice Hall.
- ✓ Design Patterns. Elements of Reusable Object-Oriented Software - Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides - Addison Wesley (GoF- Gang of Four) .
- ✓ A System of Patterns - Buschmann, Meunier, Rohnert, Sommerlad, Stal – Wiley.
- ✓ Fundamentos de Ingeniería de software, Marcello Visconti y Hernán Astudillo. [En línea]
- ✓ Booch Grady, Jacobson Ivar, Rumbaugh James. El Proceso Unificado de Desarrollo de Software. s.l. [En línea] Addison Wesley, 2000.
- ✓ Murphey, Rebecca. Fundamentos de Jquery. [En línea] febrero de 2012. http://librojquery.com/.

Bibliografía:

- ✓ Hernando, Rocio Martin. Sistemas de e-Participación: evaluación de la arquitectura, diseño e implementación de un prototipo que incorpora gestión de organizaciones y territorios. [En línea] Febrero de 2010.

Glosario de Términos:

- ✓ **IU:** Interfaz de Usuario.
- ✓ **Plataforma:** Es una aplicación que se encarga de prestar servicios para la descarga y otros servicios relacionados con cierto contenido.
- ✓ **Aplicación:** Programa informático diseñado para permitir a un usuario realizar diversos tipos de trabajo.
- ✓ **JSP:** Java Server Pages (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.
- ✓ **XML:** Extensible Markup Language (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas.
- ✓ **CSS**: Las hojas de estilo en cascada (en inglés *Cascading Style Sheets*), CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML).
- ✓ **Arquitectura**: Descripción de la organización y estructura de un sistema.
- ✓ **API**: (Del ingles Application Programming Interface). Es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usados generalmente en las bibliotecas.
- ✓ **Actores**: Se le llama actor a toda entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad.
- ✓ **Casos de uso**: Un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema.
- ✓ **Código fuente**: El código fuente de un programa informático (o software) es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa.

Glosario de Términos:

- ✓ **DAO**: (Data Access Object, en español Objeto de Acceso a Datos). Es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.
- ✓ **GRASP**: En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsabilidades, es el acrónimo de "General Responsibility Assignment Software Patterns".
- ✓ **HTTP**: Hypertext Transfer Protocolo (en español protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web. Define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse.
- ✓ **IoC**: (Inversion of Control, en español Inversión de Control): Es un método de programación en el que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales, en los que la interacción se expresa de forma imperativa haciendo llamadas a procedimientos (procedure calls) o funciones.
- ✓ **JSP**: (Del inglés Java Server Pages). Es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.
- ✓ **Java Beans**: Los JavaBeans son un modelo de componentes creado por Sun Microsystems para la construcción de aplicaciones en Java. Se usan para encapsular varios objetos en un único objeto (la vaina o Bean en inglés), para hacer uso de un solo objeto en lugar de varios más simples.
- ✓ **SQL**: El lenguaje de consulta estructurado (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas.
- ✓ **Open Source**: El software libre y de código abierto (también conocido como FOSS o FLOSS, siglas de free/libre and open source software, en inglés) es el software que está licenciado de tal manera que los usuarios pueden estudiar, modificar y mejorar su diseño mediante la disponibilidad de su código fuente.
- ✓ **Controller:** Maneja la lógica de navegación e interactúa con la capa de servicio para la lógica de negocio.