Universidad de las Ciencias Informáticas

Facultad 6

SOFTWARE PARA LA CONVERSIÓN DE MÚLTIPLES CAPAS DE INFORMACIÓN GEOGRÁFICA DE ESRI SHAPEFILE A POSTGRESQL

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

AUTOR: Joel Michel León Estrada

TUTOR: Ing. Armando Batista Piñeda

DECLARACIÓN DE AUTORÍA

Por este medio declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente del mismo.

Para que así conste firmo la presente a los ____ días del mes de junio del 2011.

_____	_____
Nombre y Apellidos del Autor	Ing. Armando Batista Piñeda
Autor	Tutor

DEDICATORIA

El presente Trabajo de Diploma está dedicado a las personas más importantes de mi vida:

A mis padres Gisel e Isanto, por entregarme todo su amor, confianza, dedicación y comprensión, por su apoyo y sacrificio incondicional en todo momento, a ustedes les debo la vida. Gracias por existir….

A mis abuelos Isabel y Rubén Antonio (El papa), sus consejos y ejemplo me han hecho la persona que soy hoy, sin ustedes la vida no sería igual, los adoro mis viejucos.

A mí querida tía María Adelita, por ser como una madre para mí y hacerme sentir como en casa, por tener ese corazón tan grande y bondadoso y enseñarme a luchar por lo que se quiere sin importar las adversidades, he aprendido mucho de ti, te quiero mucho tía.

A mis lindas hermanitas Roxana y Lilian, por ser mi alegría.

A todos ustedes y a mi familia en general va dedicado este trabajo de diploma.

AGRADECIMIENTOS

El presente trabajo no es producto de la labor individual, sino más bien el resultado del esfuerzo de muchas personas durante todos los años de formación; para todos ellos mis más sinceros agradecimientos. Quiero agradecer especialmente:

A mis padres, abuelos y hermanas, por ser mi razón de ser e inspiración.

A mis tias: Maria Adela, Maria Adelita, Gloria Yanet y Sarita, a mis tios: Enrique y Rodolfito, a mi primo Enris y Sucel por acogerme en su hogar y ofrecerme apoyo en todo momento.

A mi tutor Armando Batista Piñeda por estar disponible en todo momento y brindarme su mano en cualquier situación.

A mis amigos de la infancia, La Itera, José, Peña, El negro, El Gago, por lo que hemos aprendido y pasado juntos, por estar siempre presentes en cualquier situación.

Al piquete de los "Pentium Boys" Filo, Ameba, Fresa, Puty, Bana, Pitbull, Mamita, El Guacho, por ser tan buenos compañeros, más que compañeros amigos.

A mi novia por permitirme descubrir tantas cosas lindas y brindarme su afecto y cariño.

A todos mis profesores.

RESUMEN

El formato ESRI Shapefile es considerado uno de los más difundidos actualmente en el mundo de la cartografía digital. Por otra parte, PostgreSQL se ha convertido en un medio de referencia en lo que respecta al almacenamiento de datos geográficos en Bases de Datos. PostgreSQL cuenta con la extensión PostGIS que permite manejar los datos georreferenciados y realizar un sinnúmero de operaciones sobre estos, facilitando el análisis de la información. Es por ello que el proceso de conversión de datos de ESRI Shapefile a PostgreSQL se ha hecho bastante frecuente en el desarrollo de Sistemas de Información Geográfica (SIG). Debido a que la Universidad de las Ciencias Informáticas (UCI) está inmersa en el desarrollo de varios SIG, en el presente trabajo se propone el desarrollo de una herramienta capaz de transformar múltiples archivos Shapefile a PostgreSQL desde un entorno web, presentando como característica fundamental, proporcionarle más rapidez al proceso de preparación de la cartografía para los SIG desarrollados en el departamento Geoinformática y Señales Digitales (GEYSED).

Palabras clave:

Conversión, ESRI Shapefile, Sistema de Información Geográfica, PostgreSQL

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Actualmente las Tecnologías de la Información y las Comunicaciones (en lo adelante TIC) se desarrollan de forma vertiginosa, dicha situación afecta a prácticamente todos los campos de la sociedad. Las TIC se presentan cada vez más como una necesidad en el contexto de sociedad, donde los rápidos cambios, el aumento de los conocimientos y las demandas, se convierten en una exigencia permanente.

En la actualidad, las TIC aparejado al auge de la informática ha cambiado la forma en la cual el hombre observa e interactúa con el entorno que lo rodea. De manera que desde un ordenador se puede viajar virtualmente a cualquier parte del mundo y obtener todo tipo de información con tan solo un clic. Las TIC posibilitan un rápido acceso a grandes fuentes de información de cualquier índole, gran capacidad de almacenar datos, así como la digitalización de la información.

Actualmente la globalización tecnológica es una realidad social mundial, los gobiernos de todo el mundo y en especial los países desarrollados ven en las TIC la gran posibilidad de su desarrollo cultural, científico, tecnológico y económico; por lo cual existe una gran brecha digital, donde los países más pobres son los que más sufren de este efecto. De ahí que haya surgido como alternativa el Software Libre (en lo adelante SWL).

El SWL tiende a seguir creciendo y cada vez son más los usuarios que migran hacia este camino, como alternativa a la globalización tecnológica pues ofrece una oportunidad de informatización y comercio de los países. En la actualidad sobre la base del SWL se trabaja en los Sistemas de Información Geográfica (en lo adelante SIG) con el objetivo de capturar, manipular, analizar y modelar datos geográficos espacialmente referenciados. Gracias a este tipo de sistemas se puede observar en un mapa cualquier información que tenga un componente geográfico, para poder encontrar patrones, relaciones y tendencias que no pueden verse en otro formato. Los SIG dan una perspectiva totalmente nueva y dinámica de la información que permite ser más eficiente en la toma de decisiones.

La información con la que trabaja un SIG es almacenada en base de datos espaciales, las cuales presentan la información a través de mapas y símbolos. Como ejemplo fehaciente en Cuba sobre el desarrollo del SWL y los SIG está la Universidad de las Ciencias Informáticas (UCI) donde se desarrollan aplicaciones con gran calidad en esta esfera, de la cual es rectora la Facultad 6 y su centro de desarrollo GEYSED. En este centro se lleva a cabo la construcción de varios SIG para uso nacional e internacional, potenciando así las relaciones con la hermana República Bolivariana de Venezuela.

La mayoría de los aplicativos SIG desarrollados utilizan PostgreSQL para almacenar datos, sin embargo, en muchos casos la información que tienen los clientes procede de diversas fuentes como pueden ser: Web Map Service (Servicio de mapas para la web WMS), es un estándar para publicar cartografía en Internet, servicio que usa el Software GvSig[1]; Geography Markup Language (Lenguaje de Marcado Geográfico GML) utilizado para el modelaje, transporte y almacenamiento de información geográfica y ESRI[2] Shapefile. En el caso de la última mencionada, es válido destacar que es uno de los formatos más usados a nivel mundial, prueba de ello es que aplicaciones líderes en el campo de la representación geográfica como ArcView GIS, ArcGIS y ArcInfo utilizan este formato y potencian su uso.

Por este motivo, uno de los procesos que se ejecutan con relativa frecuencia es la transformación de la información existente en formatos varios a PostgreSQL. Previendo un posible aumento de clientes, para que los productos alcancen el éxito, se requiere que el proceso sea eficiente en cuanto al tiempo, sumándole a esta razón que los equipos de desarrollo generalmente están compuestos por especialistas, profesores y estudiantes bajo un modelo que relaciona tres componentes

[1] Asociación que ofrece servicios para implantar sistemas de información geográfica con software libre.

[2] Enviromental Systems Research Institute.

fundamentales: docencia, investigación y producción. Bajo este modelo, la estabilidad del personal implicado es limitada. Con relativa frecuencia entran nuevos miembros, se realizan movimientos y otras actividades que generan un contexto bastante dinámico.

Para realizar esta tarea existen algunas herramientas que contribuyen a la conversión de este tipo de formato, pero no satisface las necesidades que se presentan a la hora de preparar la cartografía. El proceso de conversión de los datos en formato ESRI Shapefile a PostgreSQL actualmente no se considera satisfactorio y es posible introducir mejoras en el mismo.

Atendiendo a la situación problemática antes expuesta se plantea el siguiente **problema a resolver,** ¿Cómo agilizar el proceso de transformación de datos espaciales georreferenciados en formato ESRI Shapefile a PostgreSQL requerido en Sistemas de Información Geográfica desarrollados en el Departamento Geoinformática?

Definiendo como **objeto de estudio,** el proceso de migración de cartografía digital hacia base de datos espaciales, enmarcado en el **campo de acción**, el proceso de transformación de datos en formato ESRI Shapefile a PostgreSQL.

En tanto se propone como **objetivo general**, desarrollar un software que permita convertir múltiples capas de información geográfica en formato ESRI Shapefile a PostgreSQL según los estándares establecidos a nivel internacional.

Por lo que se presenta la siguiente **idea a defender,** desarrollar una herramienta informática que permita convertir múltiples capas de información geográfica en formato ESRI Shapefile a PostgreSQL, agilizará este proceso de transformación requerido en los Sistemas de Información Geográfica desarrollados en el Departamento Geoinformática.

Para el cumplimiento de los objetivos planteados se establecen las siguientes **tareas de investigación**:

- ✓ Establecer los fundamentos teóricos asociados al proceso de migración de cartografía digital hacia base de datos espaciales.
- ✓ Seleccionar herramientas y metodología adecuadas para el desarrollo de la solución.

- ✓ Identificar los requisitos del sistema.
- ✓ Diseñar la solución de la aplicación.
- ✓ Implementar el diseño realizado.
- ✓ Probar que el producto cumpla las especificaciones definidas.
- ✓ Documentar el proceso realizado para su socialización.

Para desarrollar las tareas de investigación antes propuestas se hace necesario auxiliarse de los siguientes métodos científicos.

- ✓ ***Histórico Lógico*****:** Para el análisis histórico de los procesos de conversión del formato ESRI Shapefile en el mundo y como está compuesto internamente.
- ✓ ***Analítico Sintético*****:** Este método permite el estudio de la bibliografía consultada y la recopilación de la misma, además del análisis de los conceptos asociados al tema que se investiga.
- ✓ ***Observación:*** Se utiliza para conocer las distintas vías de conversión del formato ESRI Shapefile, así como para definir tanto los requisitos funcionales como no funcionales.
- ✓ ***Modelación:*** Para mostrar los diversos diagramas que se construyen como resultado del proceso de Ingeniería de Software.

El presente trabajo de diploma está compuesto por 4 capítulos:

Capítulo 1: Fundamentación Teórica. Se exponen los conceptos fundamentales que sustentan la base de los capítulos restantes. Se describe la situación problemática y el análisis de otras soluciones que puedan brindar respuesta al problema científico planteado en el presente trabajo.

Capítulo 2: Tendencias y tecnologías a utilizar. Se explican las principales tecnologías, lenguajes de programación, metodologías y herramientas para utilizar en la construcción de la solución propuesta, así como las ventajas de su uso.

Capítulo 3: Presentación de la solución propuesta. Se describen los procesos actuales a través de un modelo de dominio, se seleccionan los requisitos funcionales y no funcionales y los casos de uso del sistema, todo esto unido a los correspondientes diagramas que lo modelan.

Capítulo 4: Construcción de la solución propuesta. Se plantea la construcción propuesta en el capítulo anterior, en función de diagramas de clases, estándares del diseño y las pruebas pertinentes.

1

CAPÍTULO 1.FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se abordarán los conceptos fundamentales al dominio del problema para una mejor comprensión referente a la conversión del formato ESRI Shapefile. Así como el análisis de soluciones existentes que conviertan este tipo de formato.

1.2 Conceptos asociados al dominio del problema

Para una mejor comprensión de esta investigación es necesario definir algunos conceptos asociados al problema que no necesariamente deben formar parte del fondo de conocimiento de personas interesadas en este trabajo.

La cartografía digital es el procedimiento que transforma la información geográfica de los mapas de papel a coordenadas digitales (Perú Global Consultin, 2008).

Una base de datos (BD) es el lugar donde se agrupan un conjunto de datos organizados que pertenecen a un mismo contexto. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenar datos (Gavin, 2010).

PostgreSQL es un servidor de bases de datos relacionales, que destaca por llevar 15 años en desarrollo y contar con una arquitectura robusta, ofrecer una gran fiabilidad e integridad de datos. Funciona en la mayoría de los sistemas operativos actuales, incluyendo Linux Unix y Windows (PostgreSQL, 2009).

PostgreSQL es un software libre bajo licencia Distribución de Software Berkeley (*Berkeley Software Distribution* BSD), y como en el caso de muchos otros proyectos de código abierto, su desarrollo no

depende de una sola compañía, sino de una comunidad de desarrolladores y organizaciones comerciales.

PostGIS permite usar PostgreSQL como una base de datos espaciales incluida dentro de un SIG, tal y como hacen otros software como: Motor de Base de Datos Espaciales (*Spatial Database Engine* SDE) de ESRI u Oracle Spatial. Así, PostgreSQL con la extensión PostGIS permite almacenar tablas con datos alfanuméricos que además pueden contener columnas compuestas por geometrías, tales como puntos, líneas poligonales o polígonos. También añade operadores geométricos (por ejemplo distancia) y topológicos (por ejemplo intersección). Además, tiene soporte para diferentes sistemas de referencia espacial. Fue desarrollado y es mantenido por Refractions Research, como un proyecto de código abierto distribuido bajo licencia GNU (Muñoz-Cruzado García, 2006).

Los Datos Espaciales son el componente fundamental de cada proyecto o aplicación SIG. Contienen las ubicaciones y formas de características cartográficas. Son también conocidos como Datos Cartográficos Digitales, el tipo de datos necesarios para crear mapas y estudiar relaciones espaciales. Dentro de su contexto, almacenan información sobre la localización y las formas de un objeto geográfico y las relaciones entre ellos, normalmente con coordenadas y topología. Se refieren a entidades o fenómenos que cumplen los siguientes principios básicos (Geotecnologias, 2007):

- ✓ Posición absoluta: sobre un sistema de coordenadas (x, y, z).
- ✓ Posición relativa: frente a otros elementos del paisaje (topología: incluido, adyacente, cruzado).
- ✓ Figura geométrica que representan (punto, línea, polígono).
- ✓ Atributos que describen (características del elemento o fenómeno).

El Lenguaje de Marcado (*Keyhole Markup Language* KML) es un formato de archivo que se utiliza para mostrar información geográfica en navegadores terrestres como Google Earth, Google Maps y Google Maps para móviles. KML utiliza una estructura basada en etiquetas con atributos y elementos anidados y está basado en el estándar Lenguaje de Marcado Extensible (Extensible *Markup Language* XML) (Google Code, 2010).

El Lenguaje de Marcado Geográfico (*Geography Markup Language* GML) es una gramática XML para expresar las características geográficas. GML sirve como un lenguaje de modelado de sistemas geográficos, así como un formato de intercambio abierto para transacciones geográficas en Internet. Como con la mayoría de las gramáticas basadas en XML, hay dos partes de la gramática, el esquema que describe el documento y el documento de instancia que contiene los datos reales. Un documento GML se describe mediante un esquema de GML. Esto permite a los usuarios y desarrolladores para describir conjuntos de datos geográficos genéricos que contienen puntos, líneas y polígonos (Opengeospatial, 2010).

Un Sistema de Información Geográfica es una integración organizada de hardware, software y datos geográficos diseñado para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestres y construidas para satisfacer necesidades concretas de información. En el sentido más estricto, es cualquier sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada. En un sentido más genérico, los SIG son herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones (Salinas, 2007).

1.3 Soluciones existentes

En el mundo se han creado varios programas y herramientas que posibilitan la transformación del formato ESRI Shapefile a otros tipos de formatos, sin embargo las búsquedas realizadas no arrojaron soluciones similares en entornos web. Existen soluciones de tipo Desktop que pueden servir de referencia para el sistema a desarrollar. A continuación se exponen las más significativas a consideración del autor.

1.3.1 Shp2kml 2.0: Forma de archivos a Google Earth

Shp2kml es una herramienta independiente que transforma las capas de los SIG para Google Earth. Se utiliza como entrada el archivo en formato más común para los SIG (ESRI Shapefile) y genera un

archivo KML. Google Earth requiere las coordenadas para estar en Latitud / Longitud y referidos a la datum WGS84[3]. Shp2kml es capaz de transformar el sistema de coordenadas.

La entrada de un archivo puede ser de latitud y longitud (geográfica) o UTM[4] (proyectada) sistema de coordenadas. También shp2kml cambiará datums si es necesario. Contiene una lista de alrededor de 200 datums (2009).

1.3.2 Conversión de archivos shape2Text

Shape2Text transforma archivos de forma ESRI en archivos ASCII, los ficheros ASCII tienen la ventaja de que puede ser interpretado por casi cualquier aplicación. Algunas de las extensiones más comunes para los archivos CSV[5] son ASCII (coma valores separados) que se pueden abrir directamente en una hoja de cálculo de Microsoft Excel y TXT que se puede abrir con la aplicación Bloc de notas. El shape2Text puede guardar el archivo de salida como CSV, TXT y el formato TAB (2009).

1.3.3 Quantum GIS

Quantum GIS (QGIS) es un SIG de código libre para plataformas GNU/Linux, Unix, Mac OS y Windows. Permite manejar formatos raster y vectoriales, así como BD. Algunas de sus características son:

- ✓ Soporte para la extensión espacial de PostgreSQL (PostGIS).

[3] Es un sistema de coordenadas cartográficas mundial que permite localizar cualquier punto de la Tierra.

[4] Universal Transverse Mercator Mapa del mundo en proyección transversa de Mercator, centrado sobre el meridiano 0º y el ecuador.

[5] Comma Separated Values (son un tipo de documento en formato abierto sencillo para representar datos en forma de tabla)

- ✓ Manejo de archivos vectoriales Shapefile.
- ✓ Soporte para un importante número de tipos de archivos raster (GeoTIFF, JPEG).

Una de sus mayores ventajas es la posibilidad de usar Quantum GIS como GUI del SIG GRASS[6], utilizando toda la potencia de análisis de este último en un entorno de trabajo más amigable. Tiene integrado un plugin para la conversión de archivos Shapefile a PostgreSQL, está desarrollado en C++, usando la biblioteca QT para su interfaz gráfica de usuario.

1.4 Objeto de Estudio

1.4.1 Descripción general

La revolución de la informática llegó a la cartografía un poco después que a otras disciplinas. Las primeras computadoras servían para almacenar números y textos, pero los mapas, en cambio, son complejos y para un mapa digital se requiere una gran capacidad para almacenar esos datos, una tarea para la cual las primeras computadoras no tenían mucha capacidad.

Las modernas tecnologías SIG trabajan con información digital, para la cual existen varios métodos utilizados en la creación de datos digitales. El método más utilizado es la digitalización, donde a partir de un mapa impreso o con información tomada en campo se transfiere a un medio digital por el empleo de un programa de Diseño Asistido por Ordenador (DAO o CAD) con capacidades de georreferenciación[7].

Dada la amplia disponibilidad de imágenes orto-rectificadas[8], la digitalización por esta vía se está convirtiendo en la principal fuente de extracción de datos geográficos que después son almacenados

[6] Geographic Resources Analysis Support System es un programa de gestión de SIG.
[7] Se refiere al posicionamiento con el que se define la localización de un objeto espacial.

[8] Cuando se elimina distorsión de imágenes tomadas por vía satelital o aéreas.

en base de datos espaciales. Esta forma de digitalización implica la búsqueda de datos geográficos directamente en las imágenes aéreas, en lugar del método tradicional de la localización de formas geográficas sobre un tablero de digitalización.

1.4.2 Tipos de fuentes de datos

Mediante la siguiente tabla se recogen los distintos tipos de fuentes de datos que son recogidos por distintas vías para después ser guardados en las bases de datos

Tabla 1 Tipos de fuentes de datos

Fuentes de datos	Digitales	No Digitales
Primarios	Levantamiento Topográfico (vectorial). Telemetría (vectorial). Medidas de GPS (vectorial). Imágenes de Satélites (raster).	Observaciones de campo. Documentos lineales (censos, encuestas). Mapas. Fotografías aéreas.
Secundarios	Imágenes de Satélites (raster). Base de Datos digitales. Listas (de direcciones, coordenadas). Documentos de scanner (raster).	Documentos de archivos. Otros mapas e imágenes.

1.4.3 Shapefile

El Shapefile es un formato de representación vectorial desarrollado por ESRI. Consta de un número variable entre 3 y 8 archivos. Cada uno de estos archivos tiene una función específica y almacena un tipo de información (elementos geométricos, atributos, proyección, metadatos). Los elementos geométricos se almacenan mediante sus vértices en el archivo shape. Actualmente, cada Shapefile solo puede tener un tipo de elementos (puntuales, lineales o zonales). Dependiendo del tipo de Shapefile, también podremos almacenar valores de altura (PointZ) o mediciones (PointM) en estos vértices.

Partes de un Shapefile

Hay tres archivos que resultan imprescindibles en todo Shapefile, los cuales son:

Shape (.shp): Se trata del archivo principal, almacena la información geométrica de los elementos de la capa en formato vectorial. Pueden contener puntos, líneas o polígonos y cada vértice lleva implícitas sus coordenadas en un sistema de referencia concreto. Se componen de una cabecera con información general sobre el tipo de Shapefile y un número variable de registros, que a su vez pueden estar compuestos por varias entidades geométricas independientes.

Shape Index (.shx): Consiste en un índice de las entidades geométricas que permite refinar las búsquedas dentro del archivo .shp.

dBase (.dbf): Se trata de una tabla de datos en la que se registran los atributos de cada elemento. Es un formato con larga historia, muy compatible y sencillo que permite almacenar datos estructurados. En los Shapefile, las tablas dBase se emplean para asignar atributos numéricos, de texto o de fecha a los registros contenidos en el archivo principal. Cada registro debe estar asociado con una única entrada en la tabla, ambos archivos se vinculan mediante un número de registro en el archivo principal.

1.5 Conclusiones

En la elaboración de este capítulo se realizó un estudio del arte donde se abordaron conceptos generales que sirvieron de guía para lograr un mejor entendimiento de la presente investigación. Estudiar y comparar soluciones existentes de programas y herramientas que transforman el formato ESRI Shapefile a otros formatos, dieron una visión más amplia de lo que se quiere lograr y ofreció una ayuda para la selección de las herramientas a utilizar en el desarrollo de la aplicación. El estudio de las partes que conforman un Shapefile permitió una mejor concepción y entendimiento de la forma en la que se trabaja con dicho tipo de archivos.

2

CAPÍTULO 2.TENDENCIAS Y TECNOLOGÍAS

2.1 Introducción

Para el desarrollo de un software seguro y rentable es necesario documentar y controlar todo lo relacionado con la aplicación que se está desarrollando, para así disminuir o evitar los riesgos que puedan presentarse a lo largo de la construcción del mismo. Se hace necesario poseer conocimientos de varias herramientas ya sean de modelado o de programación, de elegir la metodología más adecuada que se adapte a las especificaciones propias del producto que se está desarrollando para lograr un mejor resultado.

En el presente capitulo se realizará un análisis de las principales tecnologías y tendencias actuales a usar en la construcción de la solución propuesta, así como la metodología de desarrollo a emplear, el lenguaje de programación y el lenguaje de modelado, entre otros aspectos.

2.2 El Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado o UML, por sus siglas en inglés, Unified Modeling Language es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software (James Rumbaugh, 2000). Es el más conocido y utilizado en la actualidad. UML ofrece un estándar para describir el sistema, además de poseer características y propiedades que son las que lo validan como el más aceptado en la actualidad.

UML prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos y describe la semántica esencial de lo que estos diagramas y símbolos significan. Es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que UML es un lenguaje, cuenta con reglas para combinar tales elementos.

UML permite modelar (analizar y diseñar) sistemas orientados a objetos, se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. Está respaldado por el OMG[9]. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de base datos y componentes reutilizables (James Rumbaugh, 2000).

2.2.1 Características de UML

Algunas de las propiedades de UML como lenguaje de modelado estándar son (James Rumbaugh, 2000):

- ✓ Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- ✓ Ampliamente utilizado por la industria desde su adopción por OMG.
- ✓ Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- ✓ Modela estructuras complejas.
- ✓ Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- ✓ Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- ✓ Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

[9] La OMG (Object Management Group) es una asociación sin fines de lucro formada por grandes corporaciones, muchas de ellas de la industria del software, como IBM, Apple, Sun Microsystems y HP.

UML implementa un lenguaje de modelado común para todos los desarrolladores, se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML es ahora un estándar, su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que ha sido diseñado para modelar todo tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro.

2.3 Metodología de Desarrollo de Software

En un proyecto de desarrollo de software la metodología define quién debe hacer qué, cuándo y cómo debe hacerse. No existe una metodología de software universal, las características de cada proyecto (equipo de desarrollo, recursos) exigen que el proceso sea configurable (EVA, 2011).

Una metodología puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo.

Una metodología de desarrollo de software no es más que completar una serie de tareas para obtener un producto final, donde los componentes de software deben pasar por distintas fases o etapas durante el desarrollo. Estas tareas pueden ser resueltas de distintas maneras con diferentes herramientas y técnicas. La metodología indica cómo hay que obtener los distintos productos parciales y finales.

Existen dos grupos fundamentales de metodologías, las metodologías tradicionales las cuales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas que se usarán. En el otro grupo se encuentran las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas.

Las metodologías tradicionales centran su atención en cumplir con un plan estratégico y organizado del proyecto, definido fundamentalmente en la fase inicial de su desarrollo. Las metodologías tradicionales (formales) se focalizan en documentación, planificación y procesos (Plantillas, técnicas

de administración, revisiones, etc.) (Roberth G. Figueroa, 2008). Dentro de las metodologías tradicionales se destacan el Rational Unified Process (RUP) y el Microsoft Solution Framework (MSF).

Las metodologías ágiles están especialmente orientadas para proyectos pequeños, constituyen una solución para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del software (José H. Canós, 2009).

En las metodologías ágiles se intenta ser lo más flexible posible, que el cliente pueda cambiar los requisitos cuando lo desee. Los individuos y sus interacciones son más importantes que los procesos y herramientas, dado que es mejor que el software funcione a tener una documentación exhaustiva. Las metodologías ágiles potencian la colaboración con el cliente más que la negociación del contrato ya que la respuesta ante el cambio tiene que ser más importante que el seguimiento de un plan.

2.3.1 Proceso Unificado de Desarrollo

Rational Unified Process (en lo adelante RUP) es una infraestructura flexible de desarrollo de software que proporciona prácticas recomendadas probadas y una arquitectura configurable. Es un proceso práctico (Rational).

RUP es un proceso de desarrollo dirigido por casos de uso, iterativo e incremental y centrado en la arquitectura. Es el resultado de la evolución e integración de diferentes metodologías de desarrollo, es un proceso pesado, donde una de sus debilidades es la gran cantidad de documentos que se genera a lo largo del desarrollo del software. Es uno de los procesos de desarrollo de software más generales de los que existen actualmente ya que está pensado para adaptarse a cualquier proyecto, incluso proyectos que van más allá del desarrollo de software (Molpecere, 2002).

2.3.1.1 Características y Beneficios

La característica de RUP de ser **Iterativo e Incremental** propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla

fundamentalmente algunos más que otros. El trabajo se divide en partes más cortas o en mini proyectos.

RUP es una metodología **Dirigida por Casos de Uso,** es un medio para determinar los requisitos funcionales de la aplicación y para definir los contenidos de las iteraciones. Cada iteración toma un conjunto de casos de uso y los desarrolla a través de las distintas fases.

Para entender el planteamiento de que RUP está **Centrado en la Arquitectura** hay que tener en cuenta que el concepto de arquitectura de software involucra los aspectos estáticos y dinámicos más significativos del sistema. La arquitectura de un sistema es la organización o estructura de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.

No existen dos proyectos de desarrollo de software que sean iguales. Cada uno tiene prioridades, requerimientos, y tecnologías muy diferentes. Sin embargo, en todos los proyectos, se debe minimizar el riesgo, garantizar la predictibilidad de los resultados y entregar software de calidad superior a tiempo. RUP, es una plataforma flexible de procesos de desarrollo de software que ayuda proveyendo guías consistentes y personalizadas de procesos para todo el equipo de proyecto (Soluciones y Propuestas Rational, 2010).

- ✓ **Las mejores prácticas más probadas de la industria**: Son las mejores prácticas de desarrollo adoptadas en cientos proyectos mundialmente y enseñadas como parte del currículo en cientos de universidades, la metodología RUP se convirtió rápidamente en el estándar de facto para el proceso de desarrollo en la industria de software.
- ✓ **Proceso hecho práctico**: Diferente a otras metodologías comerciales, la plataforma RUP hace que el proceso sea práctico con bases de conocimiento y guías para ayudar en el despegue de la planificación del proyecto, integrar rápidamente a los miembros del equipo y poner en acción el proceso personalizado.
- ✓ **Se adapta a las necesidades de los proyectos**: Solo la plataforma RUP proporciona un framework de proceso configurable que permite seleccionar e implantar los componentes

específicos de proceso necesarios para proporcionar un proceso consistente y customizado[10] para cada equipo y proyecto.

RUP es un proceso, que en su modelación define como principales elementos: trabajadores que conforman el (quién); actividades que representan el (cómo); artefactos que son los productos tangibles del proyecto que concretan el (qué); y el flujo de actividades que precisan la secuencia de actividades (cuándo). Además, está compuesto por nueve flujos de trabajo y cuatro fases, cada una de ellas con un número variable de iteraciones como se muestra en la figura.

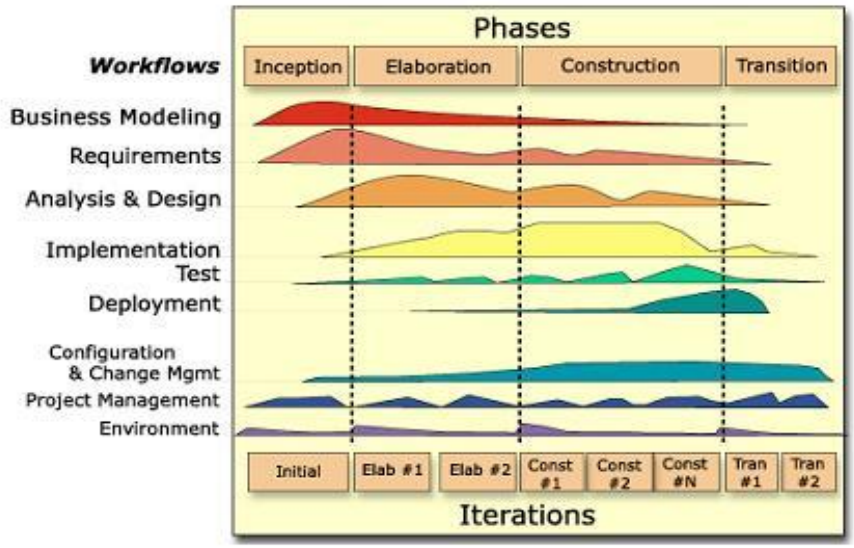


Figura 1 Ciclo de Vida de RUP

[10] Que se adapte a las preferencias de los usuarios.

2.3.2 Ágil UP (AUP)

El Agile UP (AUP) describe un enfoque simple y fácil de entender para el desarrollo de software usando técnicas y conceptos que aún se mantienen vigentes en RUP. Los enfoques aplican técnicas ágiles incluidas en el Desarrollo Dirigido por Pruebas (*Test Driven Desing* TDD), Desarrollo Dirigido por Modelado Ágil (*Agile Model Driven Development* AMDD), administración de cambios ágil, y refactorización de bases de datos para mejorar la productividad (2006).

Agile Up a sintetizado los flujos de RUP para poseer un ciclo de vida más robusto. La disciplina de Modelado abarca las disciplinas de Modelado del Negocio, de Requerimientos y de Análisis y Diseño de RUP. Las disciplinas de la Administración de la Configuración y Cambios ahora son la disciplina de la Administración de la Configuración. En el desarrollo ágil las actividades de administración de cambios son típicamente parte del esfuerzo de la administración de requerimientos, la cual es parte de la disciplina de Modelado.

Esta metodología ha emergido en la industria de desarrollo de software como una de las completas ya que está bien documentada. Está constituida por siete disciplinas: Modelo; Implementación; Prueba; Despliegue; Gestión de la Configuración; Gestión de Proyectos y Entorno. Y al ser una versión de RUP posee las mismas cuatro fases: Inicio; Elaboración; Construcción; Transición.

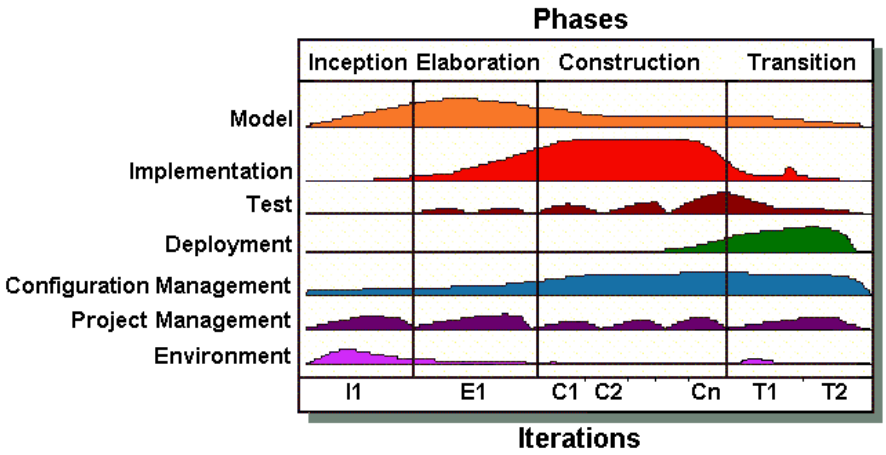


Figura 2 Ciclo de Vida de Agile UP

2.3.3 Selección de la metodología de desarrollo

La Tabla 2 muestra las principales diferencias de las metodologías ágiles con respecto a las tradicionales (no ágiles). Estas diferencias afectan no sólo al proceso en sí, sino también al contexto del equipo así como a su organización.

Tabla 2 Diferencias entre metodologías ágiles y no ágiles

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia a los cambios.
Impuestas internamente (por el equipo).	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles.	Más roles.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.

Como se ha demostrado, Agile UP es una metodología ligera pero muy abarcadora, ya que está basada fundamentalmente en los conceptos de RUP, la cual es una de las metodologías más generales que existen mundialmente. Lo más apreciable de Agile UP es precisamente que es más

rápida y no necesita toda la cantidad de artefactos que genera RUP, pero a la vez es muy eficiente, ya que sus fases y disciplinas están muy ligadas con las de esta robusta metodología.

2.4 ¿Aplicación web o Escritorio?

Aplicación de escritorio: Son aplicaciones que se ejecutan sobre un sistema operativo como Windows o Linux (de interfaz visual).

En una aplicación de escritorio normalmente no se inicia sesión por cada aplicación que se use, sólo se inicia sesión una vez cuando levanta el sistema operativo, asumiendo que se va a abrir una aplicación para ver nuestra lista de tareas:

- ✓ El usuario carga la aplicación.
- ✓ La aplicación (el código), se conecta a la base de datos y recupera la información del usuario.
- ✓ La aplicación muestra al usuario la información solicitada.

Aplicación web: Son todas las aplicaciones de software que se codifican en un lenguaje soportado por los navegadores web (HTML, Java Script, Java entre otros.) Los usuarios las pueden utilizar accediendo a un servidor web a través de Internet o de una intranet.

- ✓ No hay necesidad de instalar y distribuir el software a miles de usuarios.
- ✓ No ocupan espacio en el disco duro.
- ✓ Como la aplicación no se encuentra en el ordenador donde se utiliza, las tareas que realiza el software no consumen recursos.
- ✓ Permiten que el usuario acceda a los datos de modo interactivo garantizando que exista una comunicación activa entre el usuario y la información.
- ✓ Los virus no dañan los datos por medio del ordenador donde se utiliza porque éstos están guardados en el servidor de la aplicación.
- ✓ Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- ✓ Se puede usar desde cualquier sistema operativo, solo es necesario tener un navegador, por lo tanto es multiplataforma.

- ✓ Sus actualizaciones se hacen sin la necesidad de comprar el producto físicamente, sin descargas.

Las aplicaciones web son portables porque no dependen del ordenador donde se utilice (un ordenador de escritorio, un portátil, un móvil) porque se accede a través de una página web, sólo es necesario disponer de acceso a Internet o intranet. Para su ejecución desde cualquier sitio en cualquier navegador web simplemente basta con teclear su dirección URL, estas siglas indican en inglés (*Uniform Resource Locator*) y su traducción en español localizador uniforme de recurso.

Las aplicaciones web son populares y ofrecen mayores ventajas que las aplicaciones de escritorio en cuanto a movilidad, facilidad de soporte y mantenimiento. Por todas las ventajas antes mencionadas y expuestas para la solución propuesta se ha optado por una aplicación de tipo web.

2.5 Lenguajes de Programación

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo. Entre los lenguajes de programación más conocidos y utilizados en la actualidad se pueden citar: Delphi, Visual Basic, Python, Pascal, C#, Java, C++, PHP.

2.5.1 Lenguajes del lado del cliente

Los lenguajes del lado del cliente son aquellos que son interpretados directamente por el navegador y no requieren de un tratamiento previo. Los mismos, se encargan de garantizar el control sobre los formularios y sus principales eventos. A continuación se introducen algunos de ellos, los cuales se tomarán en cuenta a la hora de desarrollar la aplicación propuesta.

2.1.1.1 Lenguaje de Marcas de Hipertexto (HTML)

El Lenguaje de Marcas de Hipertexto (*HyperText Markup Language*, HTML) es, como su nombre lo indica, un lenguaje de marcado, o sea, es un lenguaje que incorpora junto con el texto, etiquetas o marcas que contienen información adicional acerca de la estructura de dicho texto. Es muy utilizado

en la construcción de páginas Web. HTML se escribe en forma de etiquetas, cada una con un significado específico, y rodeadas por corchetes angulares, que permiten describir la apariencia o estructura de un documento, es decir, se encargan de indicar al navegador donde debe colocar cada texto, imagen o video y la forma que tendrán estos al ser colocados en la página.

2.1.1.2 JavaScript

JavaScript es un lenguaje de programación usado principalmente para construir páginas Web dinámicas. Es un lenguaje sencillo y pensado para trabajar con rapidez, incluso para aquellas personas con poca experiencia en la programación, aprenderlo y practicarlo resulta bastante fácil. JavaScript proporciona dinamismo a la página Web, ya que permite incorporar efectos especiales a la misma, así como interactividad con los usuarios. Teniendo en cuenta que es un lenguaje de programación interpretado, no existe la necesidad de compilar los programas para ejecutarlos, es decir, que aquellos programas escritos en este lenguaje pueden ser probados directamente desde el navegador sin precisar procesos intermedios.

2.1.1.3 ExtJS

ExtJS es un framework basado en JavaScript Orientado a Objeto, uno de los más usados en el desarrollo de aplicaciones web en la parte del cliente, debido a que cuenta con una gran cantidad de componentes, clases y validaciones que pueden ser configuradas según las necesidades de cada sistema o del cliente.

Cuenta con una ayuda en inglés, la que tiene diversos ejemplos, de los cuales se pueden auxiliar los desarrolladores para realizar las aplicaciones. Tiene un diseñador de interfaces en línea, lo que facilita la creación de ventanas y componentes, garantizando un rápido desarrollo de sistemas web (ExtJS, 2010).

Ventajas:

- ✓ Existe un balance entre Cliente – Servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
- ✓ Comunicación asíncrona. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente note esta acción.
- ✓ Eficiencia de la red. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.

Desventajas:

- ✓ Necesita una plataforma para mostrar los componentes y hacer el render de la aplicación.
- ✓ El JavaScript no es tan rápido, sin embargo con la entrada de Google Chrome y el nuevo motor de Mozilla lo agilizan.
- ✓ Descargas lentas. Al ser aplicaciones grandes, especialmente porque cargan todo al inicio, hace que el tiempo de descarga sea mayor al de una aplicación web tradicional.
- ✓ Problemas con los motores de búsqueda. Los motores de búsqueda indexan el contenido web estático por lo que los textos cargados de manera dinámica no serán encontrados.
- ✓ Accesibilidad. Estas aplicaciones tienen problemas con los programas de accesibilidad pues, al igual que los motores de búsqueda, no trabajan bien con texto cargado dinámicamente.
- ✓ No se pueden usar fuera de línea. Por su naturaleza web estas aplicaciones no pueden ser usadas en el cliente como cualquier otra aplicación.

2.5.2 Lenguajes del lado del servidor

Son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él.

2.5.2.1 Lenguaje de programación PHP

PHP es el acrónimo de *Hipertext Pre-processor*, es una tecnología de código abierto, gratuita e independiente de plataforma, diseñada específicamente con el fin de permitir a los programadores la generación dinámica de páginas Web de forma fácil y rápida. Es un lenguaje sencillo, de sintaxis cómoda, que en su mayoría ha sido tomada de otros lenguajes de programación como C, Java y Perl, aunque posee características específicas de sí mismo.

PHP es un lenguaje interpretado de programación ampliamente usado que está orientado al desarrollo de aplicaciones web. Es un lenguaje sencillo y de sintaxis cómoda. PHP es un lenguaje para programar scripts del lado del servidor. Se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar y con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Tiene además una gran librería de funciones y mucha documentación. Existe un módulo de PHP para casi cualquier servidor web. Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado (Alvarez, 2009)

Figura 3 Funcionamiento de PHP

La figura muestra como PHP se ejecuta en el servidor, lo que permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado enviado al navegador. El resultado es normalmente una página HTML.

2.5.2.2 Ventajas de PHP

- ✓ Es un lenguaje multiplataforma.
- ✓ Tiene capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- ✓ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Permite las técnicas de Programación Orientada a Objetos.
- ✓ Tiene una biblioteca nativa de funciones sumamente amplia e incluida.
- ✓ No requiere definición de tipos de variables y tiene manejo de excepciones.

2.5.2.3 Lenguaje de programación ASP

ASP (*Active Server Pages*), es una tecnología propietaria de Microsoft. Se trata básicamente de un lenguaje de tratamiento de textos (scripts), basado en Basic, y que se denomina VBScript (Visual Basic Script). Se utiliza casi exclusivamente en los servidores Web de Microsoft (*Internet Information Server y Personal Web Server*). Los scripts ASP se ejecutan, por lo tanto, en el servidor y puede utilizarse conjuntamente con HTML y JavaScript para realizar tareas interactivas y en tiempo real con el cliente.

La principal ventaja de ASP es que hay un flujo constante de trabajo para estos desarrolladores. Sin embargo, se debe tomar esta información con cautela pues las tendencias actuales pronostican un decremento de los servidores de Microsoft y un aumento en los sistemas Linux. Además ASP es un sistema con nula portabilidad pues requiere necesariamente de un servidor Windows, con todas las implicaciones de alto costo, poca flexibilidad y escasa seguridad que estos equipos conllevan (Webnova, 2010).

2.5.3 Selección del lenguaje de programación a utilizar

Después de un estudio de los lenguajes de programación y analizadas sus características y en aras de potenciar la utilización de recursos libres, se propone la utilización del lenguaje de programación PHP para el desarrollo de la aplicación que se desea realizar. La razón por la cual ha sido

seleccionado es por la gran cantidad de ventajas que presenta sobre otros lenguajes de programación como ASP para el desarrollo de aplicaciones web.

Este es un lenguaje de código abierto, gratuito que brinda muchas ventajas, se integra con varios servidores web y puede ser utilizado en diferentes sistemas operativos. Es fácil de aprender y de utilizar, ya que permite la programación orientada a objetos brinda en su página oficial una gran cantidad de documentación y cuenta con una amplia biblioteca incluida de funciones, lo que hace más fácil el trabajo a desarrollar. Además de ser uno de los lenguajes más utilizados en nuestra universidad para el desarrollo de aplicaciones web.

2.6 Servidores

En el presente epígrafe se describen los servidores usados en el desarrollo de la aplicación web, presentando primeramente el servidor web Apache el cual será el encargado de publicar en línea la aplicación desarrollada, y por otra parte el servidor de base dados PostgreSQL, específicamente la extensión PostGIS que permite el manejo de datos espaciales.

Un servidor Web es un programa que implementa el protocolo HTTP con el propósito de transferir hipertextos, páginas Web o páginas HTML, figuras, formularios, entre otro conjunto de objetos y animaciones. Este se mantiene atento a las peticiones realizadas por el cliente, conocido como navegador Web y las responde adecuadamente. En dependencia del tipo de petición, el servidor Web busca una página Web o bien ejecuta un programa en el servidor. De cualquier manera, siempre devuelve algún tipo de resultado HTML al cliente o navegador que realizó la petición HTTP, es un protocolo de transferencia de hipertexto usado en cada transacción de la Web.

2.6.1 Servidor Web Apache

Apache es un servidor web que se encarga de atender las solicitudes realizadas por los usuarios mediante la Internet, brinda la funcionalidad de publicar sitios en línea. El proyecto de Apache es creado y actualizado por la Apache Software Fundation (ASF). Es una organización no lucrativa, ya que distribuyen el servidor apache de forma libre para todo aquel que desee usarlo. ASF es una comunidad descentralizada de desarrolladores que trabajan cada uno en sus propios proyectos de código abierto. Se fundó en 1999 a partir de la creación del Grupo Apache, el cual tenía como

objetivo la creación de un servidor Http de código abierto. Entre los objetivos principales de la ASF está la de dar protección legal a todos los voluntarios que trabajan en “Proyectos Apaches”. El Proyecto Apache es el origen de un grupo de licencias de código abierto, las cuales se denominan “Estilo Apache” (Apache - Web Server, 2010).

Es un servidor que posee disímiles características como (Linux-Ciberula, 2011):

Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.

- ✓ Es una tecnología gratuita de código abierto.
- ✓ Es un servidor altamente configurable de diseño modular.

Trabaja con gran cantidad de lenguajes de script como PHP teniendo todo el soporte que se necesita para tener páginas dinámicas.

- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

2.6.2 Servidor Web (IIS) de Microsoft

Internet Information Services (en lo adelante IIS) es un potente servidor Web, y aunque ofrece una estructura de gran fiabilidad, capacidad de manejo y escalabilidad para las aplicaciones Web, es un software propietario y sus servicios se limitan únicamente a los ordenadores que trabajan con Windows. Estos servicios proporcionan las funciones y herramientas que se necesitan a la hora de administrar de manera sencilla un servidor Web seguro. IIS se basa en varios módulos, los que le permiten procesar distintos tipos de páginas. Actualmente se conoce como el segundo servidor Web más utilizado en el mundo, detrás del anteriormente mencionado servidor Apache.

Los pilares clave de IIS son (Microsoft TechNet, 2010):

- ✓ Modelo de extensibilidad flexible para una eficaz personalización.
- ✓ Herramientas eficaces de diagnóstico y solución de problemas.
- ✓ Administración delegada.

- ✓ Seguridad mejorada y superficie expuesta a ataques reducida mediante personalización
- ✓ Administración de mantenimiento y aplicación integrada para servicios de Windows Communication Foundation (WCF).
- ✓ Herramientas de administración mejoradas.

2.6.3 Selección del servidor Web

Como servidor Web para el desarrollo de la aplicación propuesta se seleccionó el Apache, producto a que el mismo es reconocido por su escalabilidad, seguridad, y rendimiento, y se considera el servidor más utilizado en la actualidad a nivel mundial. Apache, a diferencia de IIS, es multiplataforma, por lo que está disponible para varios sistemas operativos. Es un servidor Web HTTP de código abierto que se adecua al lenguaje de programación que fue seleccionado para el desarrollo de la aplicación.

2.6.4 Servidor de Base de Datos PostgreSQL

PostgreSQL es compatible con ANSI SQL[11], y se provee de características que hace posible un diseño de software más complejo. PostgreSQL es extremadamente modular, soporta un gran número de tipos de datos, y es compatible hoy día un gran número de interfaces de programación. PostgreSQL está referenciado por los lenguajes de programación más importantes, incluyendo C, Perl, Python, Tel, Java, PHP, ODBC, JDBC, entre otros. PostgreSQL da soporte para la herencia y la seguridad de la capa de dispositivo de transportación de datos (SSL, Secure Sockets Layer). Además cumple completamente con las características ACID (acrónimo de Atomicity, Consistency, Isolation and Durability: Atomicidad, Consistencia, Aislamiento y Durabilidad en español) para realizar transacciones seguras, es multiplataforma, está disponible para 34 plataformas en su última versión estable.

[11] Lenguaje de consultas estructurado.

PostgreSQL es un SGBD relacional orientado a objetos y libre, basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre de este proyecto, que utiliza el lenguaje SQL. Al igual que otro grupo de proyectos de código abierto, su desarrollo no es manejado por una sola empresa, sino por una comunidad de desarrolladores y organizaciones comerciales que trabajan en su desarrollo. PostgreSQL utiliza Control de Concurrencia Multi-Versión con el fin de evitar bloqueos innecesarios, lo que le permite obtener una mejor respuesta en ambientes de grandes volúmenes.

Algunas de las características de PostgreSQL son (PostgreSQL, 2009):

- ✓ Es altamente adaptable a las necesidades del cliente.
- ✓ Corre en casi todos los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Windows.
- ✓ Documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios.
- ✓ Añade una estructura de datos array.
- ✓ Incorpora diversas funciones, tales como: manejo de fechas, geométricas, orientadas a operaciones con redes, entre otras.
- ✓ Posibilita la declaración de funciones propias, así como la definición de disparadores.
- ✓ Soporta el uso de índices, reglas y vistas.
- ✓ Contiene herencia entre tablas (no entre objetos, debido a que no existen), por tanto se considera un gestor de base de datos objeto-relacional.

2.6.5 PostGIS

PostGIS es una extensión al sistema de base de datos objeto-relacional PostgreSQL. Permite el uso de objetos *GIS* (Geographic Information Systems). PostGIS incluye soporte para funciones básicas para el análisis de objetos GIS.

Esta creado por Refractions Research Inc, como un proyecto de investigación de tecnologías de bases de datos espaciales. El código fuente está publicado bajo licencia GNU de PostgreSQL, la cual concede la libertad de usar, modificar y distribuir libremente el software. Con PostGIS podemos usar todos los objetos que aparecen en la especificación OpenGIS como puntos, líneas, polígonos, multilíneas, multipuntos, y colecciones geométricas.

2.7 Entorno Integrado de Desarrollo

2.7.1 Eclipse

Eclipse es un IDE de código abierto multiplataforma, desarrollado originalmente por IBM, aunque pasó luego a manos de la Fundación Eclipse, una organización independiente que fomenta la comunidad del código abierto. Actualmente existen otras herramientas similares, pero esta es reconocida por la calidad y facilidad que brinda a sus usuarios. Una de sus principales ventajas radica en la gran comunidad de usuarios que extiende constantemente su código fuente, creando así funcionalidades nuevas que hacen posible el desarrollo en diversos lenguajes de programación, no solo Web, sino también de escritorio.

Entre sus características principales se encuentran (Gomez, 2007):

- ✓ Dispone de un editor de texto con resaltado de sintaxis.
- ✓ La compilación es en tiempo real.
- ✓ Realiza pruebas unitarias.
- ✓ Mediante su arquitectura de plugins permite añadir soporte para lenguajes adicionales.

2.7.2 Netbeans

El NetBeans nació como un proyecto estudiantil en la República Checa en 1996. Su nombre original era Xelfi y fue el primer IDE para Java, escrito en Java. El emprendedor Roman Stanek, invierte en Xelfi y nace un negocio, mientras que Jarda Tulach, miembro del equipo original propone el nombre NetBeans (Cerda).

NetBeans es un IDE desarrollado por Sun Microsystems, de código abierto y multiplataforma. Permite diseñar aplicaciones de forma fácil con solo arrastrar objetos a la interfaz de un formulario. Es una plataforma pensada para escribir, compilar, depurar y ejecutar programas. NetBeans no solo permite el desarrollo de aplicaciones de escritorio, también permite el desarrollo de aplicaciones para la web y para dispositivos portátiles.

La programación en este IDE se realiza a través de componentes modulares o módulos. Las aplicaciones construidas a partir de módulos pueden ser extendidas ya que estos permiten ser desarrollados independientemente por otros desarrolladores de software, de ahí que sea una aplicación flexible/extensible.

El NetBeans permite desarrollar aplicaciones en lenguajes como Java, C/C++, PHP, Python, entre otros. Es un IDE multilenguaje completo y modular que le brinda facilidades al programador para el desarrollo intuitivo, a la vez que posee una gran cantidad de módulos o plugins que sirven de ayuda al implementador para hacer su trabajo más sencillo y eficiente.

2.7.3 Selección del entorno de desarrollo

Para la implementación del componente se utilizará NetBeans por las características antes expuestas, aunque también es válido destacar que existen otros IDEs disponibles y muy tentadores como Eclipse. Tanto NetBeans como Eclipse se integran con Visual Paradigm, pero NetBeans posee mejor completamiento de código, es necesario recalcar que NetBeans además de generar código, también genera documentación del mismo, lo que aporta una buena utilidad a ser tomada en cuenta en el momento de desarrollar el software.

2.8 Herramientas de Ingeniería de Software Asistidas por Computadoras

Las herramientas de Ingeniería de Software Asistidas por Computadoras (*Computer Aided Software Engineering*, CASE) no son más que aplicaciones informáticas que persiguen incrementar la productividad y calidad durante el desarrollo de un software mediante la reducción de tiempo, esfuerzo y dinero. Estas automatizan el dibujo de diagramas y ayudan en la creación de relaciones en la BD, provocando que el trabajo de diseño del software sea más fácil y agradable. El principal objetivo de estas herramientas es proporcionar un lenguaje para describir el sistema general, que sea lo suficientemente explícito para generar todos los programas necesarios.

2.8.1 Visual Paradigm

Visual Paradigm (en lo adelante VP) para UML es una herramienta 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 (Visual Paradigm Design Group, 2009).

Se integra con varios ambientes de desarrollo integrados (IDE) lo cual posibilita pasar del código al modelado y viceversa. Establece interoperabilidad con otras aplicaciones como el Visio y el Rational Rose. Disponible en múltiples lenguajes y plataformas: Microsoft Windows (98, 2000, XP, o Vista) Linux, Mac OS X, Solaris o Java (Visual Paradigm Design Group, 2009).

Esta herramienta posee unas características gráficas muy cómodas que facilitan la realización de los diagramas de modelado que sigue el estándar de UML y los utilizados para el diseño de la aplicación:

- ✓ Diagramas de clases.
- ✓ Diagramas de Casos de Uso.
- ✓ Diagramas de Secuencia.
- ✓ Diagramas de Componentes.

VP proporciona el código y compatibilidad hasta con 10 lenguajes , VP-UML[12] es apoyado por un conjunto de idiomas tanto en las generaciones del código como en la ingeniería inversa, por mencionar algunos ejemplos, los cuales tienen la capacidad de soporte; podríamos hablar de java, C++, Cobra, PHP, XML, Schema, Ada, Python.

VP ofrece entorno de creación de diagramas para UML; Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad; Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación; Capacidades de ingeniería directa (versión profesional) e inversa; Modelo y código que permanece sincronizado en todo el ciclo de desarrollo;

[12] Visual Paradigm for UML.

Disponibilidad de múltiples versiones, para cada necesidad; Disponibilidad de integrarse en los principales IDEs[13] (Sierra, 2010).

Por su alta interoperabilidad VP sin importar el IDE ofrece los beneficios siguientes (Visual Paradigm Design Group, 2009):

- ✓ Navegación intuitiva entre código y el modelo.
- ✓ Poderoso generador de documentos y reportes UML, PDF/HTML/MS Word.
- ✓ Demanda en tiempo real, modelo incremental de viaje redondo y sincronización de código fuente.
- ✓ Superior entorno de modelado visual.
- ✓ Soporte completo de notaciones UML.
- ✓ Diagramas de diseño automático sofisticado.

2.8.2 Rational Rose Enterprise Edition

Rational Rose Enterprise Edition es el producto más completo de la familia Rational Rose, incluye, al igual que el resto de los productos de esta familia, soporte UML. Es una herramienta propietaria de diseño de software, que se encarga de llevar a cabo la automatización de los sistemas para la posterior generación de código, o sea, realiza la modelación visual y construcción de componentes de dichos sistemas.

Incluye las siguientes características (Pérez, 2007):

- ✓ Permite el modelado en UML para diseñar BD, que integra los requisitos de datos y aplicaciones mediante diseños lógicos y analíticos.

[13] Varios Entornos Integrados de Desarrollo

- ✓ Proporciona al equipo de desarrollo un lenguaje común de modelado, que facilita la rápida creación de un software de calidad.
- ✓ Incluye funciones de visualización, modelado y herramientas para desarrollar aplicaciones Web.
- ✓ Posibilita la integración con otras herramientas de desarrollo de IBM Rational.
- ✓ Consiente la generación de código a partir de modelos Ada, ANSI C++, C++, CORBA, Java/J2EE, Visual C++ y Visual Basic.
- ✓ Capacidad de análisis de calidad de código.

2.8.3 Selección de la herramienta CASE

Entre las herramientas CASE analizadas se seleccionó para la construcción de la aplicación a Visual Paradigm, ya que constituye una herramienta factible para trabajar con el lenguaje de modelado UML. Resulta de gran ayuda a la hora de desarrollar aplicaciones de calidad en menor tiempo y costo. Es una herramienta de fácil instalación y uso, lo que apoya la comunicación del equipo de desarrollo y contribuye a alcanzar un producto de excelencia. Entre sus características se destacan que es una herramienta multiplataforma y colaborativa, además de proporcionar la integración con los principales entornos de desarrollo integrados, como por ejemplo Eclipse y NetBeans.

2.9 Conclusiones

Con el análisis realizado a través de este capítulo se ha logrado la selección de la metodología que regirá el desarrollo del componente, así como las herramientas y tecnologías que darán solución al mismo. Se concluye que la metodología a usar será Agile UP por estar muy complementada con RUP, además por ser una aplicación pequeña con pocos desarrolladores y contar para la implementación de la misma con un tiempo limitado; UML como lenguaje de modelado para describir todos los procesos en general y Visual Paradigm será la herramienta CASE para modelar la aplicación y generar todos los diagramas pertinentes. El lenguaje de programación escogido es PHP por las potencialidades que presenta y como IDE de desarrollo NetBeans por las facilidades de completamiento de código.

3

CAPÍTULO 3. PRESENTACIÓN DE LA SOLUCIÓN

3.1 Introducción

El objetivo del presente trabajo es realizar una aplicación web que transforme el formato ESRI Shapefile a PostgreSQL. En el actual capítulo se especifican cuales son las funcionalidades del software, definiéndose los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF); partiendo de estos requisitos se obtienen y describen los casos de uso, los cuales guían todo el proceso llevado a cabo para la solución del problema. Se presenta un modelo de dominio en aras de lograr la comprensión de los conceptos asociados al entorno donde trabajará el sistema. Se realizará una descripción de la solución propuesta acorde con la metodología de desarrollo escogida. El propósito es asegurar la calidad del producto final a partir de una guía adecuada del proceso de desarrollo del software.

3.2 Modelo de dominio

Como no existe un negocio bien definido, y no se puede precisar la estructura de los procesos del negocio, se empleó un modelo de dominio. El modelo de dominio es un medio para comprender el sector del negocio al cual el sistema va a servir, con el objetivo de mostrar al usuario los principales conceptos que se tratan en el entorno, logrando un mejor entendimiento del sistema. Esto ayuda a los usuarios, clientes, desarrolladores e interesados a utilizar un vocabulario común para poder entender el contexto en que se ubica el sistema.

El modelo de dominio es la vista estática del dominio de la aplicación, así como los conceptos internos inventados como parte de la implementación de la aplicación. Esta visión es estática porque no describe el comportamiento del sistema dependiente del tiempo, que se describe en otras vistas. Los componentes principales de la vista estática son las clases y sus relaciones: asociación, generalización, y varias clases de dependencia, tales como realización

y uso. Una clase es la descripción de un concepto del dominio de la aplicación o de la solución de la aplicación. Las clases son el centro alrededor del cual se organiza la vista de clases; otros elementos pertenecen o se unen a las clases. La visión estática se exhibe en los diagramas de clases, llamados así porque su objetivo principal es la descripción de clases (James Rumbaugh, 2000).

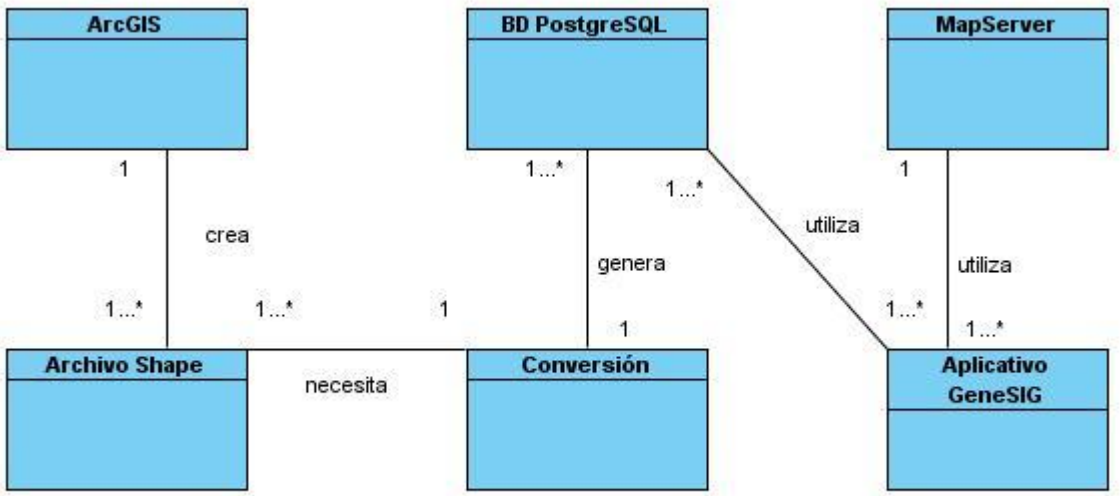


Figura 4 Modelo de Dominio

3.2.1 Glosario de términos del Modelo de Dominio

ArcGIS: ArcGis es una serie integrada de software de Sistemas de Información Geográfica que trabaja como un motor compilador de información geográfica alfanumérica (Bases de Datos) y gráfica (Mapas).

Archivo Shape: Es un formato que almacenan tipos de datos geométricos de forma vectorial como puntos, líneas y polígonos usados por los SIG.

Conversión: Proceso mediante el cual se transformaran los archivos shape a PostgreSQL.

BD PostgreSQL: Gestor de base datos que utiliza la aplicación.

Aplicativos Genesig: Estructura que aglutina la creación de los SIG.

MapServer: Es una plataforma de código abierto para la publicación de datos espaciales y aplicaciones de cartografía interactiva para la web.

3.3 Especificación de requisitos

Según el estándar 1233 de la IEEE: Guía para el desarrollo de Especificaciones de Requerimientos de Sistemas, un requisito se define como (IEEE, 1998):

- ✓ 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.

Los requisitos de software son las características y propiedades que debe cumplir el sistema. Todas las ideas que se tengan acerca de lo que debe hacer el sistema, tanto de los clientes, usuarios y miembros del equipo de proyecto deben ser analizadas como candidatas a requisitos.

3.3.1 Requisitos Funcionales

El sistema debe cumplir con una serie de requisitos funcionales (capacidades o condiciones) para satisfacer las funcionalidades requeridas en la aplicación.

RF1. Crear conexión a PostgreSQL.
Descripción: La aplicación debe permitir insertar los valores necesarios para definir una nueva conexión a una Base de Datos PostgreSQL que se podrá seleccionar posteriormente para establecer la conexión.

RF2. Probar conexión a PostgreSQL.
Descripción: La aplicación debe permitir establecer una conexión de prueba luego de insertar los valores necesarios para definir una nueva conexión a una Base de Datos PostgreSQL.

RF3. Editar conexión PostgreSQL.

Descripción: La aplicación debe permitir actualizar los valores de una conexión a una Base de Datos PostgreSQL definida anteriormente.

RF4. Eliminar conexión PostgreSQL.
Descripción: La aplicación debe permitir eliminar una conexión a una Base de Datos PostgreSQL definida anteriormente.

RF5. Establecer conexión con una Base de Datos PostgreSQL.
Descripción: La aplicación debe permitir establecer una conexión con una Base de Datos PostgreSQL definida anteriormente.

RF6. Añadir múltiples archivos shape.
Descripción: La aplicación debe permitir seleccionar múltiples archivos shape mediante un cuadro de diálogo para luego ser convertidos a PostgreSQL.

RF7. Configurar opciones de conversión.
Descripción: La aplicación debe permitir configurar las opciones de conversión de los archivos en formato ESRI Shapefile a PostgreSQL.

RF8. Convertir archivos añadidos a PostgreSQL.
Descripción: La aplicación debe realizar la conversión de los archivos en formato ESRI Shapefile a PostgreSQL.

3.3.2 Requisitos No Funcionales

Requisito que especifica propiedades del sistema, como restricciones del entorno o de implementación, rendimiento, dependencias de la plataforma, mantenibilidad, extensibilidad o fiabilidad. Requisito que especifica restricciones físicas sobre un requisito funcional (James Rumbaugh, 2000).

Los Requisitos No Funcionales son aquellos que describen las cualidades, características y propiedades del producto.

Requisitos de Usabilidad

- ✓ El sistema podrá ser usado por personas con conocimientos básicos de informática.
- ✓ Usará botones y formularios de un tamaño adecuado para la vista del usuario.

- ✓ Se proporcionará un diseño ordenado.

Requisito de Fiabilidad

El factor fiabilidad del sistema se maneja mediante una atención rápida y esmerada de los errores producidos en tiempo de ejecución.

- ✓ Se controlará ofreciendo oportunamente los mensajes e indicaciones pertinentes al usuario.
- ✓ Se informa la presencia de algún error de forma inmediata luego de haberse producido.

Restricciones de diseño

- ✓ La base de datos será desarrollada en PostgreSQL.
- ✓ Para garantizar el desarrollo de la aplicación se utilizó como metodología el Proceso Unificado Ágil (AUP).
- ✓ Se utilizó para realizar los modelos del sistema, UML y como herramienta de apoyo a este lenguaje de modelación Visual Paradigm.

Requisito de Apariencia o interfaz externa

El diseño del sistema debe estar enfocado en una interfaz sencilla y funcional, de fácil operación.

Interfaces de usuario

El sistema debe:

- ✓ Tener una apariencia profesional y un diseño gráfico sencillo.
- ✓ Ser intuitivo.

Interfaces de Hardware

Para las PCs clientes:

- ✓ Se requiere tengan tarjeta de red.
- ✓ Al menos 500 MB de memoria RAM.

- ✓ Procesador 1 GHz como mínimo.

Para los servidores:

- ✓ Se requiere tarjeta de red.
- ✓ El Servidor de BD tenga como mínimo 2GB de RAM y 10GB de disco duro.
- ✓ Procesador 3 GHz como mínimo.

Interfaces de Software

Para las PCs clientes:

- ✓ Tener instalado un navegador Web, por ejemplo Mozilla Firefox, Internet Explorer o Chrome.
- ✓ Tener instalado en el navegador Flash Player 9 o superior.

Para los servidores:

- ✓ Instalado Apache como servidor Web.
- ✓ Gestor de base de datos PostgreSQL con la extensión de PostGIS instalada.

3.4 Descripción del sistema propuesto

Teniendo en cuenta el modelo de dominio, los requisitos planteados anteriormente y los objetivos de esta investigación, se propone la elaboración de un convertidor de archivos ESRI Shapefile a PostgreSQL. Por otra parte se considera la existencia de un solo rol, el usuario del sistema, el cual se beneficiara de los resultados del componente.

3.4.1 Definición de los casos de usos del sistema

Actor	Descripción
Usuario	Cualquier usuario que utiliza las funcionalidades del sistema.

A continuación se presentan los casos de uso determinados para satisfacer los requisitos funcionales del sistema:

CU-1	**Convertir Shapefile**
Actor	Usuario
Descripción	El usuario proporciona los archivos que desea convertir y se realiza la conversión.
Referencia	RF6, RF7, RF8
CU-2	**Gestionar Conexiones PostgreSQL**
Actor	Usuario
Descripción	El usuario puede crear una nueva conexión a la BD, editar o eliminar la ya existente, para guardar los archivos Shapefile convertidos.
Referencia	RF1, RF2, RF3, RF4, RF5

3.4.2 Diagrama de casos de Uso del Sistema

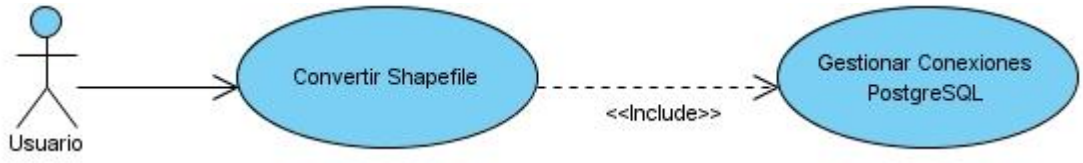


Figura 5 Diagrama de Casos de Uso del Sistema

3.4.3 Descripción Textual de los Casos de Uso

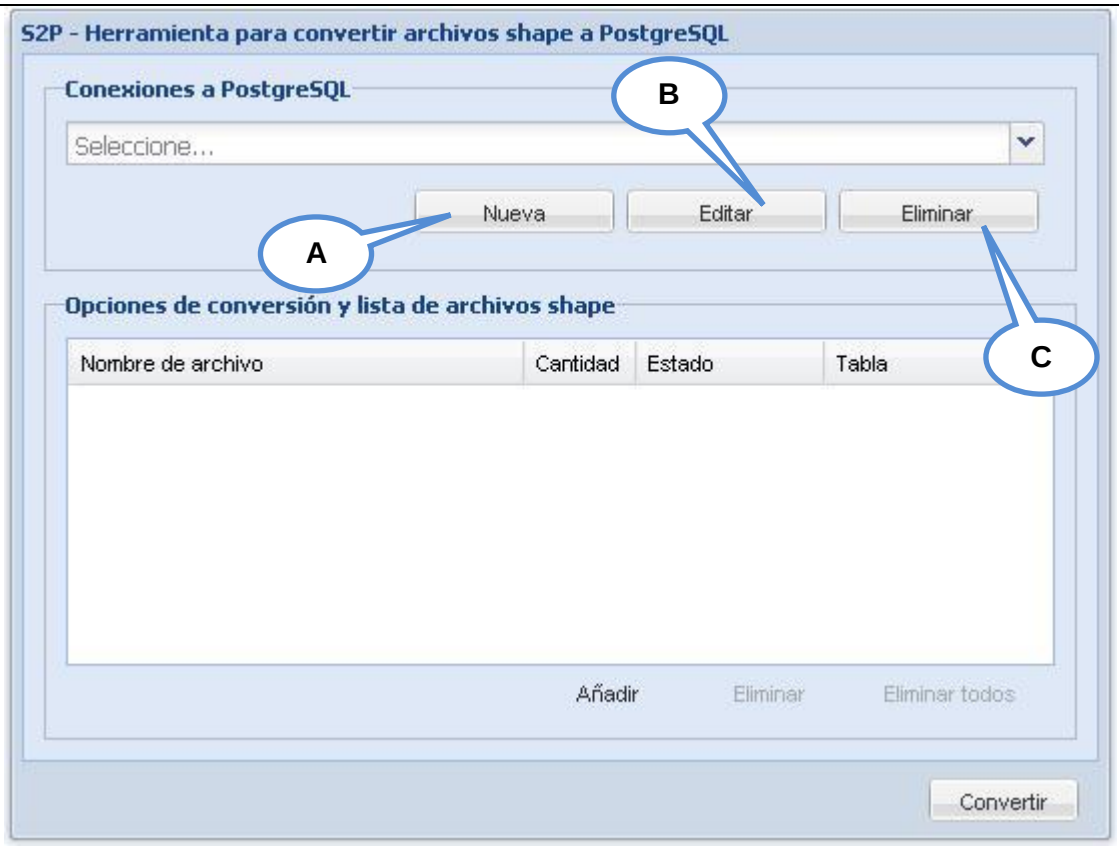
En la descripción textual de los casos de uso se detallan las actividades que se realizan dentro de los procesos, dando así un mayor entendimiento y seguimiento de los mismos.

Tabla 3 Descripción Textual CU Gestionar Conexiones PostgreSQL

Caso de Uso:	**Gestionar Conexiones PostgreSQL**

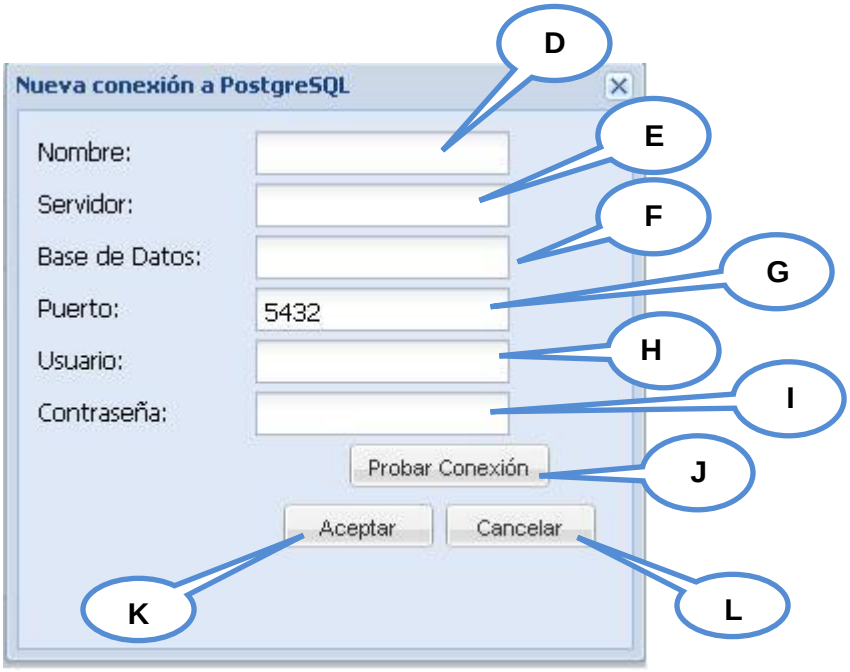
Actores:	Usuario
Propósito	Este caso de uso se lleva a cabo con el objetivo de establecer la conexión a la BD donde se guardaran los archivos Shapefile convertidos.
Resumen:	El caso de uso se inicia cuando el usuario escoge la opción de nueva, editar o eliminar una conexión a la BD y el sistema muestra un formulario para que se introduzcan los datos para realizar la conexión en caso de nueva o editar, en caso de eliminar se eliminaría una conexión antes hecha.
Referencias	RF1, RF2, RF3, RF4, RF5
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona una de las opciones siguientes: a) Nueva conexión a la BD (A). b) Editar conexión a la BD (B). c) Eliminar conexión a la BD (C).	**2.** El sistema realiza la operación seleccionada por el usuario. ✓ Si selecciona la opción a), ir a la sección: Nueva conexión a la BD. ✓ Si selecciona la opción a), ir a la sección: Editar conexión a la BD. ✓ Si selecciona la opción c), ir a la sección: Eliminar conexión a la BD.



Sección "Nueva conexión a la BD"	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra la ventana para crear una nueva conexión a la BD con los campos 'Nombre' (D), 'Servidor' (E), 'Base de Datos '(F), 'Puerto' (G), 'Usuario' (H), 'Contraseña' (I), y los botones 'Probar Conexión' (J), 'Aceptar' (K), 'Cancelar' (L).
2. El usuario introduce los datos para crear la conexión a la BD. **3**. El usuario presiona el botón	**4.** El sistema verifica los datos introducidos. **5.** El sistema establece una conexión a la BD. **6.** El sistema almacena los datos introducidos y cierra la

Aceptar (I)	ventana, finalizando el caso de uso.



Flujo Alternos	
Acción del Actor	**Respuesta del Sistema**
2.1 El usuario presiona el botón Probar Conexión (J). **2.2** El usuario presiona el botón Aceptar y regresa al paso 3 del flujo normal de eventos.	**2.2** El sistema muestra un mensaje de si es correcta o no la conexión a la BD.

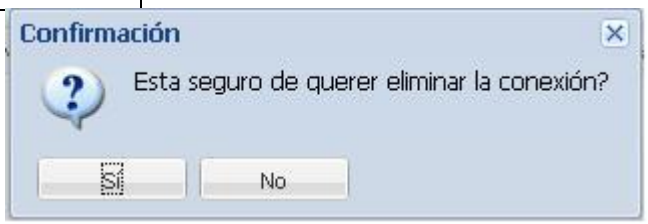
Sección "Editar conexión a la BD"	
1. El usuario escoge la BD a editar.	**2.** El sistema muestra la opción de editar la conexión a la BD.

	Ver Sección Crear conexión a la BD.
3. El usuario introduce los datos que desea cambiar. **4** .El usuario presiona el botón Aceptar. Finalizando el caso de uso	**5.** El sistema verifica los datos y actualizada la BD.

Ver Prototipo de Interfaz de "Nueva conexión a la BD"

Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
3.1 El usuario presiona el botón Probar Conexión.	**3.2**. El sistema muestra un mensaje de si es correcta o no la conexión a la BD.

Sección "Eliminar conexión a la BD"	
	1. El sistema muestra la opción de escoger la BD que desee eliminar.
2. El usuario escoge la BD que desea eliminar. **3.** El usuario presiona el botón eliminar. **4.** El usuario escoge la opción mediante un cuadro de confirmación	**5.** El sistema elimina la conexión a la BD. Finaliza el caso de uso.



Pos condiciones	Se logra crear, editar o eliminar la conexión a la BD.

Tabla 4 Descripción Textual CU Convertir Shapefile

Caso de Uso:	**Convertir Shapefile**
Actores:	Usuario
Propósito	Este caso de uso se lleva a cabo con el objetivo de convertir los archivos Shapefile.
Resumen:	El caso de uso se inicia cuando el usuario carga los archivos Shapefile que desea convertir y termina cuando el sistema guarda en la base de datos los archivos antes convertidos.
CU asociados	Caso de Uso incluido Gestionar Conexiones PostgreSQL.
Referencias	RF6, RF7, RF8
Prioridad	Crítico
Precondición	Tiene que estar establecida la conexión con la BD PostgreSQL.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el usuario presiona el botón Seleccionar (M).	**2.** El sistema muestra la opción para cargar archivos.
3. El usuario selecciona los archivos que desea convertir.	**4.** El sistema sube los archivos al servidor (N).
5. El usuario escoge una de las siguientes opciones: Eliminar (P), Eliminar todos (Q), Editar nombre de la tabla (R) **6.** El usuario ejecuta la opción	**7.** El sistema convierte los archivos a un script SQL y los guarda en la BD, finalizando el caso de uso.

convertir (O).	
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	5.1 El sistema muestra un mensaje de que debe seleccionar una conexión con la BD.

Pos condiciones	Se logra realizar la conversión de los archivos y quedan almacenados en la BD.

3.5 Conclusiones

En el presente capítulo quedaron expuestos los principales conceptos del entorno en el que se encuentra enmarcado el sistema, realizándose un modelo de domino para una mejor comprensión del medio donde coexiste el problema, definiéndose a partir de este modelo, conceptos, entidades y sus relaciones. Mediante un profundo estudio se identificaron los requisitos que el sistema deberá cumplir, clasificándolos en Requisitos Funcionales (RF) y Requisitos No Funcionales (RNF) con el objetivo de satisfacer las restricciones y necesidades de la aplicación. Se ha determinado que se cuenta con dos procesos principales los cuales contienen todos los requisitos encontrados, ambos procesos se han descrito como Casos de Uso, donde los dos CU existentes poseen prioridad crítica, especificando en detalle los pasos que los componen.

4

CAPÍTULO 4. CONSTRUCCIÓN DE LA SOLUCIÓN

4.1 Introducción

El presente capítulo presenta el diseño y la implementación del sistema realizado, así como el diagrama de despliegue y de componentes, describiéndose la construcción de la solución propuesta, así como de los patrones utilizados en la elaboración de la aplicación. Los requisitos se traducen a una especificación que describen cómo implementar el sistema, donde partiendo de los mismos, se realiza el diagrama de clases del diseño. Se exponen las pruebas que se le aplicarán al sistema para verificar su funcionamiento.

4.2 Descripción de la arquitectura

La arquitectura del software es el proceso de definición de una solución estructurada que cumple con los requisitos técnicos y operativos, optimizando al mismo tiempo atributos de calidad tales como: rendimiento, seguridad y manejabilidad. Se trata de una serie de decisiones basadas en una amplia gama de factores, donde cada una de estas decisiones puede tener un impacto considerable en la calidad, rendimiento, facilidad de mantenimiento y en general, el éxito de la aplicación (Microsoft Corporation, 2009).

4.2.1 Arquitectura en capas

Este patrón define cómo organizar el modelo de diseño en capas, que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además, ayuda a identificar qué puede reutilizarse.

4.2.1.1 Arquitectura en tres capas

Para el desarrollo de la aplicación se utilizará el patrón arquitectónico tres capas, las cuales se dividen en tres capas lógicas distintas, cada una de ellas con un grupo de interfaces perfectamente definidas. La primera capa se denomina capa de presentación y normalmente consiste en una interfaz gráfica de usuario. La capa intermedia, consiste en la lógica del negocio, y la tercera capa, la capa de datos, contiene los datos necesarios para la aplicación.

- ✓ **Capa de presentación:** También se le denomina "capa de usuario", presenta el sistema al usuario, le comunica la información y captura la información del usuario. Debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.
- ✓ **Capa de negocio:** Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Además, residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. 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 BD almacenar o recuperar datos de él.
- ✓ **Capa de datos:** Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de BD que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

4.3 Patrones

Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno y ofrece una solución a ese problema, en tal sentido es posible usar esa solución millones de veces, sin tener que hacer lo mismo dos veces (A Patern Languaje, 1977). Los Patrones de Diseño determinan cómo construir software, cómo utilizar las clases y los objetos de forma conocida.

4.3.1 Patrones GOF

Singleton: El patrón *Singleton* o Instancia única garantiza que una clase sólo tenga una instancia y proporciona un punto de acceso global a ésta instancia. Este patrón se pone de manifiesto en la clase *ConnectionMgr*.

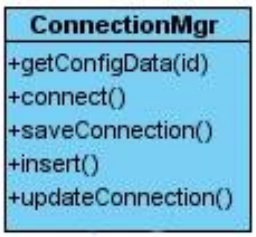


Figura 6 Clase que implementa el patrón Singleton

Ventajas:

- ✓ Acceso controlado a la única instancia. Puede tener un control estricto sobre cómo y cuándo acceden los clientes a la instancia.
- ✓ Espacio de nombres reducido. El patrón *Singleton* es una mejora sobre las variables globales.
- ✓ Permite el refinamiento de operaciones y la representación.

4.3.2 Patrones GRASP

GRASP[14] (patrones generales de software para asignar responsabilidades), describen los principios fundamentales de la asignación de responsabilidades a objetos. A continuación una breve descripción de los patrones GRASP utilizados en la aplicación.

Experto

[14]General Responsibility Asignment Software Patterns.

El patrón *Experto* plantea que cada objeto realiza la funcionalidad de acuerdo a la información que domina, la cuestión a la hora de diseñar es asignar responsabilidades a la clase que mayor información posee para cumplir con dicha tarea.

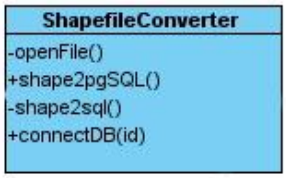


Figura 7 Clase que implementa el patrón Experto

Creador

El patrón *Creador* guía la asignación de responsabilidades relacionadas con la creación de objetos, frecuentemente usado en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. Este patrón se pone de manifiesto en el paquete de clases JS, que contiene las clases Ext y SwfUpload.

Patrón Bajo Acoplamiento

El *Bajo Acoplamiento* soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. Este patrón se pone en evidencia en la clase Ext contenida en el paquete JS.

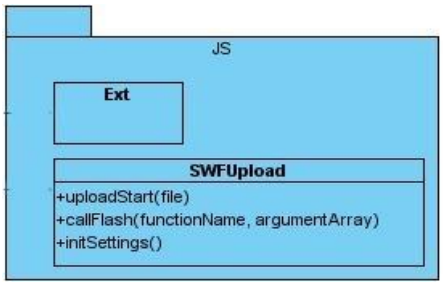


Figura 8 Clases que implementan los patrones Creador y Bajo Acoplamiento

Beneficios que propicia este patrón:

- ✓ No se afectan por cambios de otros componentes.
- ✓ Fáciles de entender por separado.
- ✓ Fáciles de reutilizar.

Patrón Alta Cohesión

Estamos en presencia de una alta cohesión funcional cuando los elementos de un componente (clase) colaboran para producir algún comportamiento bien definido. Es decir cuando varias clases colaboran entre sí, para dar una solución en conjunto. Este patrón se evidencia en el conjunto de clases File, que entre todas colaboran para permitir el trabajo con ficheros.

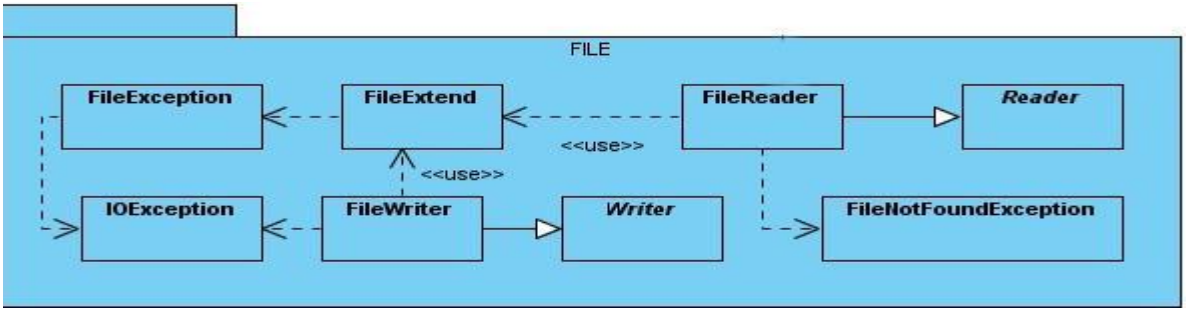


Figura 9 Clase que implementa el patrón Alta Cohesión

4.4 Diagrama de clases del diseño

El diagrama de clases de diseño de un sistema refleja los detalles que tienen que ver más concretamente con la implementación, mostrando la estructura interna del sistema, en cuanto a la información concerniente a cada una de las clases que forman el mismo, atributos y sus tipos de datos, métodos y sus tipos de datos de retorno y además brinda una representación gráfica de las relaciones entre todas las clases del sistema. Un diagrama de clases del diseño muestra un conjunto de interfaces, colaboraciones y sus relaciones, principalmente se utilizan para modelar la vista de diseño estática de un sistema. En la Figura 10 Diagrama de Clases del Diseño CU Convertir Shapefile, se muestra el diagrama de clases del diseño del caso de uso Convertir Shapefile.

Con el objetivo de simplificar el diagrama para lograr un mejor entendimiento del mismo, los paquetes PGSQLMGR y FILE se detallan en el **¡Error! No se encuentra el origen de la referencia.**y **¡Error! No se encuentra el origen de la referencia.**respectivamente.

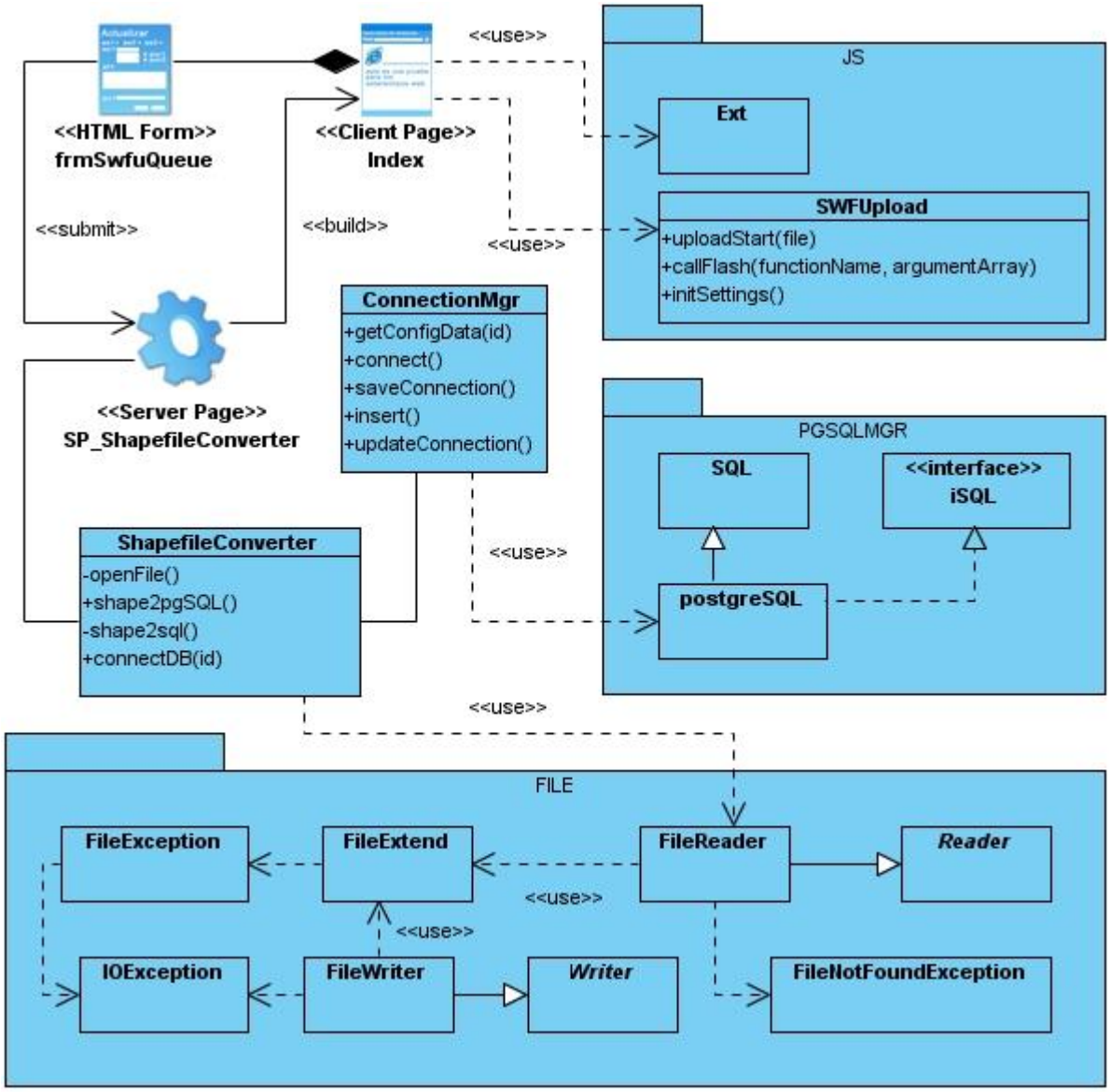


Figura 10 Diagrama de Clases del Diseño CU Convertir Shapefile

4.4.1 Diseño de la Base de Datos

A continuación se muestra el diagrama entidad-relación de la base de datos de la aplicación, donde se guardaran todas las conexiones de los usuarios a distintas BD.

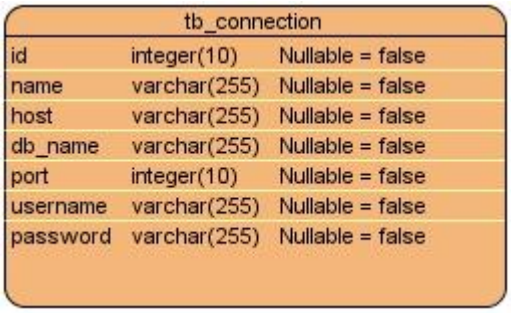


Figura 11 Diagrama entidad-relación de la BD del sistema

4.5 Diagrama de Interacción

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

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

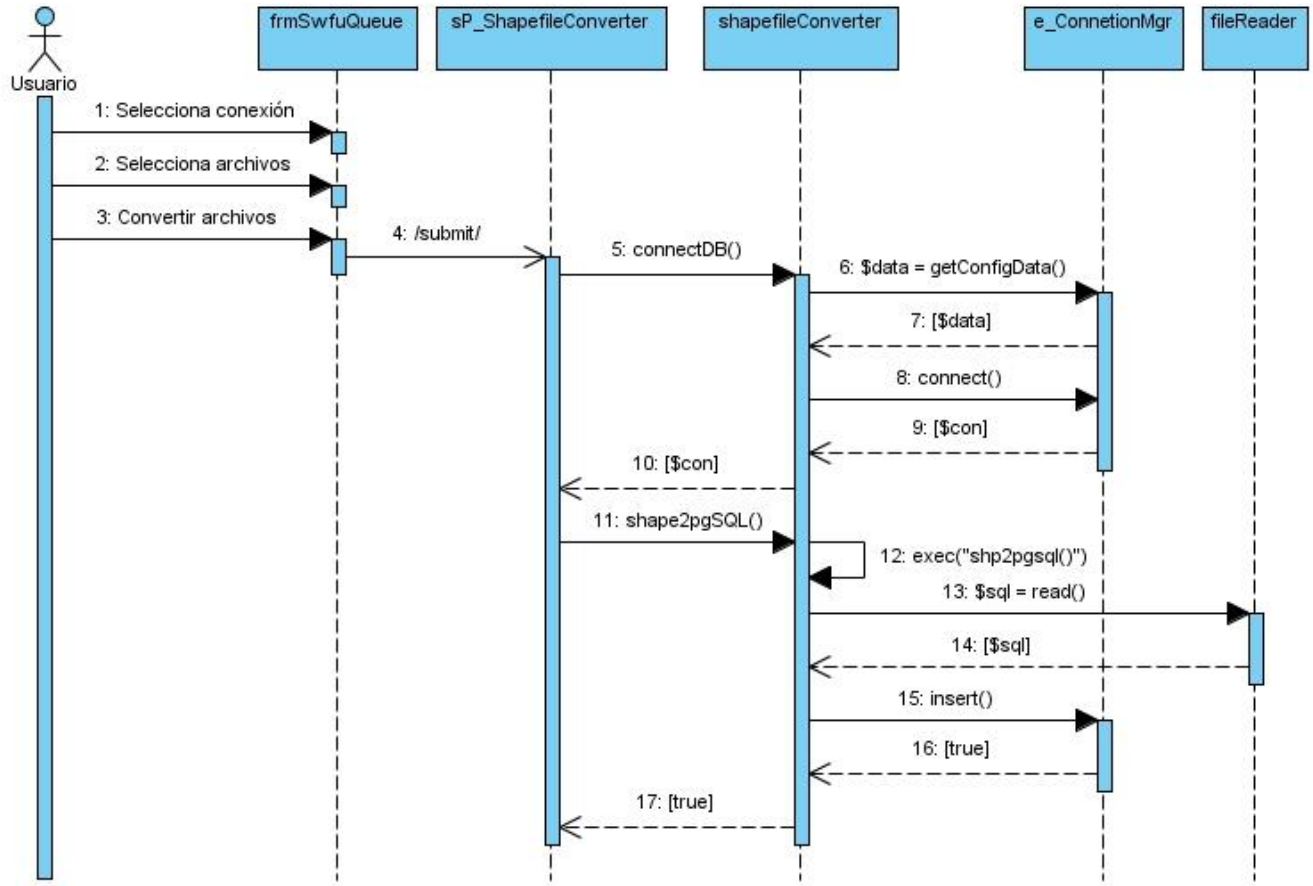


Figura 12 Diagrama de secuencia CU Convertir Shapefile

4.6 Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física de un sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. En esta vista se realiza una representación gráfica de los nodos físicos en los que estará desplegado el sistema propuesto y la comunicación entre ellos.

Los diagramas de despliegue muestran la configuración en funcionamiento del sistema, incluyendo su hardware y su software. Seguidamente se muestra el diagrama de despliegue de la aplicación.

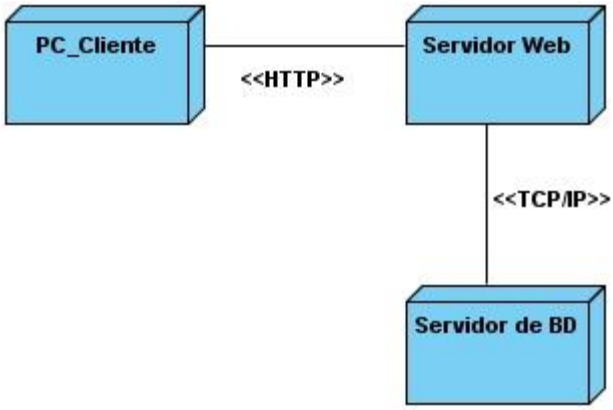


Figura 13 Diagrama de despliegue

4.7 Modelo de Implementación

La implementación empieza con el resultado del diseño y se implementa en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. El flujo de trabajo de implementación detalla cómo los elementos del modelo del diseño se implementan en términos de componentes y representa cómo se organizan en el modelo de despliegue. Permite obtener los diagramas de despliegue y componentes los cuales son artefactos que conforman lo que se conoce como un modelo de implementación. A continuación se muestran los diagramas de componentes y de despliegue de sistema.

4.7.1 Diagrama de componentes

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. El Diagrama de Componente muestra la relación entre los componentes del software, sus dependencias, comunicaciones y localización. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables o paquetes.

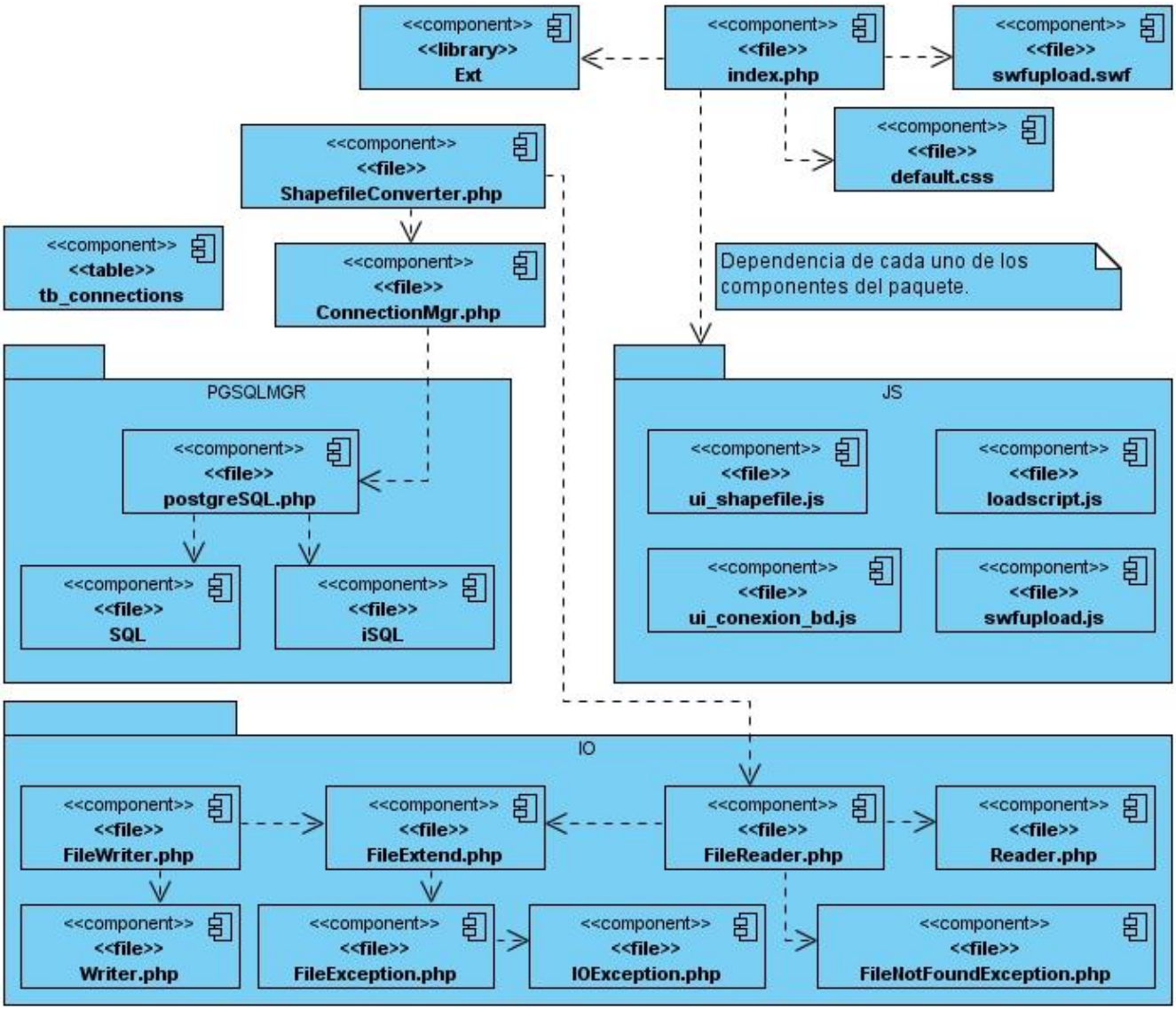


Figura 14 Diagrama de componentes

4.8 Pruebas

Las pruebas son actividades en las cuales un sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos. Los resultados son observados y registrados, y se realiza una evaluación del sistema. Las pruebas verifican los resultados de la implementación del sistema. El modelo de prueba describe las formas en que han de ser probados aspectos específicos del sistema.

El objetivo de realizar este tipo de prueba al sistema, es detectar el incorrecto o incompleto funcionamiento de este, así como los errores de interfaces, rendimiento, errores de inicialización y terminación.

4.8.1 Pruebas de Caja Negra

Para probar un sistema mediante el método de caja negra pueden ser utilizadas diferentes técnicas, una de las más prácticas es la técnica de la Partición de Equivalencia, que divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del sistema. Esta técnica brinda la posibilidad de examinar las entradas del sistema con valores válidos e inválidos, descubriendo de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico.

Las pruebas de Caja Negra permiten encontrar:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de Interfaz.
- ✓ Errores en estructura de datos o acceso a Base Datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y de terminación.

A continuación se muestra el caso de prueba para el caso de uso Gestionar Conexiones PostgreSQL.

Caso de Uso: Gestionar Conexiones PostgreSQL

Descripción general: El caso de uso se inicia cuando el usuario escoge la opción de crear, actualizar o eliminar una conexión a la BD y el sistema muestra un formulario para que se introduzcan los datos para realizar la conexión en caso de crear o editar, en caso de eliminar se eliminaría una conexión antes hecha.

Tabla 5: Secciones a probar en el CU

Nombre	Escenarios de la	Descripción de la	Flujo Central

de la sección	sección	Funcionalidad	
SC1: Nueva conexión a la BD.	EC 1.1. Nueva conexión a la BD satisfactoriamente.	Esta funcionalidad crea una nueva conexión con la BD.	1. Seleccionar la opción nueva. 2. El sistema muestra el formulario para insertar los datos. 3. Se introducen los datos y seleccionar la opción Aceptar. 4. El sistema verifica que los datos fueron introducidos correctamente. 5. El sistema guarda los datos y crea la conexión con la BD.
	EC 1.2. Crear conexión a la BD erróneamente.	Esta funcionalidad crea una nueva conexión con la BD.	1. Seleccionar la opción nueva. 2. El sistema muestra el formulario para insertar los datos. 3. Se introducen los datos y seleccionar la opción Aceptar. 4. El sistema muestra un mensaje de error informando que existe algún campo vacio. Mensaje: No puede haber campos vacios.
SC2: Editar conexión a la BD.	EC 2.1. Editar conexión a la BD satisfactoriamente.	Esta funcionalidad edita una conexión existente con la BD.	1. Seleccionar la opción editar. 2. El sistema muestra el formulario para cambiar los datos. 3. Se cambian los datos y seleccionar la opción Aceptar. 4. El sistema verifica que los datos fueron introducidos correctamente. 5. El sistema guarda los datos y crea la conexión con la BD.

	EC 2.2. Editar conexión a la BD erróneamente.	Esta funcionalidad edita una conexión existente con la BD.	1. Seleccionar la opción editar. 2. El sistema muestra el formulario para cambiar los datos. 3. Se cambian los datos y seleccionar la opción Aceptar. 4. El sistema muestra un mensaje de error informando que existe algún campo vacio. Mensaje: No puede haber campos vacios.
SC3: Eliminar Conexión a la BD.	EC 3.1 Eliminar conexión.	Esta funcionalidad elimina una conexión existente con la BD.	1. Seleccionar el nombre de la BD que se quiere eliminar. 2. El sistema carga el nombre de la BD seleccionada. 3. Presionar eliminar. 4. El sistema elimina la conexión a la BD.

Luego de definidos los escenarios se procede a describir las variables que intervienen en los mismos.

Tabla 6: Descripción de las variables

No.	Nombre del campo	Clasificación	Valor Válido (V)	Descripción
1	Nombre	Campo de texto.	Cualquier combinación de letras y números.	Nombre que tendrá la conexión.
2	Servidor	Campo de texto.	Direcciones ip válidas o localhost.	Servidor de BD.

3	BD	Campo de texto.	Cualquier combinación de letras y números.	Nombre de la BD a conectarse.
4	Puerto	Botón de incremento y decremento.	Números enteros comprendidos en el rango entre 1024 y 65535.	Numero del puerto de la BD.
5	Usuario	Campo de texto.	Cualquier combinación de letras y números.	Nombre de usuario de la conexión.
6	Contraseña	Campo de texto.	Cualquier combinación de caracteres.	Contraseña de la conexión.

A continuación se presentan las matrices de datos asociadas a los distintos escenarios. Los valores de las variables se expresan en valido (V) o invalido (I).

Tabla 7 Matriz de datos SC 1: Nueva conexión a la BD.

Escenarios	Nombre	Servidor	BD	Puerto	Usuario	Contraseña	Respuesta del sistema
EC 1.1	**V**	**V**	**V**	**V**	**V**	**V**	El sistema guarda la conexión efectuada y la añade junto a las demás conexiones.
EC 1.2	Algunos de los datos toman valor I.						El sistema emite un mensaje de que no puede haber campos vacios.

Tabla 8 Matriz de datos SC 2: Editar conexión a la BD.

Escenarios	Nombre	Servidor	BD	Puerto	Usuario	Contraseña	Respuesta del sistema
EC 2.1	**V**	**V**	**V**	**V**	**V**	**V**	El sistema guarda la conexión antes editada y la añade junto a las demás conexiones.
EC 2.2	Algunos de los datos toman valor I.						El sistema emite un mensaje de que no puede haber campos vacios.

Matriz de datos SC 3: Eliminar conexión a la BD.

Descripción de las variables: No existen variables asociadas en este caso de uso.

Escenarios	Respuesta del sistema
EC3.1 Eliminar conexión.	El sistema emite un mensaje que ha sido eliminada la BD.

Ejecución de la prueba para SC 1: Nueva conexión a la BD.

Escenarios	Nombre	Servidor	BD	Puerto	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC 1.1	Convertir	localhost	db	5432	pruf	pass	El sistema emite un	Satisfactori

							mensaje que ha sido efectuada la conexión a la BD.	a
EC 1.2	Algunos de los datos toman valor I.						El sistema emite un mensaje para que llene los campos vacios.	

Ejecución de la prueba para SC 2: Editar conexión a la BD.

Escenarios	**Nombre**	**Servidor**	**BD**	**Puerto**	**Usuario**	**Contraseña**	**Respuesta del sistema**	**Resultado de la prueba**
EC 2.1	**Shape**	**10.34.2.3**	**Shape**	**1563**	**lbaes**	**Shp2sql**	El sistema emite un mensaje que ha sido efectuada la conexión a la BD.	Satisfactoria
EC	Algunos de los datos toman valor I.						El sistema	

2.2		emite un mensaje para que llene los campos vacios.	

Ejecución de la prueba para SC 3: Eliminar conexión con la BD.

Escenarios	Respuesta del sistema	Resultado de la prueba
EC 3.1	El sistema emite un mensaje que ha sido eliminada la BD.	Satisfactoria

4.9 Conclusiones

En el transcurso de este capítulo se obtuvieron los artefactos más significativos del diseño de la aplicación, como son el diagrama de clases del diseño y el diagrama de interacción. Estos modelos sirven de entrada para el flujo de implementación, facilitando un mayor entendimiento de la aplicación a los desarrolladores sobre las funcionalidades a implementar, así como también la utilidad de los diferentes patrones de diseño y arquitectura en el desarrollo del software.

Los elementos obtenidos mediante la realización del modelo de despliegue y del diagrama de componentes, permiten un mejor entendimiento de la distribución física y lógica del sistema implementado, facilitando la fase de implantación del producto.

Las pruebas realizadas a la aplicación arrojaron resultados satisfactorios, en consecuencia a lo anterior se puede arribar a que las funcionalidades requeridas fueron totalmente cumplimentadas.

CONCLUSIONES GENERALES

Con la culminación de este trabajo se concluye que:

- ✓ La metodología de desarrollo de software, el lenguaje de modelado y la herramienta CASE utilizados, permitieron la realización de todo el proceso con éxito y obtener los resultados esperados de esta investigación por lo que se propone su uso en caso de futuros desarrollos sobre el producto obtenido.
- ✓ La captura de los requisitos funcionales y no funcionales del sistema sentó las bases para desarrollar el trabajo de manera más organizada y poder dar cumplimiento a los objetivos perseguidos.
- ✓ Las pruebas realizadas a la aplicación arrojaron resultados satisfactorios, en consecuencia a lo anterior se arriba a que las funcionalidades definidas fueron totalmente cumplimentadas.
- ✓ La implementación de la selección y carga de múltiples archivos en entornos web es posible, suprimiendo así las limitaciones respecto a las aplicaciones de tipo Desktop.
- ✓ Las herramientas utilizadas, al ser software libre, contribuyen al cumplimiento de las políticas del país respecto a la soberanía tecnológica. Además, de esta forma se facilita la colaboración a fin de mejorar el producto actual.

Finalmente, luego de diseñar e implementar una aplicación, se obtuvo un producto de software capaz realizar la transformación de múltiples archivos ESRI Shapefile a PostgreSQL proporcionando mayor comodidad y rapidez al proceso de preparación de datos requerido en Sistemas de Información Geográfica desarrollados en el Departamento Geoinformática. Por tanto, se concluye que se cumplieron satisfactoriamente los objetivos trazados al inicio de la investigación.

RECOMENDACIONES

Al concluir esta investigación y después de haber alcanzado los objetivos planteados, se recomienda darle continuidad al producto logrado, agregando nuevas funcionalidades. A continuación se sugieren algunas:

- ✓ Visualización de los datos de un archivo ESRI Shapefile en un mapa.
- ✓ Control de usuarios.

REFERENCIAS BIBLIOGRÁFICAS

Microsoft Corporation. 2009. *Application Architecture Guide.* 2009.

A Patern Languaje. **Alexander Christopher, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Angel Shlomo. 1977.** New York : Oxford University Press, 1977.

Alvarez, Miguel Angel. 2009. Desarrolloweb. [En línea] 2009. [Citado el: 3 de marzo de 2011.] 9.

Apache - Web Server. 2010. Apache - Web Server. [En línea] 5 de noviembre de 2010. http://www.apache.org/.

2006. Centro de Gestion Informática. [En línea] mayo de 2006. [Citado el: 6 de febrero de 2011.] http://cgi.una.ac.cr/AUP/html/overview.html..

Cerda, Felipe. Techbloog. Tendencias Tecnológicas en Español. [En línea] [Citado el: 21 de marzo de 2011.] www.techbloog.com/talks/netbeans65es_cl.pdf..

EVA. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 15 de enero de 2011.] http://eva.uci.cu/mod/resource/view.php?id=33748/Metodologías_de_Desarrollo_de_Software.

2010. ExtJS development Group. [En línea] 3 de marzo de 2010. [Citado el: 9 de marzo de 2011.] http://www.extjs.com/.

Gavin, Powell. 2010. Beginning Database Desing. *Beginning Database Desing.* [En línea] 5 de noviembre de 2010.

Geotecnologias. 2007. Geotecnologias. [En línea] 2007. [Citado el: 12 de noviembre de 2010.] http://www.geotecnologias.com/Documentos/GIS.pdf.

Gomez, Laura Bermejo y Enrique. 2007. *Eclipse como IDE.Características principales, funcionalidad, utilización y caso práctico.* 2007.

Google Code. 2010. Google Code. [En línea] 2010. [Citado el: 13 de noviembre de 2010.] http://code.google.com/intl/es-AR/apis/kml/documentation/kml_tut.html.

IEEE. 1998. *IEEE:Guide for Developing System Requirements Specification.* 1998.

James Rumbaugh, Grady Booch y Ivar Jacobson. 2000. *El lenguaje unificado de modelado.Manual de Referencia.* Madrid : Pearson Education, 2000.

José H. Canós, Patricio Letelier y Carmen Penadés. *Métodologías Ágiles en el Desarrollo de Software.* Valencia : s.n.

Linux-Ciberula. 2011. Ciberaula. [En línea] 23 de enero de 2011. [Citado el: 12 de marzo de 2011.] http://linux.ciberaula.com/articulo/linux_apache_intro..

Microsoft TechNet. 2010. Microsoft TechNet. [En línea] 2010. //technet.microsoft.com/es-es/library/cc730981%28WS.10%29.aspx..

Molpecere, Alberto. 2002. *Procesos de desarrollo: RUP, XP, FDD.* 2002.

MUÑOZ-CRUZADO GARCÍA, Carmen. 2006. [En línea] 2006. [Citado el: 7 de noviembre de 2010.]

Opengeospatial. 2010. Opengeospatial. [En línea] 2010. [Citado el: 24 de noviembre de 2010.] http://www.opengeospatial.org/standards/gml.

Pérez, Rubén González y Sergio. 2007. *Introducción al Rational Rose. Funcionalidad general.* Barcelona : s.n., 2007.

Perú Global Consultin. 2008. Perú Global Consultin. [En línea] 2008. [Citado el: 5 de noviembre de 2010.] peruglbalconsultin.com.

PostgreSQL. 2009. PostgreSQL. [En línea] 2009. [Citado el: 7 de noviembre de 2010.] postgresql.org/.

Rational. *Rational.* [En línea] [Citado el: 26 de enero de 2011.] http://www.rational.com.ar/herramientas/rup.html.

Roberth G. Figueroa, Camilo J. Solís, Armando A. Cabrera. 2008. Adonisnet's Weblog. [En línea] 2008. [Citado el: 15 de enero de 2011.] http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato.doc..

Salinas, J. 2007. Información Geográfica, Software Libre e Infraestructuras de Datos Espaciales. [En línea] 2007. [Citado el: 1 de diciembre de 2010.]

Sierra, María. 2010. Trabajando con Visual Paradigm for UML. [En línea] 2010. [Citado el: 24 de febrero de 2011.] http://personales.unican.es/ruizfr/is1/doc/lab/01/is1-p01-trans.pdf.

Soluciones y Propuestas Rational. Soluciones y Propuestas Rational. [En línea] http://www.rational.com.ar/herramientas/rup.html..

Visual Paradigm Design Group. 2009. Visual Paradigm. [En línea] 12 de Noviembre de 2009. [Citado el: 26 de marzo de 2011.] http://www.visual-paradigm.com/.

Webnova. 2010. Webnova. [En línea] 2010. [Citado el: 9 de marzo de 2011.] http://www.webnova.com.ar/articulo.php?recurso=95.

2009. Zonum Solutions. [En línea] 2009. [Citado el: 3 de diciembre de 2010.] http://zonums.com/shp2kml.html.

BIBLIOGRAFÍA

Microsoft Corporation. 2009. *Application Architecture Guide.* 2009.

A Patern Languaje. **Alexander Christopher, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Angel Shlomo. 1977.** New York : Oxford University Press, 1977.

Academia, Real. Diccionario de la Real Academia de la Lengua Española. *Diccionario de la Real Academia de la Lengua Española.* [En línea] [Citado el: 18 de enero de 2011.] http:// www.rae.es.

Alfaro, F.M. 2005. *Tecnologia Cliente - Servidor.* 2005.

Alvarez, Miguel Angel. 2009. Desarrolloweb. [En línea] 2009. [Citado el: 3 de marzo de 2011.] 9.

Amaro Calderón, Sarah Dámaris. 2007. *Metodologías Ágiles.* Perú : s.n., 2007.

Apache - Web Server. 2010. Apache - Web Server. [En línea] 5 de noviembre de 2010. http://www.apache.org/.

2006. Centro de Gestion Informática. [En línea] mayo de 2006. [Citado el: 6 de febrero de 2011.] http://cgi.una.ac.cr/AUP/html/overview.html..

Cerda, Felipe. Techbloog. Tendencias Tecnológicas en Español. [En línea] [Citado el: 21 de marzo de 2011.] www.techbloog.com/talks/netbeans65es_cl.pdf..

EVA. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 15 de enero de 2011.] http://eva.uci.cu/mod/resource/view.php?id=33748/Metodologías_de_Desarrollo_de_Software.

2010. ExtJS development Group. [En línea] 3 de marzo de 2010. [Citado el: 9 de marzo de 2011.] http://www.extjs.com/.

ExtJS, development Group. 2010. ExtJS Javascript Framework and RIA Plataform. *ExtJS Javascript Framework and RIA Plataform.* [En línea] 03 de marzo de 2010. [Citado el: 19 de febrero de 2011.] http://www.extjs.com/.

Frederick, Shea, Ramsay, Colin y Blades, Steve 'Cutter'. 2008. *Learning ExtJS.* s.l. : PACKT, 2008. ISBN 978-1-847195 .

Gavin, Powell. 2010. Beginning Database Desing. *Beginning Database Desing.* [En línea] 5 de noviembre de 2010.

Geotecnologias. 2007. Geotecnologias. [En línea] 2007. [Citado el: 12 de noviembre de 2010.] http://www.geotecnologias.com/Documentos/GIS.pdf.

Gomez, Laura Bermejo y Enrique. 2007. *Eclipse como IDE.Características principales, funcionalidad, utilización y caso práctico.* 2007.

Google Code. 2010. Google Code. [En línea] 2010. [Citado el: 13 de noviembre de 2010.] http://code.google.com/intl/es-AR/apis/kml/documentation/kml_tut.html.

IEEE. 1998. *IEEE:Guide for Developing System Requirements Specification.* 1998.

James Rumbaugh, Grady Booch y Ivar Jacobson. 2000. *El lenguaje unificado de modelado.Manual de Referencia.* Madrid : Pearson Education, 2000.

José H. Canós, Patricio Letelier y Carmen Penadés. *Métodologías Ágiles en el Desarrollo de Software.* Valencia : s.n.

Lenguajes-de-Programacion. [En línea] [Citado el: 13 de febrero de 2011.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

Linux-Ciberula. 2011. Ciberaula. [En línea] 23 de enero de 2011. [Citado el: 12 de marzo de 2011.] http://linux.ciberaula.com/articulo/linux_apache_intro..

Marley, Jimi. 2011. Programación en castellano. *¿Por qué elegir PHP?* [En línea] 2011. [Citado el: 21 de febrero de 2011.] http://www.programacion.com/articulo/por_que_elegir_php_143.

Martín, Manuel Martín. 2008. *Manual PostGIS.* 2008.

Martínez, Jose Carlos. 2010. *PostGIS. Bases de datos espaciales. .* España : s.n., 2010.

Microsoft TechNet. 2010. Microsoft TechNet. [En línea] 2010. //technet.microsoft.com/es-es/library/cc730981%28WS.10%29.aspx..

Molpecere, Alberto. 2002. *Procesos de desarrollo: RUP, XP, FDD.* 2002.

MUÑOZ-CRUZADO GARCÍA, Carmen. 2006. [En línea] 2006. [Citado el: 7 de noviembre de 2010.]

Opengeospatial. 2010. Opengeospatial. [En línea] 2010. [Citado el: 24 de noviembre de 2010.] http://www.opengeospatial.org/standards/gml.

Pérez, Rubén González y Sergio. 2007. *Introducción al Rational Rose. Funcionalidad general.* Barcelona : s.n., 2007.

Perú Global Consultin. 2008. Perú Global Consultin. [En línea] 2008. [Citado el: 5 de noviembre de 2010.] peruglbalconsultin.com.

PHP Group. 2010. PHP Hipertext Preprocesor. [En línea] 17 de Marzo de 2010. [Citado el: 17 de Marzo de 2010.] http://www.php.net/.

php.net. 2010. *PHP Development.* s.l. : Philip Olson, 2010.

Pino, José Luis López. 2010. Servidores web más usados. [En línea] 30 de julio de 2010. [Citado el: 31 de marzo de 2011.] http://lopezpino.es/2010/07/30/servidores-web-mas-usados/.

PostgreSQL Fundation. 2009. PostgreSQL - Batabase Server. [En línea] 15 de 12 de 2009. http://www.postgresql.org/.

PostgreSQL. 2009. PostgreSQL. [En línea] 2009. [Citado el: 7 de noviembre de 2010.] postgresql.org/.

Pressman, R.S. 2000. *Ingeniería del Software. Un enfoque práctico.* 2000.

Rational. *Rational.* [En línea] [Citado el: 26 de enero de 2011.] http://www.rational.com.ar/herramientas/rup.html.

Roberth G. Figueroa, Camilo J. Solís, Armando A. Cabrera. 2008. Adonisnet's Weblog. [En línea] 2008. [Citado el: 15 de enero de 2011.] http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato.doc..

Rojas, Alan D. Osorio. 2008. *Manual introductorio a VP.* 2008.

Salinas, J. 2007. Información Geográfica, Software Libre e Infraestructuras de Datos Espaciales. [En línea] 2007. [Citado el: 1 de diciembre de 2010.]

Sierra, María. 2010. Trabajando con Visual Paradigm for UML. [En línea] 2010. [Citado el: 24 de febrero de 2011.] http://personales.unican.es/ruizfr/is1/doc/lab/01/is1-p01-trans.pdf.

Soluciones y Propuestas Rational. Soluciones y Propuestas Rational. [En línea] http://www.rational.com.ar/herramientas/rup.html..

Visual Paradigm Design Group. 2009. Visual Paradigm. [En línea] 12 de Noviembre de 2009. [Citado el: 26 de marzo de 2011.] http://www.visual-paradigm.com/.

Webnova. 2010. Webnova. [En línea] 2010. [Citado el: 9 de marzo de 2011.] http://www.webnova.com.ar/articulo.php?recurso=95.

2009. Zonum Solutions. [En línea] 2009. [Citado el: 3 de diciembre de 2010.] http://zonums.com/shp2kml.html.