UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

TÍTULO: Catálogo de archivos contenedores de datos provenientes de dispositivos GPS.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA

AUTOR: *Jorge Javier Olivera Lemes*

TUTOR: Ing. Jesse Daniel Cano Otero

DEDICATORIA

A mis padres por siempre preocuparse y confiar en mí.

A mi hermana, que ha sido mi mejor amiga y confidente.

A mi sobrinita Orialis, mi niñita querida.

A mis tíos y tías por siempre estar a mi lado dándome consejos, en espacial a mis tías Irma, Inés María e Iris que han sido como mis segundas madres.

A mis primos y primas por confiar en mí y sobre todo a Keilys (la bella) y Galina que siempre han estado conmigo apoyándome en todo momento.

A todos mis familiares que me sirvieron de apoyo.

A todas las personas que de una forma u otra se preocuparon por mis estudios.

AGRADECIMIENTOS

A mis padres quienes son mi razón de ser y me apoyaron incondicionalmente y son una gran inspiración para mí.

A mi hermanita querida, mujer que se enfrenta a los problemas que la vida le pone en su camino de manera extraordinaria.

A mis amigos que siempre han estado presentes cuando los he necesitado.

A la Revolución Cubana por garantizarme todos los medios necesarios para mi educación.

A la Universidad de Ciencias Informáticas por formarme como un buen profesional.

A mi tutor Jesse Daniel Cano Otero por su paciencia y dedicación.

A todas las personas que en un momento u otro me brindaron su ayuda.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Jorge Javier Olivera Lemes Ing. Jesse Daniel Cano Otero

Datos de Contacto

TUTOR: Ing. Jesse Daniel Cano Otero.

Correo Electrónico: jdcano@uci.cu

Universidad de las Ciencias Informáticas.

RESUMEN

El presente trabajo muestra el resultado de una investigación realizada con el objetivo de desarrollar un sistema informático que gestione y catalogue los archivos contenedores de datos provenientes de dispositivos GPS. Para un mejor entendimiento de la materia se realizó un estudio profundo de los tipos de archivos y su forma de representación, apoyándose para el estudio en los diferentes autores, y empleándose diferentes métodos: tantos teóricos como empíricos, que posibilitaron una mayor comprensión del problema actual.

Se identificaron las principales tendencias y tecnologías actuales para la modelación y desarrollo del sistema. La aplicación diseñada como producto final además de minimizar esfuerzos del personal del proyecto Control de Flotas ayuda a tener un control de los diferentes archivos provenientes del GPS en funcionamiento en el proyecto. El documento recoge los distintos requisitos que hacen posible el éxito y limitaciones de la misma, los resultados tangibles y no tangibles mediante un estudio de factibilidad realizado, además se demuestra que el sistema desarrollado cumple con la calidad requerida gracias a un conjunto de pruebas efectuadas.

Palabras Claves

Gestione, catalogue, archivos, GPS, tecnologías, tendencias, Control de Flotas, minimizar, esfuerzos, factibilidad.

Índice

RESUMEN 6

INTRODUCCIÓN 10

Capítulo 1. Fundamentación Teórica 14

1.1. Introducción. 14

1.2. Principales conceptos asociados al dominio del problema. 14

1.3. Datos contenidos en los archivos GPS. 15

1.4. Características del proceso de representación de la información contenida en los archivos provenientes de dispositivos GPS. 16

1.5. Estado del Arte. 23

1.6. Conclusiones. 25

Capítulo 2. Propuestas y Selección de Tecnologías. 26

2.1. Introducción. 26

2.2. Lenguajes de Programación. 26

2.2.1. PHP. 26

2.2.2. ASP.NET. 27

2.2.3. Java. 28

2.2.4. C# 29

2.3. Selección del lenguaje de programación a utilizar. 30

2.4. Metodologías de Desarrollo. 31

2.4.1. Programación Extrema (XP). 31

2.4.2. Microsoft Solution Framework. 32

2.4.3. RUP. 32

2.5. Selección de la Metodología a utilizar. 34

2.6. Herramientas. 35

2.6.1. Herramientas CASE. 35

2.6.1.1. Rational Rose. 35

2.6.1.2. Visual Paradigm. 36

2.7. Selección de la herramienta CASE a utilizar. 36

2.8. Entorno de desarrollo. NetBeans_IDE 6.8. 37

2.9. La Plataforma Java. 37
2.10. Sistemas Gestores de Bases de Datos (SGBD). 38
2.10.1. MySQL. 39
2.10.2. PostgreSQL. 40
2.11. Selección del Sistema Gestor de Base de Datos a utilizar. 41
2.12. Conclusiones. 41
Capítulo 3. Presentación de la solución propuesta. 42
3.1. Introducción. 42
3.2. Modelo de dominio. 42
3.2.1. Glosario de Términos del Negocio. 43
3.3. Requisitos Funcionales. 44
3.4. Requisitos no Funcionales. 45
3.5. Descripción del sistema propuesto. 46
3.5.1. Descripción de los actores del sistema. 46
3.5.2. Casos de Uso del Sistema. 46
3.5.2.1.1. Caso de Uso Autenticar Usuario. 47
3.5.2.1.2. Caso de Uso Gestionar datos del archivo. 47
3.5.2.1.3. Caso de Uso Buscar Archivos. 51
3.5.2.1.4. Caso de Uso Clasificar Archivos. 53
3.5.2.1.5. Caso de Uso Visualizar Información. 54
3.6. Conclusiones. 55
Capítulo 4. Construcción de la solución propuesta. 56
4.1. Introducción. 56
4.2. Patrones GRASP. 56
4.3. Patrón Arquitectónico. 57
4.3.1. Descripción del patrón. 57
4.4. Análisis. 58
4.4.1. Modelo de Análisis. 58
4.4.1.1. Diagramas de Clases de Análisis. 59
4.5. Diseño. 59

4.5.1. Diagrama de Interacción. 59
4.5.1.1. Diagrama de Colaboración. 59
4.5.2. Diagrama de Clases del diseño. 60
4.5.3. Diseño de la Base de Datos. 60
4.5.3.1. Diagrama de Clases Persistentes. 60
4.5.3.2. Modelo de Datos. 61
4.5.3.3. Descripción de las tablas de la Base de Datos. 61
4.5.4. Principios de diseño. 62
4.6. Implementación. 63
4.6.1. Modelo de Implementación. 63
4.6.1.1. Diagrama de Componentes. 63
4.6.1.2. Diagrama de Despliegue. 64
4.7. Pruebas del sistema propuesto 65
4.7.1. Diseño de Casos de Pruebas. 66
4.8. Conclusiones. 66
Capítulo 5. Estudio de Factibilidad 67
5.1. Método de Estimación por Puntos de Casos de Uso. 67
5.2. Beneficios Tangibles e intangibles. 73
5.2.1. Beneficios Tangibles. 73
5.2.2 Beneficios intangibles. 74
5.2.3. Análisis de Costos y Beneficios. 74
5.3. Conclusiones. 74
Conclusiones Generales 75
Recomendaciones 76
Bibliografía Referenciada 77
Bibliografía Consultada 80
Glosario de Términos 83

INTRODUCCIÓN

En los últimos años, las tecnologías han conseguido un gran protagonismo en la comunidad científico-tecnológica y, al mismo tiempo, la informática dentro de las ciencias que apoyan ese crecimiento ha aportado cambios significativos. Su impacto a nivel global se ve reflejado en todas las esferas de la sociedad y juega un papel fundamental en la vida del hombre. El software es un instrumento capaz de influir en el desarrollo y trasformación de la sociedad y ha facilitado y agilizado varios procesos que se manejaban con anterioridad manualmente, además poseen la característica de controlar o hacer accesible, en la mayoría de los casos, los adelantos electrónicos.

En el mundo, el uso de los Sistemas de Información Geográficas (SIG), ha posibilitado un gran avance en la economía y en el desarrollo de las naciones. Estos sistemas son la combinación de componentes: personas especializadas, datos descriptivos y espaciales, métodos analíticos, hardware y software, organizados para analizar, gestionar y visualizar todo tipo de información referenciada geográficamente (1). Los SIG pueden ser divididos en dos grandes modelos: rasters y vectorial, ambos tienen sus propios potencialidades y carencias, por lo que, los modernos SIG incorporan elementos de ambas técnicas de representación, además de extensiones que permiten la conversión de una a otro modelo.

El Sistema de Posicionamiento Global (GPS) es una de estas nuevas tecnologías que están tomando auge en la actualidad y forma parte del conjunto de los Sistemas de Información Geográficas. El GPS es un sistema de localización, diseñado por el Departamento de Defensa de los Estados Unidos con fines militares para proporcionar estimaciones precisas de posición, velocidad y tiempo; operativo desde 1995, utiliza conjuntamente una red de ordenadores y una constelación de 24 satélites para determinar por triangulación, la altitud, longitud y latitud de cualquier objeto en la superficie terrestre (2). En el ámbito civil y alegando razones de seguridad sólo se permite el uso de un subconjunto degradado de señales GPS. Sin embargo la comunidad civil ha encontrado alternativas para obtener una excelente precisión en la localización. Gracias a ellas las aplicaciones civiles han experimentado un gran crecimiento y actualmente existen más de 70 fabricantes de receptores GPS.

En Cuba se ha hecho un análisis sobre la importancia de que las entidades del Ministerio de Transporte incorporen a sus vehículos estos dispositivos como un instrumento de control, dándole prioridad a los vehículos de carga, sobre todo los

que trasladan mercancías, esto es fundamental porque se evitan indisciplinas y gasto innecesario de combustible al realizar gestiones ajenas a su cometido. Este dispositivo satelital no solo es usado en el control de flotas de transporte sino que permite ubicar también el lugar donde se encuentra cualquier objeto, como teléfono y hasta las personas y su movimiento. (3)

De los GPS provienen varios tipos de archivos, los cuales contienen datos espaciales. Por la diferencia que existe entre ellos se hace necesario buscar alternativas para facilitar el proceso de organización y clasificación.

La Universidad de las Ciencias Informáticas ha desarrollado varios mecanismos con el objetivo de ayudar e impulsar el desarrollo del país y ha creado un nuevo proyecto para el Control de Flotas; este proyecto surge con el objetivo de brindar algunos servicios y realizar aplicaciones GPS. Los archivos provenientes de dispositivos GPS son diversos y almacenan diferentes tipos de datos; la organización y clasificación de ellos realizada de forma manual sería un trabajo engorroso, esto hace que el proceso sea lento, con baja calidad y se emplearía mucho tiempo realizando estas acciones.

Teniendo en cuenta la situación anterior, el **problema** radica en: ¿Cómo describir y relacionar de manera organizada los archivos contenedores de datos provenientes de dispositivos GPS? El **objeto de estudio** se centra en la automatización del proceso de representación de la información contenida en los archivos provenientes de dispositivos GPS.
Se determina como **campo de acción** de la investigación, la clasificación de los archivos contenedores de datos provenientes de dispositivos GPS.

Para dar solución al problema antes mencionado se define como **objetivo general,** Elaborar un Catálogo de archivos contenedores de datos provenientes de dispositivos GPS.

Para guiar la investigación se confeccionó la siguiente **Idea a Defender:**

Con la elaboración de un Catálogo de archivos contenedores de datos GPS se podrá describir y relacionar de manera organizada estos archivos.

Para llevar a cabo el objetivo del trabajo de diploma se proponen las **Tareas de la Investigación** siguientes:

- Establecer los fundamentos teóricos correspondientes al tema en cuestión.
- Caracterizar los datos provenientes de dispositivos GPS.
- Proponer lenguajes existentes que pueden ser usados en el trabajo para realizar la mejor selección del que se utilizará para la confección del Catálogo.
- Proponer herramientas y metodologías adecuadas que permitan desarrollar una solución al problema.
- Realizar un análisis sobre la estructura de los Catálogos para crear una herramienta que logre las expectativas trazadas.
- Crear el Catálogo para la clasificación de los archivos contenedores de datos provenientes de dispositivos GPS.

Para realizar las tareas de investigación se utilizan varios **métodos científicos** cada uno con un objetivo bien definido:

Métodos Teóricos:

Analítico – Sintético: Para realizar un análisis más detallado sobre las diferentes formas de representación de la información proveniente de dispositivos GPS.

Análisis Histórico Lógico: Para posibilitar el análisis histórico y lógico del proceso de representación de la información proveniente de dispositivos GPS.

Modelación: Se utiliza para la modelación de diagramas, representar el proceso de desarrollo y propiciar un mejor entendimiento de la solución a implementar.

Métodos Empíricos:

Observación: Se emplea en todo momento. Con este método se analiza cada fase del proceso.

Entrevistas: Para recoger criterios acerca del problema en cuestión. Se escogió como muestra para realizar las entrevistas a los profesionales del proyecto Control de Flotas y algunos de sus estudiantes.

El presente trabajo se encuentra estructurado por 5 capítulos que abordan los siguientes contenidos:

En el Capítulo 1 se describe la Fundamentación Teórica. Se plantean los principales conceptos y términos abordados en la investigación, se analiza el Estado del Arte del tema tratado y se definen las características del objeto de estudio.

En el Capítulo 2 se definen las tendencias, técnicas, tecnologías, metodologías y software usados en la actualidad o en las que se apoya para darle solución al problema planteado.

En el Capítulo 3 se hace un análisis del dominio de la aplicación, se explican las principales actividades o rasgos de los eventos que logren identificarse, se describen las características del sistema y se generan documentos referentes a esta fase. También se definen requisitos funcionales y no funcionales.

En el Capítulo 4 se realiza el Análisis y Diseño del sistema, se define el Modelo de Análisis, Modelo de Clases de Análisis y sus representaciones gráficas. En el diseño se realizan los Diagramas de Interacción y el Diagrama de Clases, se realiza la implementación, que contiene el Diagrama de Componentes así como su representación gráfica. Se presenta el plan de pruebas y los casos de prueba diseñados para probar la aplicación.

En el Capítulo 5 se realiza un estudio para verificar si la solución final es factible, realizando la estimación por puntos de casos de uso, donde se obtiene como resultado si la aplicación es factible de construir.

Capítulo 1. Fundamentación Teórica.

1.1. Introducción.

En este capítulo se expondrán los conceptos y elementos que constituyen la base teórica para la realización del presente trabajo de diploma, brindando de esta forma una mejor comprensión en la consistencia de la investigación. Para el mismo se hace necesario dominar los aspectos más significativos relacionados con el objeto de estudio de la misma: la automatización del proceso de representación de la información contenida en los archivos provenientes de dispositivos GPS.

Se describen además los principales conceptos asociados al dominio del problema.

1.2. Principales conceptos asociados al dominio del problema.

Existen conceptos asociados al dominio del problema que deben ser descritos para su mejor entendimiento haciéndose énfasis en sus principales características. A continuación se mencionan y caracterizan los más importantes.

1.2.1. Sistemas de Información Geográfica (SIG).

Es un sistema de hardware, software y procedimientos, diseñados para soportar la captura, el manejo, la manipulación, el análisis, el modelado y el despliegue de datos espacialmente referenciados, para la solución de los problemas complejos del manejo y planeamiento territorial. (4)

1.2.2. Modelos de Datos Vectoriales.

El modelo vectorial es una estructura de datos utilizada para almacenar datos geográficos. Los datos vectoriales constan de líneas o arcos, definidos por sus puntos de inicio y fin, y puntos donde se cruzan varios arcos, los nodos. La localización de los nodos y la estructura topológica se almacena de forma explícita. Las entidades quedan definidas por sus límites solamente y los segmentos curvos se representan como una serie de arcos conectados. El almacenamiento de los vectores implica el almacenamiento explícito de la topología, sin embargo solo almacena aquellos puntos que definen las entidades y todo el espacio fuera de estas no está considerado. (5)

1.2.3. Modelos de Datos Rasters.

El modelo rasters es un método para el almacenamiento, el procesado y la visualización de datos geográficos. Cada superficie a representar se divide en filas y columnas, formando una malla o rejilla regular. Cada celda ha de ser rectangular, aunque no necesariamente cuadrada. Cada celda de la rejilla guarda tanto las

coordenadas de la localización como el valor temático. La localización de cada celda es implícita, dependiendo directamente del orden que ocupa en la rejilla, a diferencia de la estructura vectorial en la que se almacena de forma explícita la topología. Las áreas que contienen idéntico atributo temático son reconocidas como tal, aunque las estructuras rasters no identifican los límites de esas áreas como polígonos en sí. (6)

1.2.4. Sistema de Posicionamiento Global (GPS).

El GPS es un sistema global de navegación por satélite (GNSS) que permite determinar en todo el mundo la posición de un objeto, una persona, un vehículo o una nave, con una precisión hasta de centímetros, aunque lo habitual son unos pocos metros de precisión. (7)

1.2.5. Catálogo.

El Catálogo es una lista o inventario ordenada y clasificada de cualquier tipo de objetos, además es una enumeración organizada de las piezas que contiene una colección determinada. Supone la descripción de las piezas, elementos para la ordenación de las descripciones y recursos para la localización de las piezas en la colección. (8)

1.3. Datos contenidos en los archivos GPS.

Los datos provenientes de dispositivos GPS representan los objetos del mundo real (carreteras, el uso del suelo o altitudes). Estos objetos se pueden dividir en dos abstracciones: objetos discretos (una casa) y continuos (cantidad de lluvia caída o una elevación). Los GPS se centran en el manejo de datos en formato rasters y vectorial. Un tipo de dato rasters es, en esencia, cualquier tipo de imagen digital representada en mallas y centrado en las propiedades del espacio. Los datos rasters se componen de filas y columnas de celdas, cada celda almacena un valor único. Estos datos pueden ser imágenes (imágenes rasters), con un valor de color en cada celda (o pixel). Otros valores registrados para cada celda pueden ser un valor discreto, como el uso del suelo, valores continuos y temperaturas. Los datos rasters se almacenan en diferentes formatos, desde un archivo estándar basado en la estructura de TIFF, JPEG, etc. a grandes objetos binarios (BLOB); los cuales son datos de gran tamaño que cambian de forma dinámica y generalmente son imágenes y archivos de sonido.

En los datos vectoriales, el interés de las representaciones se centra en la precisión de localización de los elementos geográficos sobre el espacio y donde los fenómenos a representar son discretos, es decir, de límites definidos. Cada una de

estas geometrías está vinculada a una fila en una base de datos que describe sus atributos. Para modelar digitalmente las entidades del mundo real se utilizan tres elementos geométricos: el punto, la línea y el polígono.

Los puntos se utilizan para las entidades geográficas que mejor pueden ser expresadas por un único punto de referencia. En otras palabras: la simple ubicación. Por ejemplo, las ubicaciones de los pozos, picos de elevaciones o puntos de interés. Los puntos transmiten la menor cantidad de información de estos tipos de archivo y no son posibles las mediciones. También se pueden utilizar para representar zonas a una escala pequeña. Por ejemplo, las ciudades en un mapa del mundo estarán representadas por puntos en lugar de polígonos.

Las líneas unidimensionales son usadas para rasgos lineales como ríos, caminos, ferrocarriles, rastros, líneas topográficas o curvas de nivel. De igual forma que en las entidades puntuales, en pequeñas escalas pueden ser utilizados para representar polígonos. En los elementos lineales puede medirse la distancia.

Los polígonos bidimensionales se utilizan para representar elementos geográficos que cubren un área particular de la superficie de la tierra. Estas entidades pueden representar lagos, límites de parques naturales, edificios, provincias, o los usos del suelo, por ejemplo. Los polígonos transmiten la mayor cantidad de información en archivos con datos vectoriales y en ellos se pueden medir el perímetro y el área.

1.4. Características del proceso de representación de la información contenida en los archivos provenientes de dispositivos GPS.

Desde los años 60, se empezaron a utilizar medios más eficaces para el tratamiento de la información geográfica. Esta información podía ser solamente mapas que mostraban detalles descriptivos de las regiones representadas. Desde entonces se han empleado métodos sofisticados tanto para la recolección de la información, como para su tratamiento y representación. Los SIG iniciaron para solucionar las necesidades de: conocer la situación del territorio, para poder planificar posibles áreas a poblar, organizar los suelos útiles para la agricultura, etc. Un SIG implica el almacenamiento, análisis, manipulación y representación de la información geográfica.

La visualización y representación de la información geográfica funciona de forma similar a la utilización de varias transparencias de mapas temáticos[1], cada

[1] En la sección de representación utilizando mapas se indica que es un mapa temático y los tipos de mapas.

transparencia con un tema de información diferente sobre el mismo territorio. (ver figura1)

Combinación de dos capas de información. 1) capa de carreteras.

2) capa de tipo de suelos. 3) combinación de ambas capas.

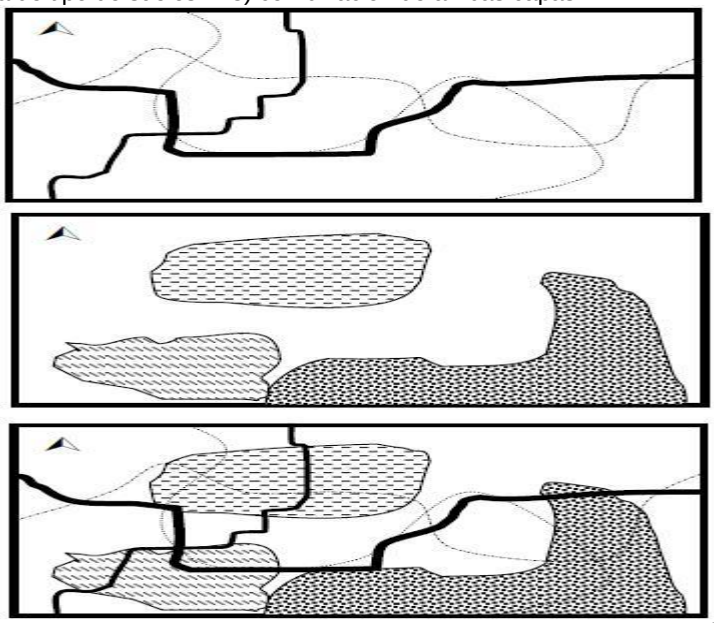


Figura1.En otras palabras, se puede pensar en tener un mapa de la información de carreteras y caminos de un lugar, otro mapa de tipo de suelo del mismo lugar, al ponerlos juntos se obtiene un mapa de carreteras, caminos y tipos de suelo. (9)

Representación de la información geográfica utilizando mapas.

Los mapas son una herramienta muy utilizada para la representación de la información geográfica. Los mapas pueden ser de dos tipos: mapas topográficos y mapas temáticos. Los mapas topográficos muestran los elementos naturales del área representada en el mapa, por ejemplo: montañas, ríos, arroyos, áreas boscosas, volcanes, sierras, áreas pantanosas, etc. (ver figura2)

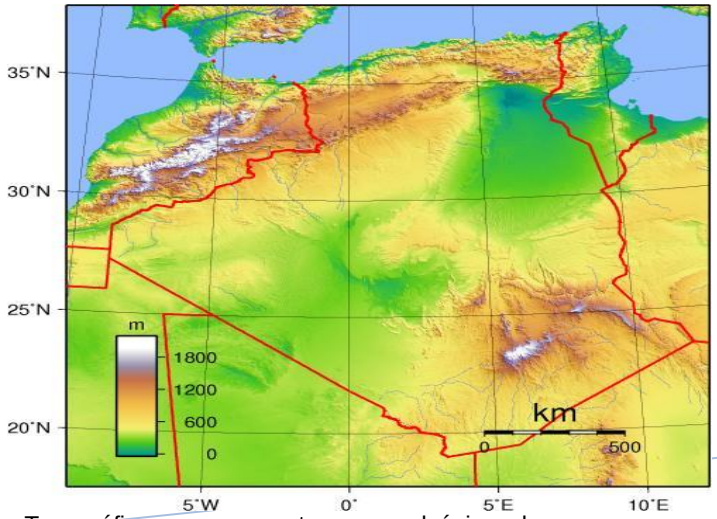


Figura2.Mapa Topográfico que representa zonas volcánicas, boscosas y masas de agua. (10)

Los mapas temáticos presentan un tipo de información específica, por ejemplo, las cartas náuticas en las que los datos son presentados, están relacionados con los océanos o grandes masas de agua que son utilizadas para navegar. Entre la información que puede presentar una carta náutica están: profundidades, bancos de arena, distancia en millas náuticas, puertos, etc. (ver figura3)

Figura3.Mapa Temático. (11)

Entre la información que proporciona un SIG se tienen los mapas, estas representaciones visuales de los datos geográficos contenidos en un SIG, se proporcionan en forma de capas o niveles, en cada capa o nivel se cuenta con un tema específico. Por ejemplo, la información de carreteras y la utilización del suelo representan 2 niveles o capas por separado. (ver figura4)

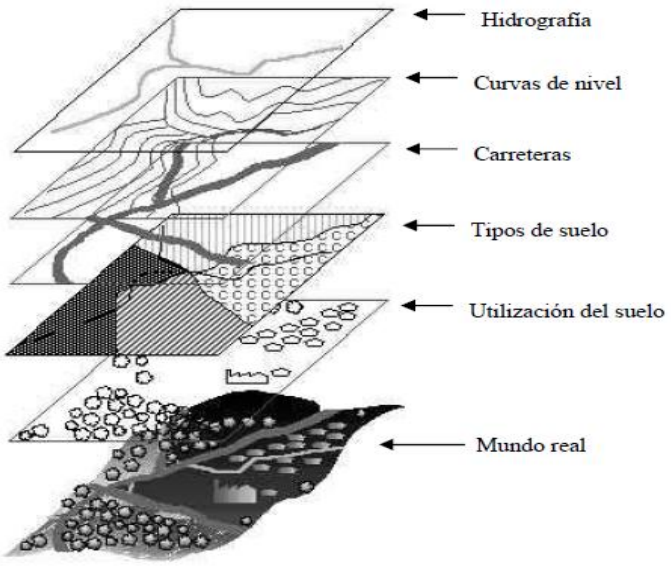


Figura4.Representación de la información geográfica en un SIG. (9)

Existen otras formas de representación de la información geográfica las cuales utilizan modelos y estructuras de datos. La información almacenada en los archivos rasters y vectorial provenientes de los GPS es representada utilizando técnicas de representación, cada uno de estos archivos tiene una forma específica de representar sus datos. Para la representación de los archivos rasters se utiliza el Modelo de Datos Rasters, este tiene como característica, llevar a cabo una representación discreta del mundo real, empleando una malla de rejillas regulares que se denominan celdillas o píxeles. Para cada celdilla se almacena un valor numérico que representa el valor de un determinado aspecto del mundo real en el interior de dicha celdilla.

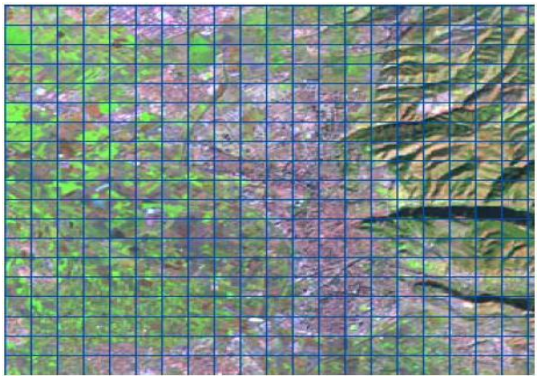


Figura5.Malla rasters. En cada una de las celdillas o pixeles se almacenan valores de un determinado aspecto o variable del mundo real. (12)

Cada capa rasters almacena en sus celdillas valores de un único aspecto del mundo real: usos del suelo, temperatura, pendiente, precio del suelo, carreteras, red hidrográfica y muchos otros más, de tal forma que cada variable de la realidad sería representado por una capa rasters.

El valor numérico almacenado para el caso de las variables cualitativas como por ejemplo los usos del suelo, se almacenaría en cada celdilla un código numérico que representaría el tipo de uso existente en dicha celdilla.

Código Numérico	Tipo de Uso
1	Césped
2	Matorral
3	Bosque
4	Zonas urbanas

Tabla1.Elaboración propia. Equivalencia de códigos numéricos y tipos de uso del suelo para una variable cualitativa.

En el caso de las variables cuantitativas, como por ejemplo el caso de las pendientes, el número que se almacenaría se correspondería con el valor de la variable en esa celda.

Otra forma de representación del modelo rasters, es que registra el interior de los objetos en lugar de codificar sus fronteras, de tal forma que los límites quedan implícitamente representados por los límites de las celdillas que presentan un único valor. Mediante las siguientes tablas se puede entender mejor este proceso: (12)

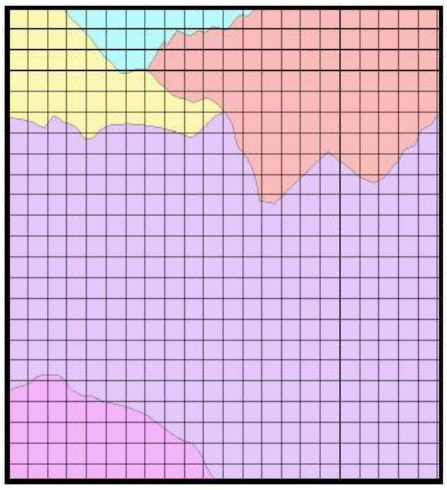


Figura 6-a. Malla rasters y usos del suelo.

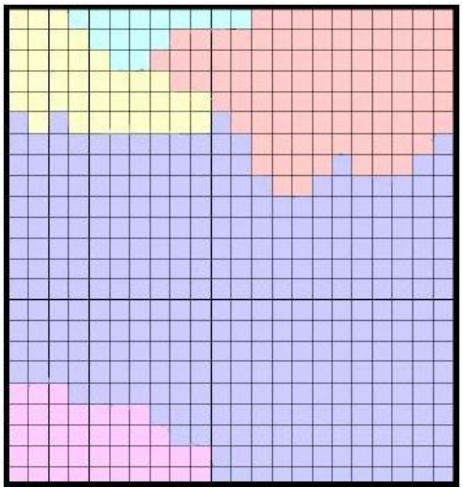


Figura 6-b. Codificación rasters de los usos del suelo.

Siguiendo con el ejemplo de los usos del suelo, se puede observar en la figura 6-a, que tras superponer la malla rasters y codificar los elementos existentes en el interior de cada pixel, se obtiene la imagen de la figura 6-b, en la que los límites de los usos del suelo quedan delimitados de forma implícita por los bordes de las celdillas, y de ahí su aspecto, que se denomina pixelado. Esta característica del modelo rasters permite representar mejor las variables continuas como son: elevaciones, precipitación, entre otras; que varían de forma gradual en el espacio y en las que no existen unos límites bien diferenciados de las diferentes entidades espaciales.

Un aspecto muy importante en este modelo son las celdillas o píxeles, estas son los elementos fundamentales del modelo. Las celdillas cumplen con ciertas características: son regulares, es decir todas son iguales y son localizables, cada celda puede ser identificada por su número de fila y columna. La celdilla es la unidad mínima en un archivo rasters.

Las imágenes digitales obtenidas desde satélites y las fotografías, se almacenan en esta forma, cada celda de la matriz o arreglo de dos dimensiones, corresponden a

un pixel cuando se habla de la imagen y a un número o pareja ordenada (de acuerdo a las coordenadas de una celda respectiva) cuando se habla de valores almacenados dentro de la estructura.

Actualmente se disponen de varios métodos para la representación de la información, sea esta descriptiva o cuantificable y para trasladarla a una base de datos geográfica, entre estos medios de obtención están: las fotografías aéreas, imágenes de satélites y otros. A continuación se muestran algunos ejemplos.

Figura7.Fotografía Aérea. (9)

Figura8.Imagen de satélite. (9)

Los archivos vectoriales son representados utilizando el Modelo de Datos Vectorial, en este modelo, los objetos reales se suponen inscritos en un sistema coordenado, dicho sistema permite determinar con precisión, su posición, sus dimensiones y el número de estas. Así, en el modelo vectorial, los objetos se representan mediante la especificación de los valores de estas magnitudes. Para representar un rasgo o fenómeno geográfico particular se usan dos componentes: la entidad (componente descriptivo no gráfico) y la representación geométrica (componente gráfico).

Una entidad es un objeto distinguible de lo que le rodea, acerca de la cual se requiere información, además es la representación digital del componente descriptivo de un rasgo geográfico. Se le asocia un nombre con el fin de distinguirla de otras entidades; ejemplos: carretera, presa, línea de transmisión, área agrícola, etc.

La representación geométrica constituye la representación digital del componente espacial de un rasgo geográfico, existen tres tipos de representación geométrica las cuales ya han sido explicadas anteriormente, estas son: punto, línea, área o polígono. Cada tipo de representación está definida en dos dimensiones (X, Y). La representaciones geométricas pueden relacionarse entre sí, esto se logra interceptando cada uno de los rasgos geográficos involucrados (ver figura 9).

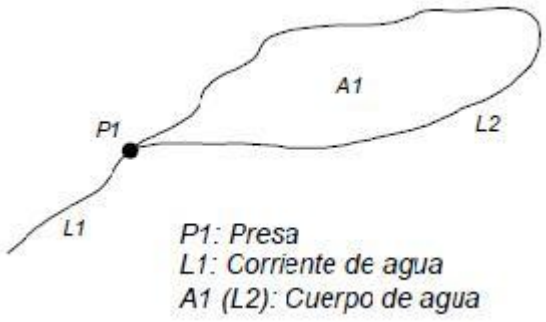


Figura9.Conectar punto, línea, área. (13)

Se han presentado los dos modelos que permiten representar objetos del mundo real. Estos modelos de datos presentan un conjunto de ventajas y desventajas el uno con respecto al otro, estos modelos son complementarios y las ventajas de uno se convierten en los inconvenientes del otro. En la siguiente tabla se describen de manera sintética las ventajas y desventajas de cada uno de ellos.

VENTAJAS E INCOVENIENTES RASTER/VECTORIAL	
VENTAJAS	
RASTER	VECTORIAL
Estructura de datos muy simple	Elevada precisión
Facilidad para la representación de entidades espaciales continuas	Facilidad de integración con softwares CAD vectoriales.
Elevada capacidad para la superposición y combinación de capas	Mapas de elevada calidad
Capacidad para realizar análisis geostadísticos	Capacidad para realizar análisis de redes
Capacidad de integrar datos de satélite	Estructuras de datos con topología
Capacidad de incorporación de imágenes	Ficheros de bajo tamaño
INCONVENIENTES	
RASTER	VECTORIAL
Baja precisión (dependiendo del tamaño de píxel)	Estructuras de datos complejas
Dificultad de integración con CAD. Entorno de trabajo diferente al software CAD	Dificultad para representar entidades espaciales continuas
Mapas de menor calidad	Menor capacidad para combinación de capas (especialmente aritmética)
Incapacidad para realizar análisis de redes	Incapacidad para realizar análisis estadísticos espaciales
Estructuras de datos sin topología	Dificultad para integrar datos de satélite
Ficheros de elevado tamaño	Incapacidad para el tratamiento de imágenes

Tabla 2. Ventajas e Inconvenientes de los Modelos Rasters y Vectorial. (12)

1.5. Estado del Arte.

En la actualidad los catálogos son herramientas que facilitan el trabajo en diferentes esferas de la sociedad. El catálogo más antiguo del cual se tiene noticias es una pequeña tablilla de arcilla de más o menos 4 x 6 cm. Esta pequeña tablilla contiene referencia de unos sesenta títulos, es decir, en ella están los primeros renglones del texto, ya que en esa época las obras carecían de título. La misma data de más de 2 000 años a. n. e: y fue confeccionado por los sumerios. Ya en el I milenio, siglo VII a. n. e., en las tablas del rey asirio Asurbanipal, aparecen al final del texto, algunos datos bibliográficos como título, número de la tabla o tomo, primeros renglones de la siguiente tabla, nombre del propietario original y nombre del copista, en caso de tratarse de una copia. La biblioteca de Asurbanipal fue muy famosa y de ella se conservan en el Museo Británico alrededor de 22 000 tablas de arcilla cocida. Ya en la Edad Media los catálogos medievales servían más para los monjes, que los custodiaban, que para el público y como el número de manuscritos era pequeño, no era un problema su organización. El préstamo era sólo bajo seguridad dado el gran

valor que tenían las obras. El énfasis en el valor material de los libros continuaría hasta el siglo XVI.

Los datos bibliográficos que se conservaban constituían sobre todo una especie de inventario, con el nombre de los autores y títulos de las obras. Y en muchos monasterios las listas de libros no eran solamente con la finalidad de hacer el catálogo sino para llevar un control de las donaciones recibidas. Los catálogos estaban formados por listas de las obras, según aparecían en los estantes e incluían guías de contenido de cada obra o instrucciones de cómo debían ser usados.

Con la Era Digital los catálogos dieron un cambio radical, en las empresas que se comercializaban productos se empezaron a utilizar catálogos digitales, siendo estos más accesibles a clientes de cualquier parte del mundo. En el mundo digital actual existen diferentes tipos de catálogos digitales, los cuales son utilizados para guardar archivos y objetos de todo tipo. En las bibliotecas se utilizan mucho los catálogos para tener un control estricto de las publicaciones. En la Universidad de Ciencias Informática existe un catálogo online que da la posibilidad al usuario que lo utiliza de búsquedas rápidas de libros digitales, revistas, documentos de tesis y artículos, filtrando criterios de búsquedas como nombre, tipo, entre otros. En el mundo, las grandes bibliotecas constan de catálogos modernos que les permiten a sus visitantes conocer de forma rápida sus publicaciones, además de otras informaciones del interés de este; un ejemplo claro se puede ver en la biblioteca de la Real Academia de Derecho Civil en España, la cual consta de un catálogo que lista los individuos y jubilados que componen la Real Academia.

No solo en las bibliotecas existen catálogos, en las empresas de software, de equipos electrónicos, etc. el catálogo ayuda a expandir más el comercio, convirtiéndose en una forma más de compra, y en muchas otras esferas los catálogos son una herramienta muy útil que facilitan el acceso a productos.

Estos catálogos le dan la posibilidad al internauta de acceder a las tiendas virtuales, buscar productos y comprar productos sin necesidad de ir personalmente a la tienda. En el mundo del software el catálogo es una herramienta que almacena los últimos avances de esta industria y da a conocer sus nuevos productos. El Catálogo de Software Libre de la comunidad de desarrolladores de Linux brinda la opción de conocer los nuevos productos desarrollados, además ofrece una descripción de estos productos, por quien fueron creados y el tamaño para la descarga.

Todos los catálogos tienen una característica en común y es que tienen una relación organizada de elementos pertenecientes al mismo conjunto, que por su número precisan de una catalogación para facilitar su localización. Es cierto que hasta el momento no existe un catálogo que catalogue los archivos provenientes de dispositivos GPS, pero sus funciones más básicas serían las mismas que cualquier otro catálogo digital.

1.6. Conclusiones.

En este capítulo se abordó a grandes rasgos el auge de las nuevas tecnologías en el mundo, se analizó el objeto de estudio y el estado del arte del sistema a desarrollar, dando como resultado una visión más amplia de los objetivos de la investigación, el establecimiento de un marco teórico ayuda a prevenir y a minimizar posibles errores que pueden presentarse durante el estudio, al formular este capítulo queda orientado el modo de conducir la investigación. Se llegó a la conclusión que la manera más sencilla y menos trabajosa de tener una lista organizada de los archivos contenedores de datos provenientes de dispositivos GPS es mediante un catálogo y así tener acceso a ellos de forma rápida y fácil para su mejor manipulación.

Capítulo 2. Propuestas y Selección de Tecnologías.

2.1. Introducción.

En este capítulo se exponen las características de las metodologías y herramientas de desarrollo empleadas para dar solución al problema y se realiza la selección adecuada de las mismas.

2.2. Lenguajes de Programación.

En la actualidad existen diferentes lenguajes de programación, estos han surgido y se han incrementado según las tendencias y las necesidades de las plataformas. Los lenguajes pueden ser de alto o bajo nivel. En los de bajo nivel las instrucciones son simples y cercanas al funcionamiento de la máquina, por ejemplo el código máquina y el ensamblador. En los lenguajes de alto nivel hay un alto grado de abstracción y el lenguaje es más próximo al de los humanos. El progreso de las tecnologías y la aparición de nuevas incógnitas dieron lugar a desarrollar lenguajes de programación que permitieran interactuar con usuarios y utilizaran sistemas de bases de datos. A continuación se mencionan los lenguajes de programación más utilizados en el desarrollo de aplicaciones de escritorio y web, resulta de importancia citar que algunos de ellos pueden utilizarse para desarrollar ambos tipos de aplicaciones. Se añade además que algunos de los lenguajes que se proponen son propietarios o privativos y otros libres, lo que se hace de forma intencional para llegar a comparaciones entre ellos, teniendo en cuenta esa característica que hoy día es muy polémica, se presentan también características, ventajas y desventajas de los lenguajes analizados.

2.2.1. PHP.

Surge en 1995, se desarrollo por el grupo PHP Group. Inicialmente se conoció como Personal Home Page (página personal principal), en la actualidad es Hypertext Pre-prosessor. Es un lenguaje script interpretado del lado del servidor utilizado para la generación de páginas web dinámicas. No necesita ser compilado para ejecutarse. PHP es un potente lenguaje de secuencia de comandos diseñado específicamente para permitir a los programadores crear aplicaciones Web con distintas prestaciones de forma rápida (14), tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX y Windows, y puede interactuar con los servidores Web más populares. Es uno de los lenguajes de programación más populares por la gran fluidez y rapidez de sus scripts. Parte de su sintaxis ha sido tomada de C, Java, Perl y otras son características específicas. Sus archivos tienen extensiones php. Presenta las siguientes ventajas y desventajas.

Ventajas:

- Fácil entendimiento.
- Rapidez.
- Soporta hasta cierto punto la orientación a objetos, clases y herencias.
- Es multiplataforma. Windows, Linux y otros.
- Tiene capacidad de conexión con la mayoría de manejadores de bases de datos.
- Capacidad de expandir su potencial utilizando módulos.
- Es libre, por lo que todos pueden acceder con facilidad.
- Posee un elevado número de funciones.

Desventajas:

- Es necesario instalar un servidor web.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La programación orientada a objetos es deficiente para aplicaciones muy grandes.

2.2.2. ASP.NET.

Este es un lenguaje comercializado por Microsoft, y usado por programadores para desarrollar entre otras funciones, sitios web. ASP.NET es el sucesor de la tecnología ASP, fue lanzada al mercado mediante una estrategia de mercado denominada .NET.

El ASP.NET fue desarrollado para resolver las limitantes que brindaba su antecesor ASP. Creado para desarrollar web sencillas o grandes aplicaciones. Para el desarrollo de ASP.NET se puede utilizar C#, VB.NET o J#. Los archivos cuentan con la extensión (aspx). Para su funcionamiento de las páginas se necesita tener instalado IIS con el Framework .Net. Microsoft Windows 2003 incluye este frameworks, solo se necesitará instalarlo en versiones anteriores.

Ventajas:

- Completamente orientado a objetos.
- Facilita el mantenimiento de grandes aplicaciones.
- Incremento de velocidad de respuesta del servidor.

- Mayor velocidad.
- Mayor seguridad.

Desventajas:

- Mayor consumo de recursos.
- Propietario.

2.2.3. Java.

El lenguaje de programación Java, fue diseñado por la compañía Sun Microsystems Inc., con el propósito de crear un lenguaje que pudiera funcionar en redes computacionales heterogéneas (redes de computadoras formadas por más de un tipo de computadora), y que fuera independiente de la plataforma en la que se vaya a ejecutar. Esto significa que un programa de Java puede ejecutarse en cualquier máquina o plataforma. La compañía Sun describe el lenguaje Java como simple, orientado a objetos, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico. (15)

Simple: Java es un lenguaje sencillo, con pocas palabras reservadas que hacen que su aprendizaje sea rápido. Además su utilización es independiente del tipo de aplicación que se desee desarrollar por lo que una vez aprendido es fácil aplicarlo en la realización de aplicaciones de escritorio.

Orientado a objeto: El paradigma de la Programación Orientado a Objetos (en lo adelante POO) permite el encapsulamiento, la herencia, y el polimorfismo, facilitando así el diseño y la implementación.

Interpretado: Al ser un lenguaje interpretado es independiente de la arquitectura en la que se vaya a ejecutar.
Multitarea: Java incorpora mecanismos para que el software desarrollado pueda ser multitarea como el uso de diferentes hilos de ejecución.
Robusto: Al no trabajar directamente con punteros y posiciones de memoria se evita que el programa intente acceder a zonas que no le corresponden.
Dinámico: Java está en constante evolución, consiguiéndose mejores rendimientos e incorporando nuevas funcionalidades a medida que Sun proporciona nuevas versiones.
Portable: El código generado es independientemente de la plataforma, solo depende de la máquina virtual donde se ejecuta.

Seguro: Los programas escritos en Java no acceden directamente a los recursos de la máquina física en la que se ejecutan sino que se ejecutan en una máquina virtual, de manera que el entorno está más controlado.

De arquitectura neutra: Como el programa en Java no se ejecuta directamente sobre la máquina sino que se ejecuta en la máquina virtual es indiferente la arquitectura real donde se ejecuta esta.

De altas prestaciones: Java cuenta con gran número de paquetes muy probados para resolver diferentes problemas o satisfacer las necesidades de componentes de cualquier desarrollo. Esto permite un buen rendimiento utilizando elementos estándar del lenguaje.

Ventajas:

- Lenguaje con alta portabilidad.
- Amplia disponibilidad de IDE's.
- Plataforma capaz de generar aplicaciones de escritorio robustas.
- Es sencillo y con mucho código disponible en la Web.
- Múltiples frameworks.
- Es multiplataforma.

Desventajas:

- En móviles, no hay una estandarización de la máquina virtual, por lo que se requiere escribir mucho código para saber que bibliotecas se pueden o no utilizar.
- La máquina virtual es muy pesada sobre algunos sistemas operativos.

2.2.4. C#.

Fue creado por Anders Heljsberg, quien además creó otros lenguajes y entornos como Turbo Pascal, Delphi o Visual J++. Es un lenguaje de propósito general orientado a objetos creado por la compañía Microsoft para su plataforma .NET. Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET el cual es similar al de Java aunque incluye mejoras derivadas de otros lenguajes. El lenguaje C# fue diseñado para combinar el control a bajo nivel de lenguajes como C y la velocidad de programación de lenguajes como Visual Basic.

Entre las principales características se encuentran: la facilidad de uso, además el ambiente de trabajo es muy cómodo, amigable y clásico en las aplicaciones de Windows. Este lenguaje presenta las características necesarias para considerarlo

como un lenguaje orientado a objetos, tales son: encapsulación, herencia y polimorfismo, esta forma de programación hace que el código sea reutilizable. Permite además la administración de memoria, la seguridad en el manejo de datos, es compatible con otros lenguajes y permite el uso de operadores.

Ventajas:

- Combina los mejores elementos de múltiples lenguajes de amplia difusión.
- Es uno de los 10 lenguajes más utilizados a nivel mundial.
- Eliminación del uso de punteros.
- Es multiplataforma.
- Es totalmente Orientado a Objetos.

Desventajas:

- Para hacer uso de este lenguaje es necesario una versión reciente de Visual Studio.NET.
- El sistema tiene que tener algunos requisitos como Windows NT 4 o superior, tener alrededor de 4 gigas de espacio libre para la pura instalación
- Es importante estar familiarizado con otros lenguajes, de lo contrario costará un poco de trabajo su uso.

2.3. Selección del lenguaje de programación a utilizar.

Como se puede observar cada uno de estos lenguajes ofrece diferencias en sus características, no se puede decir que tan superior o inferior es uno de otro pues se han desarrollado excelentes y malas aplicaciones con cada uno de ellos. De manera general no existe un lenguaje mejor que los demás, pero si puede suceder que algunos de ellos sean los más adecuados para ciertas aplicaciones, no se puede evaluar un lenguaje por su costo, fama y facilidad de código.

Por tanto se escoge como lenguaje de programación a Java por ser el lenguaje ideal para el desarrollo de la aplicación, pues este tiene como principal objetivo el funcionamiento fuera de la plataforma donde se ejecuta. Es un lenguaje moderno que elimina la complejidad que existe en otros tipos de lenguajes. Tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir muchos errores y características que resultan complejas e inseguras como la utilización de punteros, por lo que es considerado un lenguaje más fiable. Es totalmente orientado a objetos, lo cual constituye un modelo de programación más limpio y seguro.

Es un lenguaje fácil de aprender y de forma sencilla, implementa barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real. Soporta sincronización de múltiples hilos de ejecución para crear programas multitareas, existen además varios entornos de desarrollo totalmente gratuitos para desarrollar con este lenguaje, como son: NetBeans y Eclipse, esto sirve de gran ayuda a la actual migración en la que está enfrascada la universidad de Windows a Linux. Cada vez incorpora más facilidades para la creación y manipulación de gráficos, así como el acceso a base de datos. Es fiable, numerosas empresas han desarrollado excelentes proyectos por su robustez, diseño y fácil portabilidad.

El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de este uno de los lenguajes con mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática por lo que es recomendable para realizar excelentes aplicaciones escritorio.

2.4. Metodologías de Desarrollo.

Las Metodologías de Desarrollo de Software son conjuntos de procesos, procedimientos, técnicas y políticas que intervienen en la creación del software. En ellas se indican paso a paso las actividades a realizar para garantizar un producto eficiente que cumpla a cabalidad con los requisitos inicialmente planteados; también indican las personas que deben participar y el papel que deben desempeñar dentro del proceso de desarrollo. Define ¿qué hacer y quien debe hacerlo?, ¿Cuándo y cómo hacerlo? Caracteriza a un proyecto en cuanto equipo de desarrollo, recursos etc.

Entre las más usadas están: el Proceso Unificado de Software (RUP), Programación extrema (XP) y Microsoft Solution Framework (MSN).

2.4.1. Programación Extrema (XP).

Es una metodología empleada en proyectos de corto plazo y pequeño equipo de trabajo. Consiste en una programación rápida o extrema. Su distintivo es tener al usuario final como parte del equipo. Tiene objetivos primarios bien definidos: Satisfacer al cliente brindando el producto de software con calidad y en tiempo real y enfatizar en el trabajo en equipo. Tanto los jefes de proyecto, los clientes y desarrolladores, son parte del equipo y están involucrados en el desarrollo del software. XP es más una filosofía de trabajo que una metodología porque se tiene presente las necesidades del cliente y el tiempo en que este necesita el producto; las necesidades del usuario siempre están en la mente de los desarrolladores a la hora de implementar el producto. (16)

2.4.2. Microsoft Solution Framework.

Es una metodología relacionada con una serie de conceptos, modelos y prácticas de uso que controlan; la planificación, el desarrollo y la gestión de proyectos tecnológicos. Es adaptable, escalable y flexible. Más que una metodología rígida de administración de proyectos, MSF es una serie de modelos que pueden adaptarse a cualquier proyecto de tecnología de información y además están encargados de planificar las diferentes partes implicadas en el desarrollo de dicho proyecto. Contiene 8 principios básicos:

1. Originar comunicaciones abiertas
2. Trabajar para una visión compartida
3. Fortalecer los miembros del equipo
4. Establecer responsabilidades claras y compartidas
5. Centrar el foco en agregar valor al negocio
6. Permanecer ágil y esperar cambios
7. Invertir en calidad
8. Aprender de todas las experiencias (17)

2.4.3. RUP.

Es un proceso de desarrollo de software orientado a objeto. Es el resultado de la unión de técnicas y trabajo de varias metodologías utilizadas por los clientes que se adaptan a las necesidades de cada organización. Es además un proceso porque define quién está haciendo qué, cuándo lo hace y cómo alcanzar cierto objetivo, en este caso el desarrollo de software. En su modelación define como sus principales elementos los trabajadores, actividades, artefactos y el flujo de actividades. Se caracteriza por una serie continua de iteraciones y fases donde se realiza un estudio detallado de los requerimientos más críticos que se plantean, seguido de un amplio, profundo análisis y diseño a un alto nivel de desarrollo, quedando así bien definida la arquitectura del sistema. La arquitectura está basada en componentes, flexible, fácil de modificar y promueve la reutilización de componentes, está estructurada por tres capas, la capa de interfaz, la de negocio y la de acceso a datos. RUP usa para la representación de la información UML: lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. UML es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software, ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios, funciones del sistema y aspectos

concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema, para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software tal como el Proceso Unificado Rational.

Esta metodología ofrece una reducción de riesgos basado en la retroalimentación temprana, pruebas continuas e iterativas que promueven una mejor evaluación del estado del proyecto, además permite a los clientes recibir evidencia concreta del avance del mismo, también posibilita que los problemas más complejos se ataquen primero. Esta metodología genera una gran documentación, ideal para el proyecto. Por sus características el lenguaje de modelado a utilizar en el desarrollo del sistema es el UML. Con sus diagramas unidos se puede representar la arquitectura del sistema. UML introduce nuevos diagramas que señalan la visión dinámica del sistema. Es muy útil pues implementa un lenguaje de modelado evitando la propiedad del código por los desarrolladores, es decir, crea una documentación común para todos los desarrollos. Cualquier desarrollador con conocimientos de UML es capaz de entender perfectamente independientemente del lenguaje que desarrolle. Se ha convertido en un estándar de diseño orientado a objetos.

Son definidos 9 flujos de trabajos principales; 6 conocidos como flujos de ingeniería y 3 como flujos de apoyo.

Flujos de Trabajo:

- Modelamiento del negocio
- Requerimientos
- Análisis y diseño
- Implementación
- Prueba
- Instalación
- Administración de proyecto
- Administración de configuración y cambio
- Ambiente

Organiza las tareas en fases e iteraciones, que tiene como finalidad el establecimiento de hitos.

Fases	Hitos
• Inicio	Objetivos (visión)
• Elaboración	Arquitectura
• Construcción	Funcionalidad Operativa
• Transición	Release del Sistema

2.5. Selección de la Metodología a utilizar.

La metodología que más se adapta a las condiciones de la aplicación que se quiere desarrollar es RUP, puesto que recoge un grupo de buenas prácticas de muchas otras metodologías y la organización del trabajo es un aspecto fundamental, además la documentación que se genera es de gran apoyo a las siguientes iteraciones que se van a desarrollar en un futuro, permitiendo que el equipo de desarrollo del proyecto Control de Flotas esté involucrado en todo momento en futuras versiones de la aplicación. Esta metodología propicia un enfoque de trabajo organizado, donde se controla y documenta todo lo relacionado con el desarrollo del software y pueden eliminarse los riesgos que podrían presentarse en el mismo, lo cual no pudiera lograrse sin el empleo de una metodología eficaz que se adapte a estas características. RUP se va a centrar en la definición detallada de los procesos y tareas a realizar con una gestión cuidadosa de requisitos, que es necesario realizar para el desarrollo del sistema que se propone, mientras que las metodologías ágiles como XP son aplicables en entornos volátiles, los requisitos no se conocen con exactitud y están sujetos a variaciones constantes. Esta metodología está preparada para desarrollar proyectos complejos gracias a las características que presenta, por ejemplo:

Dirigido por caso de uso: Basándose en los casos de usos, los desarrolladores crean unos modelos de diseño e implementación que los llevan a cabo. Estos modelos se validan para que sean conformes a dichos casos, los cuales también sirven para realizar las pruebas sobre los componentes desarrollados.

Centrado en la arquitectura: En la arquitectura de la construcción, antes de construir un edificio este se contempla desde varios puntos de vista: estructura, conducciones eléctricas, fontanería, etc. Cada uno de estos aspectos está

representado por un gráfico con su notación correspondiente. Siguiendo este ejemplo, el concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema.

Iterativo e incremental: Todo sistema informático complejo supone un gran esfuerzo que puede durar desde varios meses hasta años. Por lo tanto, lo más práctico es dividir un proyecto en fases. Actualmente se suele hablar de ciclos de vida en los que se realizan varios recorridos por todas las fases. Cada recorrido por las fases se denomina iteración en el proyecto en la que se realizan varios tipos de trabajo (denominados flujos). Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada.

2.6. Herramientas.

2.6.1. Herramientas CASE.

Las herramientas CASE (Ingeniería de Software Asistida por Computadoras) son aplicaciones informáticas creadas con el objetivo de aumentar la productividad en el desarrollo de software disminuyendo tiempo y dinero, incrementando la velocidad del desarrollo y la calidad del sistema. También ayuda en la documentación, la fiabilidad, la utilidad y en el rendimiento. Además proveen beneficios durante todas las etapas del proceso de desarrollo de software, verificando el uso de todos los elementos en el sistema diseñado. Automatizan el dibujo de diagramas, ayudan en la creación de las relaciones en las bases de datos y hacen que el trabajo de diseño sea más fácil y agradable.

Dentro de las herramientas CASE más utilizadas se encuentran las siguientes:

2.6.1.1. Rational Rose.

Es una herramienta desarrollada por Rational Corporation. Implementada por los creadores de UML. Cubre todo el ciclo de vida de un proyecto. Permite crear diagramas que se van generando durante el proceso de ingeniería en el desarrollo de software. Presenta un número de estereotipos predefinidos que facilitan la modelación. Es capaz de generar el código fuente de las clases definidas en el flujo de trabajo de diseño. Presenta algunas desventajas; no es soportado por algunos lenguajes de programación y no posibilita exportar hacia un sistema gestor de bases de datos el diagrama de clases persistentes a partir del cual se genera la base de datos del sistema.

2.6.1.2. Visual Paradigm.

Es una herramienta diseñada para el desarrollo de software que utiliza el estilo de programación orientado a objeto, busca reducir el tiempo de desarrollo apoyando a arquitectos, analistas, diseñadores y desarrolladores. Tiene gran aceptación entre la comunidad de desarrolladores de aplicaciones informáticas, aspecto que lo distingue entre otras herramientas utilizadas para el modelado como pueden ser Argo UML y Umbrello (18). Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad y a un menor coste.

Se pueden encontrar distintas versiones:

- Enterprise
- Profesional
- Standard
- Modeler
- Personal
- Community

Esta herramienta puede soportar capacidades de ingeniería inversa y directa. Entre sus características principales se encuentran las siguientes:

- Tiene licencia gratuita y comercial.
- Genera código en diferentes lenguajes de programación.
- Se puede integrar con diferentes IDE.
- Genera documentación del modelado.
- Es fácil de instalar y actualizar.

2.7. Selección de la herramienta CASE a utilizar.

Por tanto, se decidió utilizar Visual Paradigm 3.4 para UML 6.4 Enterprise Edición. Esta herramienta visual permite construir la aplicación con mayor rapidez, mayor exactitud, mejor trabajo en equipo y aumenta además las expectativas mediante la interfaz gráfica. Además, la herramienta es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos y permite controles de versiones.

2.8. Entorno de desarrollo. NetBeans_IDE 6.8.

Un entorno de desarrollo integrado o en inglés Integrated Development Environment ('IDE') es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. (19)

Los dos entornos de desarrollo más potentes que existen en la actualidad para la programación en java son Eclipse y NetBeans. Ambos son gratuitos y poseen funcionalidades comunes que permiten a los programadores realizar cualquier operación. Por su parte Eclipse posee una serie de plugins con los cuales brinda una mayor funcionalidad al entorno de desarrollo.

En el desarrollo de la aplicación se decidió utilizar NetBeans pues es superior al Eclipse en cuanto al entorno gráfico y la interfaz de usuario, dos aspectos fundamentales a tener en cuenta en el producto a desarrollar.

El NetBeans IDE es un entorno de desarrollo, una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

2.9. La Plataforma Java.

La plataforma refiere el ambiente de hardware y software en donde el programa se ejecuta, por ejemplo, plataformas como Windows, Solaris y Linux. En casi todos los casos las plataformas son descritas como la combinación de un sistema operativo y el hardware. La plataforma Java se diferencia de estas, pues es solo de software y se ejecuta sobre las otras plataformas de hardware. La plataforma Java está compuesta por la Máquina Virtual de Java (JVM) y las API de Java.

La JVM es una de las piezas fundamentales de la plataforma Java. Básicamente se sitúa en un nivel superior al Hardware del sistema sobre el que se pretende ejecutar la aplicación, y este actúa como un puente que entiende tanto el bytecode, como el sistema sobre el que se pretende ejecutar. Así, cuando se escribe una aplicación Java, se hace pensando que será ejecutada en una máquina virtual Java en concreto, siendo esta la que en última instancia convierte de código bytecode a código nativo del dispositivo final.

El API Java es una Interfaz de Programación de Aplicaciones (API: por sus siglas en ingles) provista por los creadores del lenguaje Java, y que da a los programadores los medios para desarrollar aplicaciones Java.

Como el lenguaje Java es un Lenguaje Orientado a Objetos, la API de Java provee un conjunto de clases utilitarias para efectuar toda clase de tareas necesarias dentro de un programa. Además está organizada en paquetes lógicos, donde cada uno de ellos contiene un conjunto de clases relacionadas semánticamente.

La plataforma seleccionada para el desarrollo de la aplicación fue jdk-6. Esta plataforma contiene entre sus paquetes de desarrollo una nueva versión del Sun Java Runtime Environment 6 Update 2, necesaria para ejecutar todos los programas que utilicen Java, esta característica es muy utilizada en la aplicación; para ejecutar los archivos provenientes de los dispositivos GPS.

2.10. Sistemas Gestores de Bases de Datos (SGBD).

El ser humano siempre ha tenido la necesidad de almacenar información de forma organizada, que le permita después acceder a ella para consultarla o modificarla. Esta información debe estar bien estructurada, ser consistente, sin repeticiones y de forma coherente. Se han creado entonces los SGBD que están diseñados para gestionar grandes volúmenes de información, así como simplificar y facilitar el acceso a los datos, haciendo que los tiempos de respuesta a las solicitudes de los usuarios sean muy reducidos y el acceso a la información se pueda realizar de forma concurrente; además estos SGBD garantizan una eficiente seguridad en el tratamiento de los datos.

En resumen los SGBD permiten:

- Definición de los datos.
- Mantenimiento de la integridad de la información dentro de la base de datos.
- Control de la seguridad y privacidad de la información.
- Manipulación de los datos.

A continuación se ejemplifican diferentes SGBD con sus correspondientes características.

2.10.1. MySQL.

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma eficiente. Fue creado por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca. Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL. Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración. (20)

Brinda las siguientes ventajas y desventajas:

Ventajas:

- Acceso a las bases de datos de forma simultánea por varios usuarios y/o aplicaciones.

- Seguridad: Determinados usuarios tendrán permiso para consulta o modificación de determinadas tablas en forma de permisos y privilegios, lo que permite compartir datos sin que peligre la integridad y protegiendo determinados contenidos.

- Escalabilidad: Es posible manipular bases de datos enormes, del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla.

- Conectividad: Permite conexiones entre máquinas con distintos sistemas operativos.

- Recuperación automática ante fallas: Si se da de baja de forma anormal, no suele perder información ni corromper los datos y completa las transacciones que no se terminaron.

- Utiliza el potente lenguaje de consultas SQL.

Desventajas:

- No permite el modo de autentificación local (seguridad integrada de Windows), sólo el modo estándar.

- No tiene un panel de control gráfico y detallado.

2.10.2. PostgreSQL.

PostgreSQL es un SGBD Objeto-Relacional basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre de este proyecto, y utiliza el lenguaje SQL. Fue pionero en muchos de los conceptos del sistema objeto-relacional actual. Este proyecto lleva más de una década de desarrollo, siendo hoy día, el sistema libre más avanzado, soportando la gran mayoría de las transacciones SQL y control concurrente.
A diferencia de la mayoría de los sistemas de bases de datos que usan bloqueos para el control de concurrencia, PostgreSQL mantiene la consistencia de los datos en un modelo multiversión. Esto significa que mientras se consulta una base de datos, cada transacción mantiene una imagen o versión de los datos, sin tener en cuenta el estado actual de la información.

Entre sus principales ventajas y desventajas se destacan las siguientes:

Ventajas:

- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

- Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.

- Posee estabilidad y confiabilidad legendarias.

- Es extensible a través del código fuente disponible sin costos adicionales.

- Es multiplataforma, disponible en la mayoría de los sistemas operativos.

- Permite implementar reglas, vistas, disparadores, subconsultas y procedimientos almacenados.

- Posee herramientas para generar SQL portable para compartir con otros sistemas compatibles con SQL.

Desventajas:

- La velocidad de respuesta que ofrece con bases de datos relativamente pequeñas puede parecer un poco deficiente, pero la mantiene al gestionar bases de datos realmente grandes.

- Consume muchos recursos y carga con mucha facilidad el sistema.

2.11. Selección del Sistema Gestor de Base de Datos a utilizar.

Dentro de los SGBD estudiados se seleccionó como el más apropiado PostgreSQL en su versión 8.2, este posee características de código abierto, constituyendo uno de los SGBD libres más avanzado. Es el que más se adecua a las necesidades de la aplicación, para almacenar grandes volúmenes de datos y permitir el acceso de diferentes usuarios al mismo tiempo, acreditando permiso a cada uno de ellos. Es multiplataforma disponible en diferentes sistemas operativos y orientado a objetos, esto permite que cada uno de los componentes de la estructura de la base de datos pueda ser tratado como un objeto, característica de suma importancia en el sistema que se implementará.

2.12. Conclusiones.

Al concluir el presente capítulo se tiene un conocimiento más amplio de las distintas tecnologías y herramientas que se utilizan para el desarrollo del sistema y sus características. Así como las ventajas y desventajas de las mismas. Se realizó la selección adecuada y justificada de las tendencias y tecnologías. Para realizar esta selección se tuvo en cuenta que las herramientas se ajustaran al desarrollo de la aplicación y su libre distribución.

Capítulo 3. Presentación de la solución propuesta.

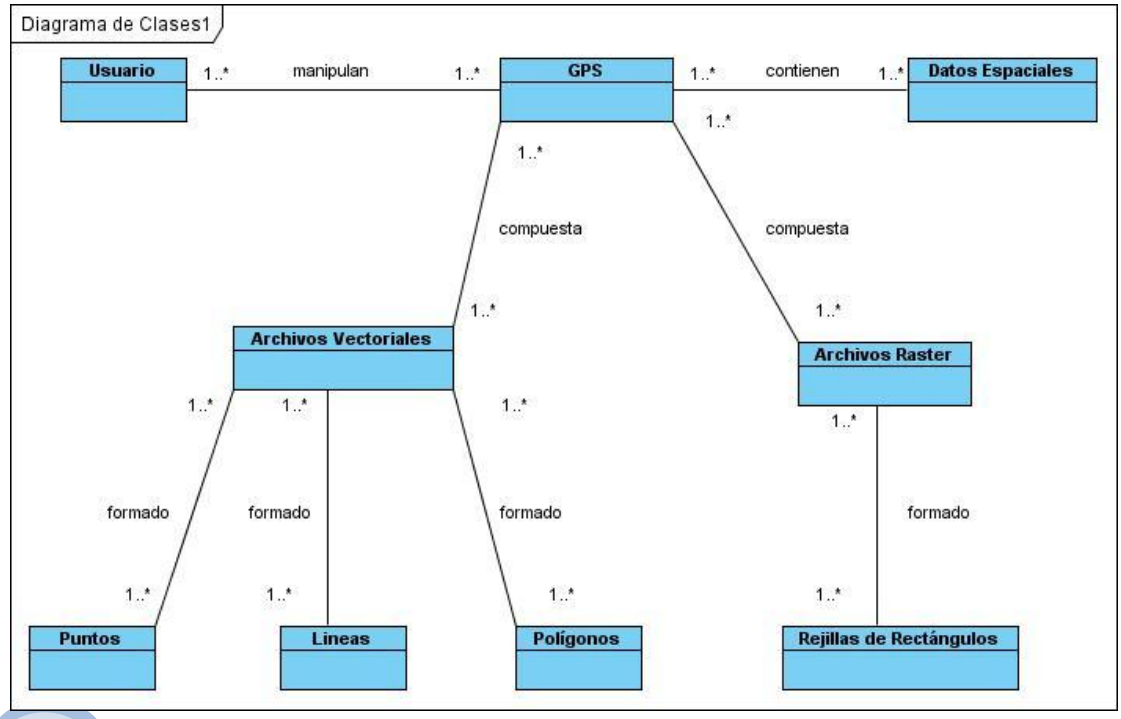
3.1. Introducción.

En este capítulo se comienza el desarrollo de la aplicación, se describen las principales funcionalidades que el sistema debe cumplir para alcanzar resultados exitosos, se exponen los principales conceptos del entorno a través del modelo de dominio así como la descripción textual de cada uno de estos conceptos y se identifican y establecen un conjunto de requisitos tanto funcionales como no funcionales que forman la base del sistema.

3.2. Modelo de dominio.

Un modelo de dominio, captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema. (21)

Debido a que no se tienen bien definidas las fronteras del negocio se propone realizar el modelo del dominio, el cual tiene como premisa fundamental comprender y describir las clases más importantes dentro del contexto del sistema e identificar todos los conceptos que se utilizarán en el diagrama, mediante un glosario de términos sobre los nombres que ayudará a los usuarios a mantener una terminología común, lo que les permite compartir el conocimiento de forma más consistente. A continuación se muestra el diagrama de clases del modelo de dominio.



3.2.1. Glosario de Términos del Negocio.

Usuario:

Persona que interactúa con el sistema.

GPS:

El Sistema de Posicionamiento Global (GPS) es un sistema de navegación basado en la localización mediante satélites.

Datos Espaciales:

Los Datos Espaciales son informaciones sobre la localización y las formas de un objeto geográfico y las relaciones entre ellos, normalmente con coordenadas y topología, es el tipo de datos necesarios para crear mapas y estudiar relaciones espaciales, son el componente fundamental de SIG. Dentro de su contexto, almacenan informaciones sobre la localización y las formas de un objeto geográfico y las relaciones entre ellos, normalmente con coordenadas y topología.

Archivos Vectoriales:

Los archivos vectoriales representan la información por medio de pares ordenados de coordenadas, este ordenamiento da lugar a las entidades universales con las que se representan los objetos gráficos, sus datos se centran principalmente en la precisión de la localización de los elementos geográficos sobre el espacio. Estos archivos se representan a través de tres elementos geométricos: el punto, la línea y el polígono.

Punto:

Los puntos se utilizan para aquellas entidades geográficas que mejor pueden ser expresadas por un único punto de referencia. Los mismos transmiten la menor cantidad de información de estos tipos de archivo y no son posibles las mediciones. También se pueden utilizar para representar zonas a una escala pequeña.

Líneas:

Las líneas se utilizan en pequeñas escalas para representar polígonos, estos elementos lineales pueden medir distancia y son usadas para rasgos lineales: como curvas de nivel y ferrocarriles.

Polígonos:

Los polígonos bidimensionales tienen gran utilidad ya que representan elementos geográficos que cubren un área particular de la tierra, los mismos transmiten el mayor volumen de información en archivos vectoriales y en ellos se puede medir el perímetro y el área.

Archivos Rasters:

Los archivos rasters pueden ser cualquier tipo de imagen digital representada en mallas o rejillas en forma de rectángulos o cuadrados a los que se les denomina células o retículas, cada retícula posee información alfanumérica asociada que representa las características de la zona o superficie geográfica que cubre, divide el espacio en celdas regulares donde cada una de ellas representa un único valor, se componen de filas y columnas de celdas, estas almacenan un único valor.

Rejillas de Rectángulos:

Las rejillas de rectángulos o cuadrados a los que se les denominan células pueden ser cualquier imagen digital que es la representación de un archivo rasters.

3.3. Requisitos Funcionales.

Los requisitos funcionales son aquellos requerimientos del sistema (condiciones o capacidades) que desde el punto de vista de las necesidades del usuario, debe cumplir el sistema y que están fuertemente ligados a las opciones del programa. (22)
El sistema deberá ser capaz de realizar las funcionalidades siguientes:

RF1. Permitir al usuario identificarse ante el sistema.

El sistema permitirá que el usuario se autentique de forma segura accediendo a las funcionalidades ofrecidas.

RF2. Gestionar datos del archivo.

RF2.1. Permitir al usuario adicionar datos del archivo.

El sistema debe permitir adicionar datos del archivo en el servidor de Base de Datos.

RF2.2. Permitir al usuario modificar datos del archivo.

El sistema debe permitir modificar datos del archivo en el servidor de Base de Datos.

RF2.3. Permitir al usuario eliminar datos del archivo.

El sistema debe permitir eliminar datos del archivo en el servidor de Base de Datos.

RF3. Visualizar la información contenida en los archivos.

El sistema debe permitir visualizar el archivo en el servidor de Archivos.

RF4. Buscar Archivos.

El sistema debe permitir buscar datos del archivo de acuerdo al criterio de búsqueda seleccionado; en el servidor de Base de Datos.

RF4.1.Permitir al usuario realizar búsquedas de los archivos en dependencia de su extensión.

RF4.2. Permitir al usuario realizar búsquedas de los archivos en dependencia de su proveedor.

RF4.3. Permitir al usuario realizar búsquedas de los archivos tanto rasters como vectoriales por su fecha de adición.

RF5. Clasificar los archivos por el tipo de archivo.

El sistema debe permitir clasificar los datos del archivo en el servidor de Base de Datos.

3.4. Requisitos no Funcionales.

Los requisitos no funcionales especifican propiedades del sistema, como restricciones del entorno o la implementación, rendimiento o facilidad de mantenimiento, etc. (23)

Existen múltiples categorías para clasificar a los requisitos no funcionales.

Requisitos de interfaz externa:

El sistema deberá tener una interfaz externa amigable, que sea sencilla y fácil de entender por el usuario para así evitar que al usuario se le haga engorroso el trabajo con la misma, debe proveer de forma ordenada y detallada las funcionalidades del sistema y debe verse correctamente en todas las resoluciones.

Requisitos de usabilidad:

El sistema podrá ser usado por personas con conocimientos básicos en el manejo de computadoras. Debe poseer una interfaz amigable y que muestre de forma clara todas las funcionalidades que brinda la aplicación.

Requisitos de rendimiento:

Tanto el tiempo de respuesta como la velocidad de procesamiento de la información no deben exceder de los 5 segundos. El tiempo de procesamiento de los archivos no debe exceder de los 3 segundos.

Requisitos de soporte:

El sistema debe permitir posteriores modificaciones y actualizaciones a fin de alcanzar mayor funcionalidad.

Requisitos de seguridad:

La información manejada por el sistema estará protegida de acceso no autorizado, ya que el usuario del sistema debe autenticarse antes de realizar cualquier actividad en el mismo.

Requisitos de software:

El sistema dispone de los Sistemas Operativos Linux y Windows. Se requiere PostgreSQL como Sistema Gestor de Base de Datos. La Máquina Virtual de Java tiene que estar instalada.

Requisitos de hardware:

Se debe contar con una computadora de 1 GHz en adelante como velocidad del microprocesador, disponibilidad de espacio superior a los 20GB y una RAM mínima de 128 MB, aunque lo ideal sería 512 MB.

3.5. Descripción del sistema propuesto.

3.5.1. Descripción de los actores del sistema.

Un actor del sistema es una persona, sistema o entidad que interactúa con el sistema y que cumplen un rol determinado. Un actor no forma parte del sistema. En la siguiente tabla se muestra el actor del sistema **(Usuario)** y la justificación.

Actor	Descripción
Usuario	El usuario, es el encargado de realizar cualquier operación de búsqueda sobre los archivos rasters y vectorial según el criterio especificado, puede gestionar la información de los archivos y realizar cualquier otra operación que el sistema brinde.

Tabla 3. Descripción del actor del sistema.

3.5.2. Casos de Uso del Sistema.

Un caso de uso representa la forma de capturar los requisitos de un nuevo sistema. Estos casos de uso se representan mediante un diagrama de casos de uso, estos muestran la relación entre los actores y los casos de uso en un sistema. Una relación es una conexión entre los elementos del modelo.

3.5.2.1. Descripción textual de los casos de uso del sistema.

3.5.2.1.1. Caso de Uso Autenticar Usuario.

Caso de Uso:	**Autenticar Usuario**
Actores:	**Usuario**
Resumen:	El CU se inicia cuando el usuario introduce en el sistema el usuario y la contraseña, el sistema verifica la validez de estos datos para determinar si puede entrar al sistema. El usuario accede al sistema; finalizando así el caso de uso.
Referencias	**RF 1.**
Prioridad	**Crítico.**

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario introduce el usuario y la contraseña y da click en el botón “Entrar”.	2. El sistema verifica que los datos sean correctos y le permite al usuario acceder al sistema, además muestra un mensaje de la correcta autenticación.

Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	2.1. Si los datos son incorrectos el sistema muestra un mensaje de error “El usuario o la contraseña son incorrectos”. Pasa directamente a la acción 1.

3.5.2.1.2. Caso de Uso Gestionar datos del archivo.

Caso de Uso:	**Gestionar datos del archivo**
Actores:	**Usuario**

Resumen:	El CU se inicia cuando el usuario desea actualizar la base de datos para adicionar, modificar o eliminar los datos de los archivos tanto rasters como vectorial. Realizadas estas acciones finaliza el caso de uso.
Precondiciones	El usuario tiene que estar autenticado en el sistema.
Referencias	**RF 2, 2.1, 2.2, 2.3.**
Prioridad	**Crítico.**

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario accede a la opción “Gestionar”.	2. El sistema muestra una interfaz donde se podrá obtener el listado de todos los archivos. En la pantalla que se muestra se puede acceder a las acciones de: Adicionar datos del archivo, Modificar datos del archivo y Eliminar datos del archivo.
3. El usuario accede a la acción “Adicionar datos del archivo”, Ver la Sección 1, o a la acción “Modificar datos del archivo”, Ver la Sección 2, o a la acción “Eliminar datos del archivo”, Ver la Sección 3.	

Sección1” Adicionar Archivos”.	
1. El usuario accede a la acción “Adicionar datos del archivo.	2. El sistema muestra una interfaz para que el usuario entre los datos del archivo y así poder agregarlo.

3. El usuario introduce los siguientes datos del archivo: (código, nombre, extensión, fecha de adición, proveedor, fecha de modificación, tamaño, ubicación, tipo de archivo y da click en el botón “Adicionar”.	4. El sistema verifica que no exista un archivo con ese mismo código; ya que este es el único atributo que no se puede repetir, además verifica la validez de los datos y también comprueba que no se queden campos vacíos; excepto por el campo fecha de modificación que puede estar vacío. Por último el sistema muestra un mensaje de confirmación con las opciones “Si” y “No” para ratificar si realmente se quiere adicionar los datos.
5. El usuario da click en el botón “Si” para adicionar los datos del archivo, si no desea adicionar los datos del archivo solo debe dar click en el botón “No”.	6. El sistema dependiendo de la opción selecciona por el usuario adiciona o no los datos y termina el caso de uso.
Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	4.1. Si el código ya existe, o los datos son incorrectos, o se queda algún campo vacío; el sistema muestra un mensaje de error. Pasa directamente a la acción 3 de la sección 1.
Sección2 “Modificar Archivos”.	
Acción del Actor	Respuesta del Sistema
1. El usuario accede a la opción “Modificar datos del archivo”.	2. El sistema muestra una interfaz con un campo código y un botón “Buscar”, además muestra el listado de todos los archivos.

3. El usuario introduce el código del archivo a modificar y da click en el botón “Buscar”	4. El sistema verifica el campo código y muestra los datos del archivo.
5. El usuario modifica los datos (nombre, extensión, fecha de adición, fecha de modificación, tamaño, proveedor, ubicación y tipo), cambiando el valor del campo que desea alterar y da click en el botón “Cambiar”.	6. El sistema verifica que los datos a modificar son los correctos y muestra un mensaje de confirmación con las opciones “Si” y “No” para ratificar si realmente se quiere modificar los datos.
7. El usuario da click en el botón “Si” para modificar los datos del archivo, si no desea modificar los datos del archivo solo debe dar click en el botón “No”.	8. El sistema dependiendo de la opción selecciona por el usuario modifica o no los datos y termina el caso de uso.
Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	4.1. Si el código del archivo no existe muestra un mensaje de error. Pasa directamente a la acción 3 de la sección 2. 6.1. Si los datos son incorrectos o se queda algún campo vacío; el sistema muestra un mensaje de error. Pasa directamente a la acción 5 de la sección 2.
Sección3 “Eliminar Archivos”.	
Acción del Actor	Respuesta del Sistema
1. El usuario accede a la opción “Eliminar datos del archivo”.	2. El sistema muestra una interfaz con un campo código y un botón “Buscar”, además muestra el listado de todos los archivos.
3. El usuario introduce el código del archivo	4. El sistema verifica el campo código y

a eliminar y da click en el botón “Buscar”	muestra los datos del archivo.
5. El usuario da click en el botón “Eliminar” para eliminar el archivo y sus datos.	6. El sistema muestra un mensaje de confirmación con las opciones “Si” y “No” para ratificar si realmente se quiere eliminar el archivo y sus datos.
7. El usuario da click en el botón “Si” para eliminar los datos del archivo, si no desea eliminar los datos del archivo solo debe dar click en el botón “No”.	8. El sistema dependiendo de la opción selecciona por el usuario elimina o no el archivo y sus datos y termina el caso de uso.
Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	4.1. Si el código del archivo no existe muestra un mensaje de error. Pasa directamente a la acción 3 de la sección 3.
Poscondiciones	El sistema se actualiza luego de realizar cualquiera de las acciones anteriores.

3.5.2.1.3. Caso de Uso Buscar Archivos.

Caso de Uso:	**Buscar Archivos**
Actores:	**Usuario**
Resumen:	El CU se inicia cuando el usuario desea obtener información de los archivos, finalizando el caso de uso mostrando los datos.
Precondiciones	El usuario tiene que estar autenticado en el sistema y los datos del archivo adicionados.
Referencias	**RF 3, 4, 4.1, 4.2, 4.3, 5.**

Prioridad	**Crítico.**
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario accede a la opción “Buscar”.	2. El sistema muestra una interfaz con el listado de archivos rasters y vectorial, además ofrece los botones “Buscar”, “Búsqueda”, “Clasificar” y “Visualizar”.
3. El usuario escoge el criterio de búsqueda código, introduce el código y da click en el botón “Buscar” o para realizar una búsqueda más amplia escoge los criterios de búsqueda avanzada: nombre, extensión, proveedor, fecha de adición o tipo de archivo), introduce los datos y da click en el botón “Búsqueda”.	4. El sistema verifica los campos escogidos y muestra el listado de los archivos que cumplen con los criterios de búsqueda escogidos.
5. El usuario busca el archivo que desea visualizar, introduce su código y da click en el botón “Visualizar” (Remitirse al caso de uso extendido Visualizar Información.) o busca el archivo que desea clasificar, introduce su código y da click en el botón “Clasificar”. (Remitirse al caso de uso extendido Clasificar Archivos.)	6. El sistema abre el archivo o lo clasifica y termina el caso de uso.
Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	4.1. El sistema verifica que se haya escogido al menos un criterio de búsqueda, de lo contrario el sistema muestra un mensaje de error Pasa directamente a la acción 3.

Poscondiciones	El sistema se actualiza luego de realizar cualquiera de las acciones anteriores.

3.5.2.1.4. Caso de Uso Clasificar Archivos.

Caso de Uso:	**Clasificar Archivos**
Actores:	**Usuario**
Resumen:	El CU se inicia cuando el usuario desea clasificar los archivos, clasificarlos por archivos rasters o vectoriales para lograr obtener una mayor organización de los mismos. Luego de clasificar los archivos termina el caso de uso.
Referencias	**RF 4, 5.**
Precondiciones	Se tiene registrado la clasificación de los archivos por tipo archivo.
Prioridad	**Opcional.**

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario introduce el código del archivo a clasificar y da click en el botón “Buscar”.	2. El sistema verifica el dato introducido y muestra los datos del/de los archivo/s.
3. El usuario selecciona el criterio de clasificación deseado (rasters o vectorial), introduce los nuevos datos (dimensiones o forma de representación) respectivamente y da click en el botón “Clasificar”.	4. El sistema verifica que se haya seleccionado un criterio de clasificación, comprueba los nuevos datos, verifica que el archivo no ha sido ya catalogado y que cumple con el tipo de clasificación que se requiere(es decir que su tipo (rasters o vectorial) se corresponda con el criterio de clasificación deseado) y muestra un mensaje de confirmación con las opciones “Si” y “No” para ratificar si realmente se quiere clasificar el archivo.

5. El usuario da click en el botón “Si” para clasificar el archivo, si no desea clasificar el archivo solo debe dar click en el botón “No”.	6. El sistema dependiendo de la opción seleccionada por el usuario clasifica o no el archivo y termina el caso de uso.
Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	2.1. Si los datos introducidos son incorrectos el sistema muestra un mensaje de error. Pasa directamente a la acción 1. 4.1. Si no se seleccionó el criterio de clasificación deseado, si el archivo ya ha sido catalogado o no cumple con el tipo de clasificación deseado y los nuevos datos introducidos son incorrectos, el sistema muestra un mensaje de error. Pasa directamente a la acción 3.
Poscondiciones	El sistema se actualiza luego de realizar la acción anterior.

3.5.2.1.5. Caso de Uso Visualizar Información.

Caso de Uso:	**Visualizar Información**
Actores:	**Usuario**
Resumen:	El CU se inicia cuando el usuario desea obtener una visualización de la información contenida en los diferentes archivos tanto rasters como vectorial, finalizando el caso de uso visualizando el archivo.

Referencias	**RF 3, 4.**
Precondiciones	Se tienen guardados los datos de los archivos y almacenados los archivos.
Prioridad	**Crítico.**

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario introduce el código del archivo que desea visualizar y da click en el botón “Buscar”.	2. El sistema verifica el dato introducido y muestra una interfaz con la información del archivo seleccionado por el usuario.
3. El usuario da click en la opción “Visualizar”	4. El sistema verifica que el archivo existe, lo abre y termina el caso de uso.

Flujo Alterno	
Acción del Actor	Respuesta del Sistema
	2.1. Si el archivo no existe, el sistema muestra un mensaje de error. Pasa directamente a la acción 1.

3.6. Conclusiones.

En el presente capítulo se abordaron los aspectos más importantes de la propuesta de solución del sistema. Al concluir el mismo se tiene un conocimiento de lo que hará la aplicación sin entrar en los detalles de cómo lo hará. Se realizó un modelo de dominio para una mejor comprensión del entorno donde se presenta el problema. A partir del análisis de este modelo donde se exponen los principales conceptos, entidades y sus relaciones, quedaron definidas las funcionalidades que el sistema debe tener y cumplir para satisfacer el objetivo planteado, expresadas en los requisitos funcionales y no funcionales. Sobre la base de estos requisitos funcionales definidos fueron identificados y descritos los casos de uso y el actor del sistema, siendo de vital importancia para centrar los esfuerzos en la implementación satisfactoria de estas funcionalidades.

Capítulo 4. Construcción de la solución propuesta.

4.1. Introducción.

En este capítulo se describe el desarrollo del Modelo de Análisis, que aporta una visión del nuevo sistema propuesto sobre los requisitos funcionales identificados. Para representar todo esto se usaron los diferentes artefactos del UML, diagramas de clases y diagramas de colaboración del análisis, modelados con Visual Paradigm aplicando el Proceso Unificado (RUP), además se modela el diseño e implementación del sistema propuesto, con todos sus artefactos y diagramas.

4.2. Patrones GRASP.

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. En el diseño del sistema se pusieron en práctica el uso de algunos patrones de software para la asignación general de responsabilidades. (24)

A continuación se detallan algunos de los patrones GRASP que se pusieron en práctica en el diseño de la aplicación.

4.2.1. Patrón Controlador.

El patrón Controlador asigna la responsabilidad del manejo de mensajes de los eventos de un sistema a una clase. Un ejemplo de esto se ve reflejado en la clase, Gestionar, la cual brinda mensajes informativos al usuario.

4.2.2. Patrón Creador.

El patrón Creador se plantea como problema ¿Quién debería ser responsable de crear una nueva instancia de alguna clase? Y para ello la solución es asignarle a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:

1. B agrega los objetos A.

2. B contiene los objetos A.

3. B registra las instancias de los objetos A.

4. B utiliza específicamente los objetos A.

Un ejemplo claro de este patrón se muestra en la clase Adicionar, esta clase agrega un nuevo archivo.

4.2.3. Patrón Experto.

Este patrón es el principio básico de asignación de responsabilidades. El mismo indica que la responsabilidad de la creación de un objeto debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Se plantea como problema ¿Cuál es el principio general para asignar responsabilidades a los objetos?

Solución: Asignar una responsabilidad al experto en información.

Es importante la utilización de este patrón pues mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas. Se distribuye el comportamiento entre las clases que contienen la información requerida. Son más fáciles de entender y mantener. Un ejemplo se observa claramente en la clase Archivo, donde esta clase es la que presenta toda la información necesaria para el trabajo con los archivos, por tanto esta es la clase experta en este tema y es donde se realiza todo lo referente a archivos.

4.3. Patrón Arquitectónico.

El patrón que se aplicará es el Modelo Vista Controlador (MVC), que es actualmente el más usado en la confección de aplicaciones debido a las ventajas que trae consigo como la forma en que organiza los elementos de la aplicación(ver figura10). El MVC es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón de llamada y retorno MVC que sería la interfaz de usuario representado por los formularios y vistas, el modelo que es el sistema de gestión de base de datos el cual contiene los datos persistentes y la lógica de negocio, y el controlador que es el responsable de recibir los eventos de entrada desde la vista. Por todo lo anterior planteado el MVC es el patrón arquitectónico que más se adecua al sistema a desarrollar porque facilita la evolución por separado de la lógica del negocio y la interfaz de usuario incrementando así la reutilización y la flexibilidad del sistema.

4.3.1. Descripción del patrón.

- **Modelo**: Esta es la representación específica de la información con la cual el sistema opera. En resumen, el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo

uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

- **Vista**: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador**: Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista.

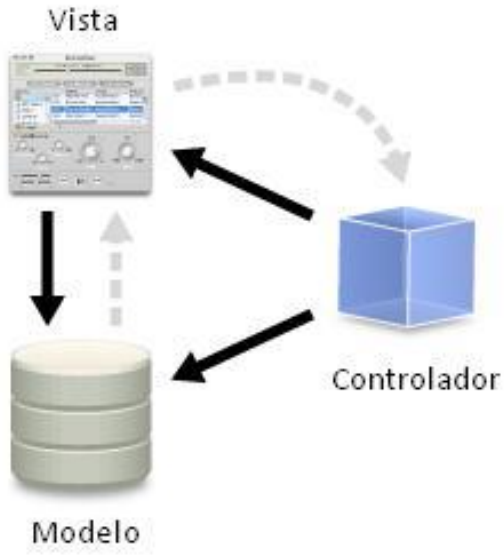


Figura10. Un diagrama sencillo que muestra la relación entre el modelo, la vista y el controlador. Nota: las líneas sólidas indican una asociación directa, y las punteadas una indirecta. (25)

4.4. Análisis.

4.4.1. Modelo de Análisis.

El análisis es el flujo de trabajo donde se refinan y estructuran los requisitos obtenidos con anterioridad con el objetivo de facilitar la compresión, preparación, modificación y manteniendo de los mismos. Un modelo de análisis es aquel que estructura los requisitos de un modo que facilita su comprensión y puede considerarse además como una primera aproximación al Modelo del Diseño.

El principal artefacto obtenido en este flujo es la realización de los Casos de Uso-Análisis, que no es más que una colaboración dentro del modelo de análisis que describe como se lleva a cabo y se ejecuta un caso de uso determinado en términos de las clases del análisis y de sus objetos del análisis en interacción. Una realización de casos de uso proporciona por tanto una traza directa hacia un caso de uso concreto del modelo de casos de uso.

4.4.1.1. Diagramas de Clases de Análisis.

Los diagramas de clases de análisis se realizan para cada uno de los casos de uso del sistema y muestran las clases participantes que se clasifican en tres tipos: interfaz, controladoras y entidades.

Clase Interfaz: se utilizan para modelar la interacción entre el sistema y sus actores, cada interfaz debe asociarse con al menos un actor y viceversa.

Clase Control: representan coordinación, secuencia, transacciones y control de otros objetos.

Clase Entidad: se utilizan para modelar la información que posee una larga vida y que es a menudo persistente.

4.5. Diseño.

En el diseño se determina la arquitectura general del sistema y su comportamiento dinámico, adaptando la especificación realizada en la etapa anterior. En esta fase se establece el comportamiento dinámico del sistema, es decir, como debe reaccionar ante los acontecimientos.

El resultado obtenido de la etapa de Diseño facilita enormemente la implementación posterior del sistema, pues proporciona la estructura básica del sistema y como los diferentes componentes actúan y se relacionan entre ellos.

4.5.1. Diagrama de Interacción.

Los diagramas de interacción muestran como se comunican los objetos en una interacción (los objetos interactúan para realizar colectivamente los servicios brindados por la aplicación).Existen dos tipos de diagramas:

- **Diagramas de Colaboración:** Resaltan la organización de los objetos que participan en una interacción
- **Diagramas de Secuencia:** Resaltan la ordenación temporal de los mensajes.

4.5.1.1. Diagrama de Colaboración.

En el desarrollo del actual software se utilizaron los diagramas de colaboración por su excepcional expresividad, su capacidad de comunicar más información contextual y su economía de espacio.

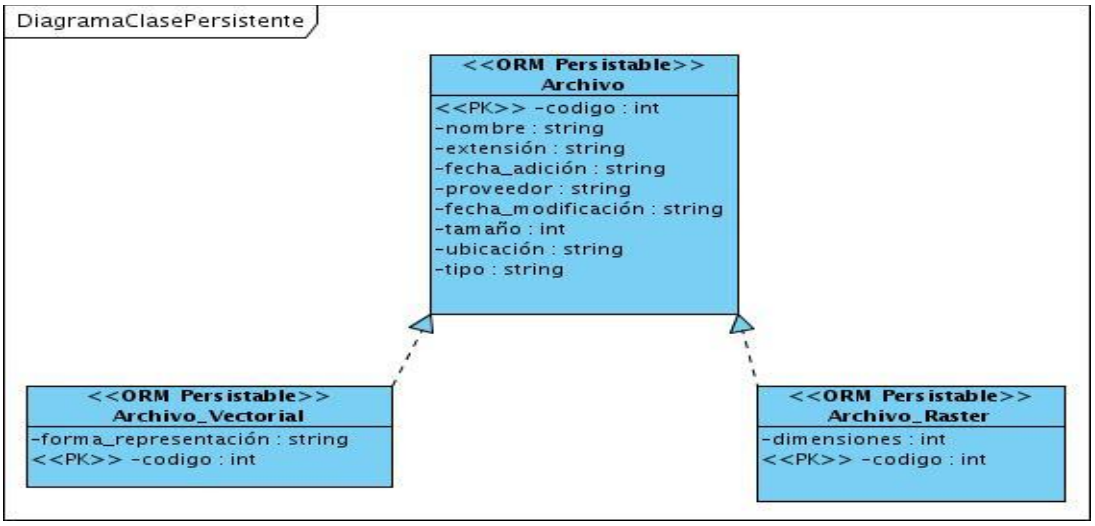
4.5.2. Diagrama de Clases del diseño.

En el diagrama de clases del diseño se describe la estructura del sistema mostrando sus clases, atributos y las relaciones entre ellas. Este diagrama es muy utilizado durante el diseño de los sistemas, y a partir de él se crea el diseño conceptual de la información que se manejará en la aplicación, y los componentes que se encargarán del funcionamiento y la relación entre uno y otro. Es importante destacar que en el sistema a desarrollar al igual que en la mayoría de los software, el diagrama de clases del diseño se actualiza, pues aparecen nuevos elementos; a medida que avanza el desarrollo de la aplicación hasta obtener la versión final del mismo que se corresponde con el diseño final del software.

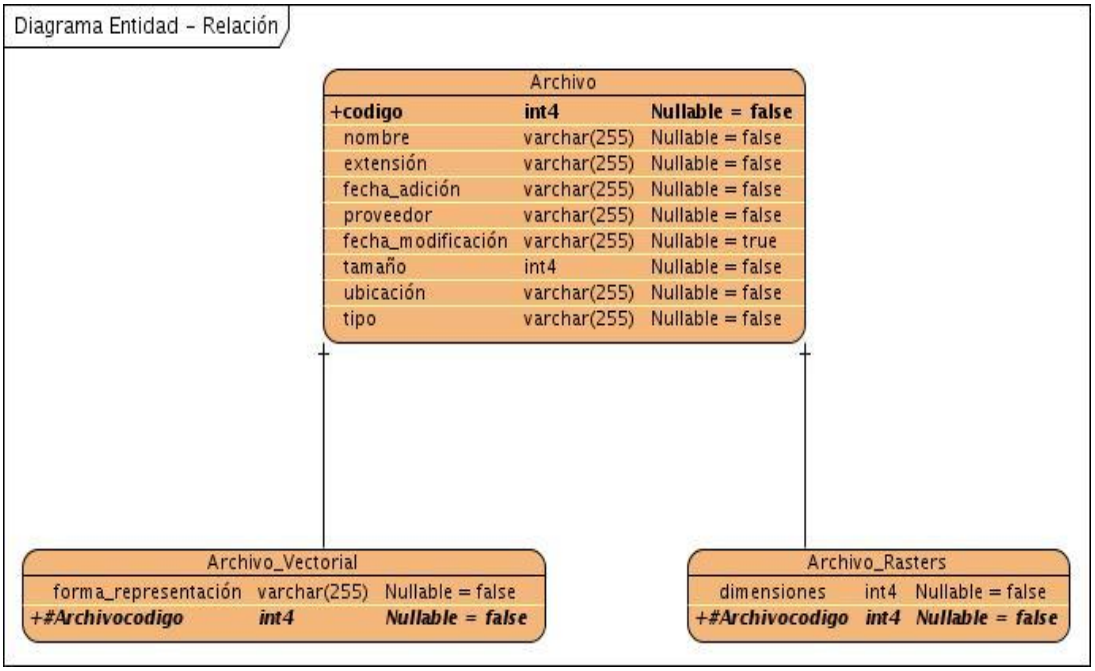
4.5.3. Diseño de la Base de Datos.

Uno de los pasos cruciales en la construcción de una aplicación que maneje una base de datos, es sin duda el diseño de la base de datos. En la base de datos que se utiliza en la aplicación que se está desarrollando como propuesta de solución, para lograr el acceso eficiente a la información con redundancia mínima, se toman varias consideraciones, entre las que se encuentran: la velocidad y facilidad de acceso a la información para extraerla. A continuación se muestra el diagrama de clases persistentes y el modelo de datos de las tablas de la base de datos.

4.5.3.1. Diagrama de Clases Persistentes.



4.5.3.2. Modelo de Datos.



4.5.3.3. Descripción de las tablas de la Base de Datos.

En este epígrafe se hace una descripción de las principales tablas, en las que se almacena toda la información relacionada con los archivos y el usuario.

Nombre: tb_Archivo		
Descripción: Contiene los datos de los archivos		
Atributo	**Tipo**	**Descripción**
código	entero	Representa el código del archivo, tiene que ser único.
nombre	varchar	Representa el nombre del archivo.
extensión	varchar	Representa la extensión del archivo.
fecha_ adición	varchar	Representa la fecha cuando fue adicionado el archivo.
proveedor	varchar	Representa el nombre del dispositivo de donde provino el archivo.
fecha_modificación	varchar	Representa la fecha en la cual fue modificado el archivo.

tamaño	entero	Representa el tamaño del archivo.
ubicación	varchar	Representa la ubicación física donde está almacenado el archivo.
tipo	varchar	Representa el tipo de archivo, rasters o vectorial.

Nombre: tb_Archivo Rasters		
Descripción: Contiene los datos de los archivos rasters.		
Atributo	**Tipo**	**Descripción**
dimensiones	entero	Representa las dimensiones del archivo.

Nombre: tb_Archivo Vectorial		
Descripción: Contiene los datos de los archivos vectorial.		
Atributo	**Tipo**	**Descripción**
forma_representación	varchar	Representa la forma de representación del archivo.

4.5.4. Principios de diseño.

El desarrollo de la aplicación se basa en los 7 principios del diseño universal. Estos principios se centran en el diseño de aplicaciones teniendo en cuenta el usuario básico y el usuario avanzado.

- El equilibrio en la organización de la información, por ejemplo, que todas las interfaces que muestran información siempre lo hagan en el mismo orden.
- La optimización de la cantidad de elementos en la pantalla, ayudando al fácil manejo y mejor comprensión de la información mostrada en la misma.
- La unidad, donde, cada elemento de la pantalla se diseñarán siguiendo un patrón de tamaño, colores y formas.

- Lograr una autonomía. La aplicación y el entorno pertenecen al usuario, pero esto no significa que se abandonen las reglas.
- Mantener un nivel alto de consistencia a través de toda la aplicación, reutilizando los criterios con que se diseñaron los íconos, diálogos, formularios, mensajes informativos.
- Los elementos que se repitan en las distintas interfaces, se situarán en un mismo lugar para mejor manejo de la información. También se trabajará sobre la base de que las interfaces no se encuentren muy cargadas, solo la información necesaria para mayor claridad.

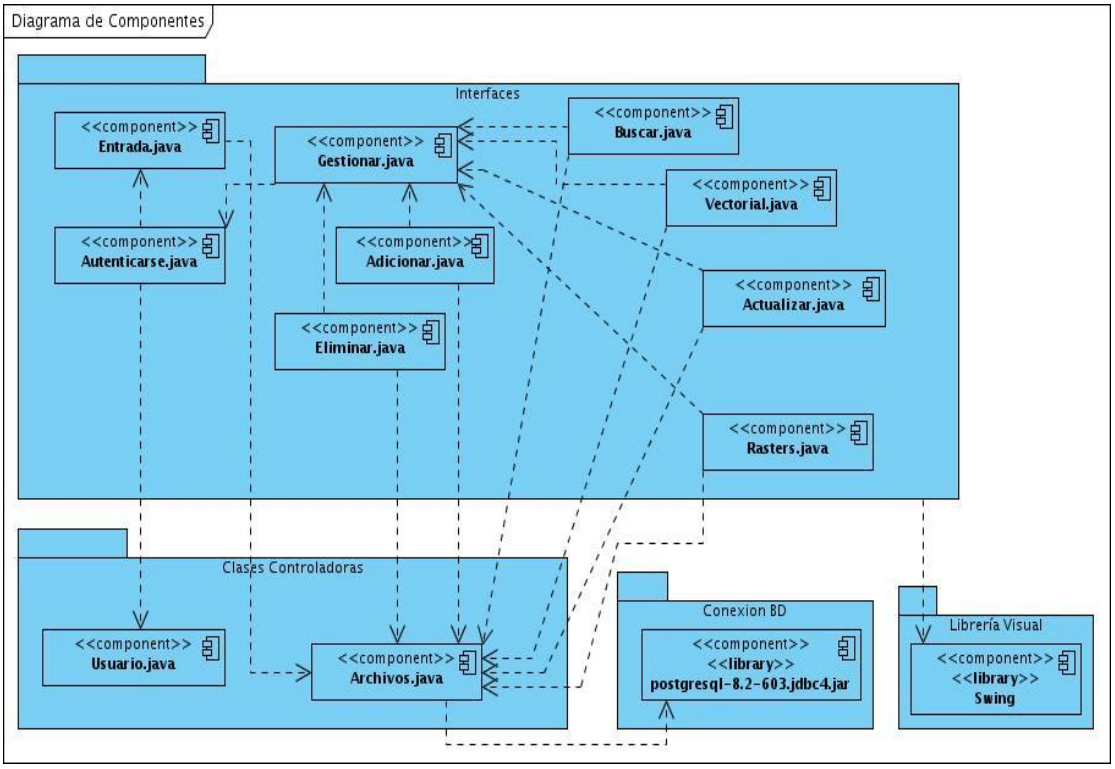
4.6. Implementación.

4.6.1. Modelo de Implementación.

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes se pueden encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos, es decir, toma el resultado del modelo del diseño para generar el código final. (26)

4.6.1.1. Diagrama de Componentes.

El diagrama de componente ilustra los componentes del software que serán usados para contribuir el sistema. Estos pueden ser construidos para el modelo de clase y escritos para satisfacer los requisitos del nuevo sistema (27). Un componente puede ser siempre considerado como una unidad autónoma dentro de un sistema o subsistema. UML define cinco estereotipos estándar que se aplican a los componentes: executable, library, Table, File y Document. Los distintos componentes pueden agruparse en paquetes según un criterio lógico y con vistas a simplificar la implementación, son paquetes estereotipados en subsistemas. Estos organizan la vista de realización de un sistema y pueden contener componentes y otros subsistemas. A continuación se muestra el diagrama de componentes del sistema.

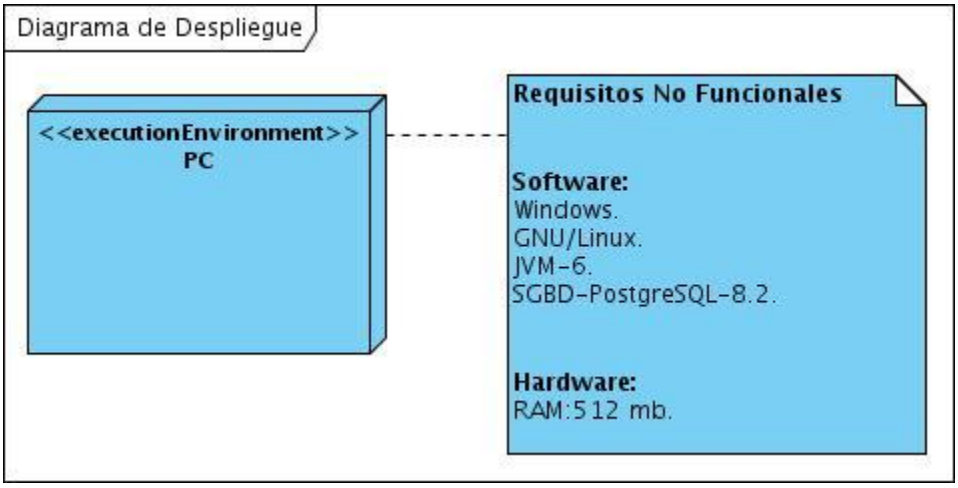


4.6.1.2. Diagrama de Despliegue.

El modelo de despliegue describe como una aplicación se despliega a través de una infraestructura. La intención de este modelo no es para describir la infraestructura, es el camino en el cual los componentes específicos deben corresponder a una aplicación que despliega a través de él. (28)

El modelo de despliegue muestra la configuración (relaciones físicas) de los nodos que participan en la ejecución y de los componentes hardware y software que residen en ellos. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que generalmente tiene algo de memoria y, a menudo, capacidad de almacenamiento. Son los elementos donde se ejecutan los componentes.

El diagrama de distribución como también se conoce, contiene: Nodos (Servidores o Procesadores y Dispositivos) y Relaciones de dependencia y asociación. A continuación se muestra el diagrama de despliegue del sistema.



Nodo PC_Servidor:

Este es el nodo que representa la computadora donde se encuentra la aplicación.

4.7. Pruebas del sistema propuesto

El modelo de pruebas es una colección de casos de prueba, procedimientos de prueba y componentes de prueba, que permite probar los componentes ejecutables en el modelo de implementación. Entre los casos de prueba se puede distinguir dos tipos comúnmente utilizados: las llamadas pruebas de “caja negra” y las de “caja blanca”.

Una prueba de “caja negra” es una prueba del comportamiento observable externamente del sistema, mientras que una prueba de “caja blanca” prueba la interacción interna entre los componentes del sistema.

En el caso de dicho sistema, se realizó pruebas de “caja negra” para cada caso de uso, con el objetivo de probar la interacción entre el usuario y el sistema, que se satisfagan las precondiciones y poscondiciones, y que se siga la secuencia de acciones intermedias especificadas por el caso de uso. La técnica utilizada en este tipo de prueba fue la de partición de equivalencia; en esta técnica, el dominio de datos de entrada es particionado en una cantidad finita de clases de equivalencias (válidas e inválidas). Cualquier dato miembro de una clase de equivalencia será un elemento representativo de dicha clase y permitirá obtener información (revelar defectos) acerca del comportamiento del software con los datos de esa clase de equivalencia. Realizar pruebas para todas y cada una de estas clases de equivalencia permite superar la necesidad de probar exhaustivamente todos los posibles datos de entrada, lo cual generalmente es imposible. (29)

4.7.1. Diseño de Casos de Pruebas.

El principal objetivo de las pruebas es la intención de descubrir errores y por su puesto una prueba tiene éxito cuando se descubre un error no detectado hasta entonces, pero en gran parte las pruebas dependen del diseño de casos de pruebas.

Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar la mayor cantidad de errores con el mínimo esfuerzo posible, en menor tiempo y su principal objetivo es cubrir todas las posibilidades sin realizar demasiados casos de pruebas.

4.8. Conclusiones.

En el presente capítulo se llevó a cabo la modelación de los diferentes procesos que presenta la aplicación, dentro de los cuales se encuentra el proceso de Análisis y Diseño, teniendo en cuenta su importancia para la realización del modelo de Implementación. Se hizo referencia a los principales artefactos generados en estos flujos de trabajo, mediante los cuales se explicó de forma más detallada las responsabilidades de cada clase, lográndose de esta forma una mayor visión y mejor comprensión de lo que se quiere obtener en el sistema. También se realizó el diseño de la Base de Datos para garantizar un funcionamiento adecuado de los datos del sistema obteniéndose el diagrama de clases persistentes y el modelo de datos, además se realizó una breve explicación de las pruebas realizadas obteniéndose como resultado 0 no conformidades.

Capítulo 5. Estudio de Factibilidad.

La necesidad que representa saber si el desarrollo de un software resultará factible se manifiesta desde los comienzos de la concepción del mismo, por ello es que se hace muy importante realizar un estudio detallado de los principales beneficios y costos que traerá consigo su ciclo de desarrollo. En este capítulo se realizará un estudio detallado de la factibilidad que representa el sistema modelado hasta el momento.

5.1. Método de Estimación por Puntos de Casos de Uso.

La estimación mediante el análisis de Puntos de Casos de Uso es un método propuesto originalmente por Gustav Karner, y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores. A continuación se despliega detalladamente dicho método.

Para realizar dicha estimación se efectúan 4 pasos fundamentales:

- Calcular los puntos de caso de uso sin ajustar.
- Calcular los puntos de casos de uso ajustados.
- Calcular la estimación del esfuerzo a través de los puntos de caso de uso.
- Calcular costo del proyecto.

Paso 1 Cálculo de los puntos de caso de uso sin ajustar (UUCP):

Para el cálculo de los puntos de caso de uso sin ajustar se hace necesario calcular también el factor de peso de los actores sin ajustar (**UAW**) y el factor de peso de los casos de uso sin ajustar. El (**UAW**), se calcula mediante un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos.

La complejidad de los actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema.

Tabla 4: Peso de los actores.

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación.	1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto	2
Complejo	Una persona que interactúa con el sistema a través de una interfaz de usuario.	3

En el sistema se tiene la siguiente información:

Tabla 5: Peso de los actores del sistema.

Nombre del actor	Complejidad
Usuario	Persona que interactúa con el sistema a través de una interfaz de usuario. Complejidad 3.

UAW = cantidad de actores * peso

UAW = 1*3= 3

El factor de peso de los casos de uso sin ajustar (**UUWC**) se calcula mediante un análisis de la cantidad de casos de uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo.

1. Si el tipo de caso de uso es simple contiene de 1 a 3 transacciones luego su factor de peso es 5.
2. Si el tipo de caso de uso es medio contiene de 4 a 7 transacciones luego su factor de peso es 10.
3. Si el tipo de caso de uso es complejo contiene más de 8 transacciones luego su factor de peso es 15.

Tabla 6: Peso de los casos de uso del sistema.

Casos de uso	Peso
Autenticar Usuario	5
Gestionar datos del archivo	15
Buscar Archivos	5
Clasificar Archivos	5
Visualizar Información	5

UUCW = Σ cantidad de caso de uso * peso
UUCW = 1*15 + 4*5= 35

Después de obtenidos los valores de **UAW** y el **UUCW** se puede calcular el valor de los puntos de caso de uso sin ajustar.

UUCP = UAW + UUCW
UUCP = 3 + 35
UUCP = 38

Paso 2 Cálculo de los puntos de casos de uso ajustados:

Para el cálculo de los puntos de caso de uso ajustados se hace necesario calcular también el factor de complejidad técnica y el factor de ambiente.

El factor de complejidad técnica (**TCF**): se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5. Se cumple que para el valor:

- 0-No presenta influencia
- 1-Influencia incidental
- 2-Influencia moderada
- 3-Influencia media
- 4-Influencia significativa
- 5-Influencia fuerte

Tabla 7: Factor de Complejidad Técnica.

Factor	Descripción	Peso	Valor Asignado	Observación	Total
T1	Sistema Distribuido	2	5	Sistema Distribuido	10
T2	Objetivos de performance o tiempo de respuesta	1	4	El tiempo de respuesta debe ser lo más rápido posible.	4
T3	Eficiencia del usuario final.	1	1	Escasas restricciones de eficiencia.	1
T4	Procesamiento interno complejo.	1	3	Existen temas de mediana complejidad en el sistema	3
T5	El código debe ser reutilizable	1	4	El código si tiene como objetivo reutilizarse.	4
T6	Facilidad de instalación.	0.5	2	Hay que tener en cuenta algunos factores para la instalación.	1
T7	Facilidad de Uso	0.5	3	Normal	1.5
T8	Portabilidad	2	1	No es un objetivo primordial que el sistema sea portable.	2
T9	Facilidad de cambio.	1	1	El sistema es un poco resistente al cambio.	1
T10	Concurrencia	1	4	El sistema presenta una alta concurrencia.	4
T11	Incluye objetivos especiales de seguridad	1	3	Seguridad Normal	3
T12	Provee acceso directo a terceras partes	1	0	No presenta acceso a terceros.	0
T13	Se requieren facilidades especiales de entrenamiento a los usuarios	1	2	Se necesita una breve explicación a los usuarios.	2

Finalmente el factor de complejidad técnica es:

TCF = 0.6 + 0.01 * Σ (peso * valor asignado)

TCF = 0.6 + 0.01 * 36.5

TCF = 0.965

El factor ambiente (**EF**): se calcula mediante la cuantificación de un conjunto de factores como son las habilidades y el entrenamiento que debe tener el grupo involucrado en el desarrollo del proyecto.

Tabla 8: Factor Ambiente.

Factor	Descripción	Peso	Valor Asignado	Comentario	Total
E1	Familiaridad con el modelo de proyecto utilizado	1.5	2	El equipo no tiene mucha experiencia con el modelo utilizado.	3
E2	Experiencia en la aplicación	0.5	2	El equipo hace poco tiempo trabaja en dicha aplicación.	1
E3	Experiencia en orientación a objetos	1	4	El equipo trabaja hace bastante tiempo orientado a objeto.	4
E4	Capacidad del analista líder	0.5	4	Existe un analista líder con experiencia en el equipo.	2
E5	Motivación	1	5	La motivación es muy alta.	5
E6	Estabilidad de los requerimientos	2	2	Se esperan cambios en los requerimientos.	6
E7	Personal part-time	-1	0	Todo el equipo es a tiempo completo.	0
E8	Dificultad del lenguaje de programación	-1	3	El lenguaje a utilizar es Java	-3
Total					**18**

EF = 1.4 – 0.03 * Σ (peso * valor asignado)

EF = 1.4 – 0.03*18

EF = 0.86

Después de obtener los valores de **UUCP, TCF y EF** se calcula el valor de los puntos de casos de uso ajustados (**UCP**):

UCP = UUCP * TCF * EF
UCP = 38 * 0.965 * 0.86
UCP = 31.53

Paso 3 Cálculo de la estimación del esfuerzo a través de los puntos de caso de uso:

El esfuerzo en horas-hombre viene dado por:
E = UCP x CF
Donde:
E: Esfuerzo estimado en horas-hombre
UCP: Puntos de Casos de Uso ajustados
CF: Factor de conversión

Este método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad (programación) especificada en los casos de uso. Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6. Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8.

CF = 20 horas-hombre (si Total EF ≤ 2)
CF = 28 horas-hombre (si Total EF= 3 ó Total EF = 4)
CF = abandonar o cambiar proyecto (si Total EF ≥ 5)
Total EF = Cant EF < 3 (entre E1 –E6) + Cant EF > 3 (entre E7, E8)
Total EF = 3+0 = 3
CF = 28 horas-hombre.
UCP =31.53
E = UCP x CF= 31.53* 28 horas-hombre =882.8 horas-hombre~883 horas-hombre.

En la siguiente tabla se muestra la distribución del esfuerzo horas-hombres en las diferentes etapas del proyecto.

Tabla 9: Distribución del esfuerzo horas-hombre por etapas.

Actividad	Esfuerzo (%)	Valor Esfuerzo
Análisis	15%	331horas-hombre
Diseño	20%	441horas-hombre
Implementación	40%	883 horas-hombre
Prueba	10%	220horas-hombre
Sobrecarga	15%	331horas-hombre
Total	100%	2207 horas-hombre

ET = 2207 horas-hombre y se estima que cada mes tiene como promedio 192 horas laborables eso daría un **ET**= 11.5 mes-hombre.
Se estima que una persona podrá realizar el problema analizado en 12 meses aproximadamente.

Paso 4 Cálculo del costo del proyecto:

Se asume como salario promedio mensual $100.00
CH: Cantidad de hombres.
T: Tiempo total del proyecto.
CHM: Costo de hombre por mes

CH = 1 hombres
CHM = 1 * Salario Promedio = 100.00 $/mes
Costo = CHM * ET / CH
Costo = 100.00 * 11.5 / 1 = $ 1150.00
Tiempo = **ET / CH**
Tiempo =11.5 / 1 = 11.5 meses.

De los resultados obtenidos se estima que con un hombre trabajando en el proyecto, el mismo se desarrolla en poco menos de 12 meses, con un costo total estimado en $1150 pesos.

5.2. Beneficios Tangibles e intangibles.

5.2.1. Beneficios Tangibles.

La solución final del software no constituye un producto comercial, por tanto, no aporta directamente ganancias económicas al país. El principal beneficio que brinda es la posibilidad de minimizar esfuerzos del personal del proyecto Control de Flotas.

5.2.2 Beneficios intangibles.

La aplicación posibilitará que el grupo de trabajo del proyecto Control de Flotas tenga el control de todos los archivos provenientes del GPS que está en funcionamiento en el proyecto, así podrá gestionar de forma ágil y segura todos los datos de los archivos.

5.2.3. Análisis de Costos y Beneficios.

El desarrollo del sistema no requerirá gastos significativos ya que todas las herramientas que se utilizaron en su confección son libres y de código abierto. Por todos los beneficios analizados anteriormente y debido a que sus costos son mínimos se puede concluir que es factible desarrollar la aplicación.

5.3. Conclusiones.

Con el estudio detallado de la factibilidad que se realizó se llegó a la conclusión que el sistema modelado será factible en muchos sentidos, analizándose los beneficios tangibles e intangibles que se desprenden del desarrollo y la utilización de la aplicación, tanto así como los costos que se tendrán en su desarrollo. Finalmente el análisis de costos contra los beneficios realizado es otra de las razones que condujo a la idea final de este estudio.

Conclusiones Generales

- ✓ La aplicación informática desarrollada permite gestionar y catalogar los archivos contenedores de datos provenientes de dispositivos GPS.

- ✓ El modelo de software diseñado sobre la base de las tecnologías y herramientas actuales garantizan la calidad y el funcionamiento del sistema, así como su mantenimiento y perduración, satisfaciendo de esta forma las necesidades del proyecto Control de Flotas.

- ✓ La factibilidad del proyecto realizado ha sido demostrada a través de la estimación por puntos de casos de usos, donde los resultados tangibles e intangibles obtenidos son mayores a los costos incurridos.

- ✓ El sistema garantiza un mayor control de los archivos contenedores de datos provenientes de dispositivos GPS ayudando en gran medida al personal del equipo de trabajo del proyecto Control de Flotas.

Recomendaciones

- ✓ Documentar las principales deficiencias que se generen durante el despliegue del sistema y utilizarlas como premisas para el desarrollo de próximas iteraciones.

- ✓ En estudios posteriores relacionados con el tema, se recomienda ampliar la arquitectura propuesta para desarrollar una aplicación lo más reutilizable posible y la realización de un conjunto de pruebas de caja blanca con el objetivo de obtener un producto de mayor calidad.

Bibliografía Referenciada

1. **Manuel Salvador Luzanía Valerio.** Los Sistemas de Información Geográfica. *www.uv.mx.* [En línea] Septiembre de 2005. [Citado el: 5 de Diciembre de 2010.] http://www.uv.mx/cienciahombre/revistae/vol18num3/articulos/informacion%20geografica/index.htm.

2. **Herring.** The Global Positioning System(GPS). *www.content.wuala.com.* [En línea] 1996, pag 32-38. [Citado el: 10 de Diciembre de 2010.] http://content.wuala.com/contents/BIDIMEACD/Documents/Documentos%20Geograf%C3%ADa%20Aplicada/Sistema%20GPS.pdf.

3. **Luis Felipe Hernández.** GPS en Cuba. *www.granma.cubaweb.cu.* [En línea] 09 de Marzo de 2010. [Citado el: 10 de Diciembre de 2010.] http://www.granma.cubaweb.cu/2010/03/09/nacional/artic20.html.

4. **David Rhind.** Definición de SIG. *www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1184.* [En línea] 1989. [Citado el: 15 de Febrero de 2011.] http://www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1184.

5. Definición de Modelo Vectorial. *www.MANUALESVECTOPDF042010.* [En línea] [Citado el: 15 de Febreo de 2011.] http://moodle.eingenia.es/file.php/16/MANUALESVECTOPDF042010/UNIDAD1.pdf.

6. **Escobar, Dr F.** Definición de Modelo Rasters. *www.sli.unimelb.edu.au/gisweb.* [En línea] [Citado el: 15 de febrero de 2011.] http://www.sli.unimelb.edu.au/gisweb/.

7. Definición de GPS. *www.granabike.com/consejos/gps-btt-mtb.html.* [En línea] [Citado el: 15 de febrero de 2011.] http://www.granabike.com/consejos/gps-btt-mtb.html.

8. Definición de Catálogo. *www.es.wiktionary.org/wiki/Catalogo.* [En línea] [Citado el: 15 de febrero de 2011.] http://es.wiktionary.org/wiki/Catalogo.

9. **López, Lorna del Rosario Ochoa.** Sistemas de Información Geográfica, ventajas y desventajas de su utilización. *www.biblioteca.usac.edu.gt.* [En línea] [Citado el: 10 de Febrero de 2011.] http://biblioteca.usac.edu.gt/tesis/08/08_5530.pdf.

10. Mapa Topográfico. *www.vmapas.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.vmapas.com/Africa/Argelia/Mapa_Topografico_Algeria.jpg/maps-es.html.

11. Mapas temáticos. *http://descubrejuanfernandez.blogspot.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://descubrejuanfernandez.blogspot.com/.

12. Modelo Rasters. *www.urbanismogranada.com.* [En línea] [Citado el: 10 de Febrero de 2011.] http://www.urbanismogranada.com/administrador/archivos/04_10_07_MODELO_RASTER.pdf.

13. Modelo Vectorial. *www.mapserver.inegi.gob.mx.* [En línea] [Citado el: 15 de Enero de 2011.] http://mapserver.inegi.gob.mx/geografia/espanol/normatividad/diccio/mod_vec.pdf.

14. **Joel Manrrique Chávez.** Lenguaje PHP. *www.monografias.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.monografias.com/trabajos38/programacion-php/programacion-php.shtml?monosearch.

15. **Gregorio Hernández Peñalver.** Lenguaje Java. *www.dma.fi.upm.es.* [En línea] [Citado el: 14 de Febrero de 2011.] http://www.dma.fi.upm.es/gregorio/JavaGC/Cconvexo/teoriaJava.html.

16. **Manuel Calero Solís.** Programación Extrema. *www.willydev.net/descargas/prev/explicaxp.pdf.* [En línea] [Citado el: 10 de marzo de 2011.] http://www.willydev.net/descargas/prev/explicaxp.pdf.

17. Microsoft Solution Framework . *www.e-gattaca.com.* [En línea] [Citado el: 10 de Marzo de 2011.] http://www.e-gattaca.com/econtent/library/documents/DocNewsNo50DocumentNo6.PDF.

18. Visual Paradigm Internacional. *www.freedownloadmanager.org.* [En línea] [Citado el: 20 de Febrero de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%.

19. IDE de Programación. *www.buenastareas.com.* [En línea] [Citado el: 10 de Marzo de 2011.] http://www.buenastareas.com/ensayos/Que-Es-Un-Ide-De-Programacion/163537.html.

20. MySQL . *www.danielpecos.com.* [En línea] [Citado el: 15 de Marzo de 2011.] http://danielpecos.com/docs/mysql_postgres/x57.html.

21. Modelo de Dominio. *www.es.scribd.com.* [En línea] [Citado el: 5 de Abril de 2011.] http://es.scribd.com/doc/51305810/40/Modelo-del-Dominio.

22. **Ivar Jacobson, g. b., James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software. p. 107.*

23. —. *El Proceso Unificado de Desarrollo de Software. p. 110.* : s.n.

24. Patrones Grasp. *www.practicasdesoftware.com.ar.* [En línea] [Citado el: 20 de Abril de 2011.] http://www.practicasdesoftware.com.ar/2011/03/patrones-grasp/.

25. Modelo Vista Controlador. *blogdeaitor.wordpress.com.* [En línea] [Citado el: 25 de Abril de 2011.] http://blogdeaitor.wordpress.com/2008/10/20/model-view-controller/.

26. Modelo de Implementación. *www.merinde.net.* [En línea] [Citado el: 10 de Mayo de 2011.] http://merinde.net/index.php?option=com_content&task=view&id=495&Itemid=291.

27. Diagrama de Componentes. *www.monografias.com.* [En línea] [Citado el: 15 de Mayo de 2011.] http://www.monografias.com/trabajos34/ingenieria-software/ingenieria-software.shtml#dcolabor.

28. Diagrama de Despliege. *www.monografias.com.* [En línea] [Citado el: 15 de Mayo de 2011.] http://www.monografias.com/trabajos34/ingenieria-software/ingenieria-software.shtml#dcomponn.

29. **Lay, Carlos Núñez.** Tecnicas-para-disenar-casos-de-prueba-del-Software-Carlos-Nunez. *www.es.scribd.com.* [En línea] [Citado el: 30 de Mayo de 2011.] http://es.scribd.com/doc/46938690/Tecnicas-para-disenar-casos-de-prueba-del-Software-Carlos-Nunez.

Bibliografía Consultada

1. **Manuel Salvador Luzanía Valerio.** Los Sistemas de Información Geográfica. *www.uv.mx.* [En línea] Septiembre de 2005. [Citado el: 5 de Diciembre de 2010.] http://www.uv.mx/cienciahombre/revistae/vol18num3/articulos/informacion%20geografica/index.htm.

2. **Herring.** The Global Positioning System(GPS). *www.content.wuala.com.* [En línea] 1996, pag 32-38. [Citado el: 10 de Diciembre de 2010.] http://content.wuala.com/contents/BIDIMEACD/Documents/Documentos%20Geograf%C3%ADa%20Aplicada/Sistema%20GPS.pdf.

3. **Luis Felipe Hernández.** GPS en Cuba. *www.granma.cubaweb.cu.* [En línea] 09 de Marzo de 2011. [Citado el: 10 de Diciembre de 2011.] http://www.granma.cubaweb.cu/2010/03/09/nacional/artic20.html.

4. **David Rhind.** Definición de SIG. *www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1184.* [En línea] 1989. [Citado el: 15 de Febrero de 2011.] http://www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1184.

5. Definición de Modelo Vectorial. *www.MANUALESVECTOPDF042010.* [En línea] [Citado el: 15 de Febreo de 2011.] http://moodle.eingenia.es/file.php/16/MANUALESVECTOPDF042010/UNIDAD1.pdf.

6. **Escobar, Dr F.** Definición de Modelo Rasters. *www.sli.unimelb.edu.au/gisweb.* [En línea] [Citado el: 15 de febrero de 2011.] http://www.sli.unimelb.edu.au/gisweb/.

7. Definición de GPS. *www.granabike.com/consejos/gps-btt-mtb.html.* [En línea] [Citado el: 15 de febrero de 2011.] http://www.granabike.com/consejos/gps-btt-mtb.html.

8. Definición de Catálogo. *www.es.wiktionary.org/wiki/Catalogo.* [En línea] [Citado el: 15 de febrero de 2011.] http://es.wiktionary.org/wiki/Catalogo.

9. **López, Lorna del Rosario Ochoa.** Sistemas de Información Geográfica, ventajas y desventajas de su utilización. *www.biblioteca.usac.edu.gt.* [En línea] [Citado el: 10 de Febrero de 2011.] http://biblioteca.usac.edu.gt/tesis/08/08_5530.pdf.

10. Mapa Topográfico. *www.vmapas.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.vmapas.com/Africa/Argelia/Mapa_Topografico_Algeria.jpg/maps-es.html.

11. Mapas temáticos. *http://descubrejuanfernandez.blogspot.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://descubrejuanfernandez.blogspot.com/.

12. Modelo Rasters. *www.urbanismogranada.com.* [En línea] [Citado el: 10 de Febrero de 2011.] http://www.urbanismogranada.com/administrador/archivos/04_10_07_MODELO_RASTER.pdf.

13. Modelo Vectorial. *www.mapserver.inegi.gob.mx.* [En línea] [Citado el: 15 de Enero de 2011.] http://mapserver.inegi.gob.mx/geografia/espanol/normatividad/diccio/mod_vec.pdf.

14. **Joel Manrrique Chávez.** Lenguaje PHP. *www.monografias.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.monografias.com/trabajos38/programacion-php/programacion-php.shtml?monosearch.

15. **Gregorio Hernández Peñalver.** Lenguaje Java. *www.dma.fi.upm.es.* [En línea] [Citado el: 14 de Febrero de 2011.] http://www.dma.fi.upm.es/gregorio/JavaGC/Cconvexo/teoriaJava.html.

16. **Manuel Calero Solís.** Programación Extrema. *www.willydev.net/descargas/prev/explicaxp.pdf.* [En línea] [Citado el: 10 de marzo de 2011.] http://www.willydev.net/descargas/prev/explicaxp.pdf.

17. Microsoft Solution Framework . *www.e-gattaca.com.* [En línea] [Citado el: 10 de Marzo de 2011.] http://www.e-gattaca.com/econtent/library/documents/DocNewsNo50DocumentNo6.PDF.

18. Visual Paradigm Internacional. *www.freedownloadmanager.org.* [En línea] [Citado el: 20 de Febrero de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%.

19. IDE de Programación. *www.buenastareas.com.* [En línea] [Citado el: 10 de Marzo de 2011.] http://www.buenastareas.com/ensayos/Que-Es-Un-Ide-De-Programacion/163537.html.

20. MySQL . *www.danielpecos.com.* [En línea] [Citado el: 15 de Marzo de 2011.] http://danielpecos.com/docs/mysql_postgres/x57.html.

21. Modelo de Dominio. *www.es.scribd.com.* [En línea] [Citado el: 5 de Abril de 2011.] http://es.scribd.com/doc/51305810/40/Modelo-del-Dominio.

22. **Ivar Jacobson, g. b., James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software. p. 107.*

23. —. *El Proceso Unificado de Desarrollo de Software. p. 110.* : s.n.

24. Patrones Grasp. *www.practicasdesoftware.com.ar.* [En línea] [Citado el: 20 de Abril de 2011.] http://www.practicasdesoftware.com.ar/2011/03/patrones-grasp/.

25. Modelo Vista Controlador. *blogdeaitor.wordpress.com.* [En línea] [Citado el: 25 de Abril de 2011.] http://blogdeaitor.wordpress.com/2008/10/20/model-view-controller/.

26. Modelo de Implementación. *www.merinde.net.* [En línea] [Citado el: 10 de Mayo de 2011.] http://merinde.net/index.php?option=com_content&task=view&id=495&Itemid=291.

27. Diagrama de Componentes. *www.monografias.com.* [En línea] [Citado el: 15 de Mayo de 2011.] http://www.monografias.com/trabajos34/ingenieria-software/ingenieria-software.shtml#dcolabor.

28. Diagrama de Despliege. *www.monografias.com.* [En línea] [Citado el: 15 de Mayo de 2011.] http://www.monografias.com/trabajos34/ingenieria-software/ingenieria-software.shtml#dcomponn.

29. **Lay, Carlos Núñez.** Tecnicas-para-disenar-casos-de-prueba-del-Software-Carlos-Nunez. *www.es.scribd.com.* [En línea] [Citado el: 30 de Mayo de 2011.] http://es.scribd.com/doc/46938690/Tecnicas-para-disenar-casos-de-prueba-del-Software-Carlos-Nunez.

30. Lenguajes de programación. *www.lenguajes-deprogramacion.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.lenguajes-deprogramacion.com.

31. Ventajas de PostgresSQL. *advocacy.postgresql.org.* [En línea] [Citado el: 20 de Febrero de 2011.] http://advocacy.postgresql.org/advantages/?lang=es.

32. Vector Rasters. *www.gemini.udistrital.edu.co.* [En línea] [Citado el: 15 de Febrero de 2011.] http://gemini.udistrital.edu.co/comunidad/profesores/rfranco/vector_raster.htm.

33. Sistema GPS. *html.rincondelvago.com.* [En línea] [Citado el: 5 de Diciembre de 2010.] http://html.rincondelvago.com/sistemas-gps.html.

34. Catálogos. *www.infobservador.blogspot.com.* [En línea] 7 de Diciembre de 2010. [Citado el: 25 de Febrero de 2011.] http://infobservador.blogspot.com/2010/12/veces-hay-muchas-referencias-de.html.

35. Historia de los Catálogos. *www.librolibertate.wordpress.com.* [En línea] 05 de Junio de 2007. [Citado el: 10 de Diciembre de 2010.] http://librolibertate.wordpress.com/2007/06/05/un-poco-de-historia-de-los-catalogos-y-la-catalogacion%E2%80%A6/.

36. Que es Java. *www.java.ciberaula.com.* [En línea] [Citado el: 15 de Febrero de 2011.] http://java.ciberaula.com/articulo/que_es_java.

37. Sistema Gestor de Base de datos. *www.scielo.cl.* [En línea] [Citado el: 25 de Febrero de 2011.] http://www.scielo.cl/pdf/ingeniare/v16n1/ART05.pdf.

38. Sistemas-de-informacin-geogrfica. *www.slideshare.net.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.slideshare.net/sacra07/sig-sistemas-de-informacin-geogrfica.

39. IDE NetBeans. *www.sun.com.* [En línea] [Citado el: 15 de Marzo de 2011.] http://www.sun.com/emrkt/innercircle/newsletter/latam/0207latam_feature.html.

40. Modelos y estructuras de datos. *www.um.es.* [En línea] [Citado el: 20 de Febrero de 2011.] http://www.um.es/geograf/sigmur/sigpdf/temario_3.pdf.

Glosario de Términos

API: Interfaz de Programación de Aplicaciones.

CASE: es una sigla, que corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al Español significa Ingeniería de Software Asistida por Computación.

CU: Caso de Uso: Es una descripción de un conjunto de secuencias de acciones, incluyendo variaciones, que un sistema lleva a cabo y que conduce a un resultado observable de interés para un actor determinado.

Factibilidad: El concepto de factibilidad debe entenderse como la posibilidad y la conveniencia de llevar a cabo un proyecto. Para el desarrollo de sistemas los proyectos son evaluados a través de tres tipos de estudios de factibilidad: técnica, económica y operacional.

GPS: Sistema de Posicionamiento Global.
IDE: Entorno de Desarrollo Integrado.
UCI: Universidad de las Ciencias Informáticas.