UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad #6

Trabajo de diploma para optar por el título de Ingeniero en Informática

Sistema para la reducción de grafos y búsqueda de camino mínimo

Autores	Tutores
Yasnary González Pérez	MSc. Rafael Rodríguez Puente
José Angel Inda Herrera	Ing. Yoandrys Lazo Nodarse

Ciudad de la Habana, mayo de 2011

"Año 53 de la Revolución"

Declaración de Autoría

Declaramos ser los únicos autores del trabajo que tiene como título “Sistema para la reducción de grafos y búsqueda de camino mínimo” y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Yasnary González Pérez

_____

Firma del Autor

José Ángel Inda Herrera

_____

Firma del Autor

Rafael Rodríguez Puente

_____

Firma del Tutor

Yoandrys Lazo Nodarse

_____

Firma del Tutor

"Emplearse en lo estéril cuando se puede hacer lo útil; ocuparse en lo fácil cuando se tienen bríos para intentar lo difícil, es despojar de su dignidad al talento".

José Martí

Dedicatoria

A mi abuela Rafaela que aunque no esté, se que se siente orgullosa de mi

A mis padres Idalis, Eugenioy Maria Isabel a quienes les debo la vida y por ellos es que estoy aquí hoy.

A mis hermanas María Fernanda, Jany e Isabella Sofia que sea para ellas un ejemplo a seguir en la vida.

Yasnary

A mi abuelo Jose Antonio que aunque no esté, se que está orgulloso de mi.

Jose Ángel

Agradecimientos

Al tutor Rafael Rodríguez Puente por guiarme en todo momento y estar siempre dispuesto cuando lo necesité.

Al co-tutor Yoandry Lazo por su apoyo brindado.

A mis padres Idalis, Eugenio y María Isabel por el amor sin límites, el sacrificio, las preocupaciones y la confianza, espero no defraudarlos y llegar a ser la persona que ellos quieren que sea.

A mis hermanas María Fernanda, Jany, Isabella Sofia y Beatriz por darme las fuerzas para seguir adelante y representar para ellas un ejemplo a seguir.

A mis abuelos Rafaela, Mirian y Jose por sus sabios consejos.

A mis tíos Irina, Alfredo, Gerardito y Juan por apoyarme en todo momento.

A mi tío Luis por sus grandes aciertos y buenos consejos.

A mi maravillosa familia de Matanzas, a los cuales quiero mucho.

A mi suegro el cual aprecio y admiro, por su confianza y apoyo.

A mi suegra y a mí cuñado por el cariño que me ha brindado, a mi tía Mirtica por sus consejos y ayuda incondicional. A abuela Gloria por su amor y por las largas charlas.

A abuelo Chucho por sus anécdotas, a los tíos Glori, Heriberto, Jose, Fide y a Mirdy por su apoyo.

A mis amistades que no podría enumerarlas porque son muchas. A Sol Elena, Yanelis Lisandra, Grechin, Mailen, Dayana, Danieyis, Made gracias por compartir conmigo estos 5 años de mi vida.

A mi amigo entrañable Manolo (Manuel) por estar aquí aún no estando.

En especial, a mi compañero de tesis, novio, amigo y dentro de poco mi esposo Jose Ángel, quien me ha dedicado estos años de su vida, por su amor, cuidado y comprensión, por entenderme y apoyarme en los momentos que hemos vivido y por ser lo mejor que me ha pasado en la Vida.

Yasnary

Agradecimientos

Al profesor y tutor, Rafael Rodríguez Puentes, creo de no haberle conocido no tuviera hoy entre mis manos ese torrencial de conocimientos que de él emergen cada día, creo que mi formación no hubiera sido la misma sin las reiteradas jornadas (madrugadas en ocasiones) que gentilmente me ha regalado, con él he aprendido el significado que tiene en la vida la satisfacción y el estímulo mayor –de largas horas de trabajo.

A Yoandry Lazo por brindarnos su apoyo en todo momento.

A mis padres Lucía, Jose y Mirty, a mi hermano Jorgito, a mis hermanas Mirdy y Betty, a mis abuelos Jose Antonio, Gloria, Chucho y Elia, a mis tíos Heri y Glory, a mi primo Erne, y a mi primo Joe que esto sea un ejemplo a seguir y se convierta en un gran profesionar.

A mi familia de Ciego por acogerme como uno más de ellos, a mi suegro por ser como un padre y mi tía Mary por ser como una madre. A mi suegra y cuñadita MariaFe.

En especial a mi amiga, compañera de tesis, hermana y novia Yasnary por estar a mi lado en estos 5 años en lo bueno y en lo malo, por entenderme y soportarme, amarme y respetarme por sobre todas las cosas. Creo que mi vida sería distinta de no haberla conocido.

A mis amigos Adrian, Alain, Leiber y Aurelio por compartir momentos de risas y juegos de futbol.

A mi amigo y hermano Humberto por compartir su amistad durante estos 5 años.

José Ángel

Resumen

Hoy en día existe un gran desarrollo de los Sistemas de Información Geográfica, los mismos poseen diversas funcionalidades entre las que se encuentra visualización de mapas, búsqueda de caminos mínimos, etc. El presente trabajo tiene como objetivo principal la implementación de un sistema para el cálculo de caminos mínimos que cuente con dos módulos, el primero transforma la cartografía que representa una red de viales, almacenada en una base de datos PostgreSQL, a un modelo matemático, específicamente un grafo que será almacenado en el motor de persistencia para almacenamiento de grafos Neo4j. El segundo módulo se encarga de tomar un grafo almacenado en el motor de persistencia para almacenamiento de grafos Neo4j y aplicarle el número de reducciones que se especifique, para ello se implementa un algoritmo de reducción de grafos que no presenta pérdida de información. También se propone la implementación de un algoritmo de búsqueda de caminos mínimos sobre un grafo reducido lo cual disminuye el tiempo de respuesta. El sistema resultante de este trabajo será capaz de integrarse con Sistemas de Información Geográfica mediante un servicio web en el cual el cliente de dicho servicio debe especificar los puntos de origen y destino para realizar la búsqueda de camino mínimo y el sistema devuelve como resultado los puntos que forman el camino.

Índice

Índice de Tablas

Índice de Tablas

Índice de Figuras

Introducción

Las Tecnologías de la Información y la Comunicación (TIC), a través de los años han ido evolucionando gradualmente, gracias a este desarrollo el hombre ha podido crear los Sistema de Información Geográfica (SIG).

Los SIG dan una perspectiva totalmente nueva y dinámica a la información, ayudando además a tomar mejores decisiones, entre el 80 y el 90% de la información involucrada en la toma de decisiones tiene una componente espacial **(1)**.

Entre las funcionalidades que brinda un SIG se puede mencionar: búsqueda de un fenómeno, creación de mapas temáticos **(2)**, búsqueda de caminos óptimos, cálculo de distancia, etc. Los SIG se han convertido en una industria importante y una disciplina académica.

Desde hace ya unos años Cuba estuvo inmerso en un proceso de transformaciones desde el punto de vista tecnológico, dentro de muchas tecnologías que hoy en día se estudian en nuestro país se encuentran las relacionadas con el desarrollo de los Sistemas de Información Geográfica y su vinculación y adaptación con las alternativas libres (software libre) existentes en el mundo. En la última década se han creado diversas instituciones a nivel nacional que contribuyen al crecimiento informático del país y como parte de los aportes que cada día sustentan ese desarrollo se encuentra el estudio de los SIG. La Universidad de las Ciencias Informáticas (UCI) es un ejemplo de estos centros, la misma está llamada a impulsar la industria del software del país.

La UCI cuenta con una infraestructura productiva, la cual está basada en la creación de centros de desarrollo acorde a las líneas temáticas de interés, que a la vez son distribuidas por todas las facultades de la misma. La Facultad 6 es un ejemplo de esta infraestructura, en la misma se encuentra el centro de desarrollo de Geoinformática y Señales Digitales (GEySED), en este centro se lleva a cabo el desarrollo de una plataforma para la representación y edición de la cartografía y de la información geoespacial. Uno de sus proyectos es el proyecto SIG DESKTOP, en el mismo se lleva a cabo la personalización de un sistema genérico capaz de representar información geográfica en aplicaciones de escritorio. En dicha personalización se utiliza PostgreSQL **(3)** como Sistema Gestor de Bases de Datos (SGBD). El SGBD PostgreSQL cuenta con el módulo PostGIS **(4)** para el manejo de datos espaciales, el cual puede utilizar la biblioteca pgRouting **(5)** para el análisis de redes. PgRouting

utiliza la cartografía como modelo para realizar análisis de redes, fundamentalmente para la búsqueda de caminos mínimos, este hecho trae como consecuencia limitaciones en las funcionalidades que brinda así como en la escalabilidad del sistema **(6)**, por ejemplo, los puntos de origen y destino para realizar una búsqueda de camino mínimo no pueden ser arbitrarios, tienen que estar exactamente en una intersección de calles o en un punto de interés del mapa de lo contrario se pueden obtener resultados erróneos; por otra parte pgRouting utiliza un modelo de grafo demasiado simple, poco escalable y no puede usarse en entorno multi-hilo. También representa y analiza las redes de forma plana, lo que trae como consecuencia que cuando dos calles están a diferentes niveles, el sistema no distingue si se intersectan o no, lo que puede traer problemas al realizar los diferentes análisis de redes. Estas deficiencias inciden directamente en el análisis de redes, lo cual se debe a que el modelo utilizado no es capaz de representar de forma natural las redes existentes en una cartografía.

Partiendo de la situación expuesta con anterioridad se ha definido como **problema a resolver** ¿Cómo implementar de forma eficiente la búsqueda de camino mínimo en un grafo grande?

En pos de resolver el problema expuesto, se hace necesario implementar un algoritmo que sea capaz de reducir un grafo sin que haya pérdida de información y realizar búsquedas de caminos mínimos en el grafo reducido de forma eficiente, lo cual figura como **objetivo general**.

Para darle cumplimiento al objetivo trazado fue necesario centrar el estudio de la investigación en el análisis de redes en Sistemas de Información Geográfica, lo cual constituye el **objeto de estudio**.

El **campo de acción** que enmarca dicho estudio es la búsqueda de caminos mínimos.

El desarrollo exitoso del objetivo expuesto anteriormente contribuirá al cumplimiento de la siguiente **idea a defender**: la implementación de un sistema que reduzca un grafo grande sin que haya pérdida de información permitirá el cálculo de caminos mínimos de forma eficiente en los grafos reducidos.

Para lograr los objetivos trazados se realizan las siguientes **tareas de investigación**:

- Definir la metodología ágil de desarrollo para el desarrollo del sistema.
- Implementar módulo de obtención del grafo a partir de la cartografía para aplicar la reducción del grafo.

- Documentar la implementación de éste módulo para lograr un mejor entendimiento del sistema.
- Realizar pruebas unitarias para comprobar que el sistema funcione correctamente.
- Integrar con el SGBD de grafos seleccionado. La aplicación debe ser capaz de obtener el grafo que representa una red de viales y almacenarlo en el SGBD orientado a grafo para poder aplicarle el algoritmo de reducción.
- Definir SGBD de grafos a utilizar para lograr el desarrollo del sistema.
- Implementar algoritmo de reducción de grafos para lograr realizar la búsqueda de camino mínimo de forma eficiente.
- Implementar un mecanismo de reescritura de grafos sencillo para el desarrollo del sistema.
- Definir forma de almacenamiento del grafo reducido para tener le grafo almacenado.
- Definir forma de almacenamiento de las reglas de reescritura para obtener el grafo original una vez reducido.
- Implementar algoritmo de búsqueda de camino más corto sobre el grafo reducido para obtener una respuesta eficiente.
- Implementar la función fv definida en el algoritmo de reducción de grafos, para obtener el costo del camino desde N_1 hasta N_2 pasando por un N_3 reducido.
- Documentar la implementación del algoritmo para obtener la documentación completa del sistema.

Métodos de investigación

Con el objetivo de llevar a cabo la implementación de un algoritmo que sea capaz de reducir un grafo y realizar búsquedas de caminos mínimo en el grafo reducido de forma eficiente se hizo necesario apoyarse en métodos teóricos de investigación. De los teóricos se utilizaron el Análisis Histórico-Lógico, Analítico Sintético y Modelación.

El método **Análisis Histórico-Lógico** permite determinar las principales técnicas de representación de redes viales utilizando un modelo matemático y que los mismos pueden ser aplicables en el sistema. Permite además realizar el estudio de determinados elementos necesarios para llevar a cabo la investigación, como son, realizar una búsqueda de las herramientas y algoritmos existentes en la actualidad, así como su evolución y desarrollo, obteniendo de dicho estudio las herramientas y los algoritmos más óptimos para el desarrollo del sistema.

El método **Analítico-Sintético** se define con el objetivo de analizar los diversos documentos relacionados con la representación de redes de viales así como el análisis de rutas sobre las mismas.

El método **Modelación** se emplea para la representación de un algoritmo que ayude a entender la lógica del sistema, obteniendo a partir de dicha modelación, estrategias para la representación de redes de viales utilizando un modelo matemático y un sistema para la reducción de grafos y búsqueda de caminos mínimo

Estructura del documento

La tesis está estructurada por tres capítulos, los cuales se describen a continuación:

Capítulo 1: El capítulo uno presenta la fundamentación teórica, exponiendo las definiciones fundamentales para lograr la buena comprensión del trabajo. Se hace un estudio de los sistemas previamente existentes relacionándolos con el objetivo que se persigue, algunos de estos sistemas presentan puntos en común con el trabajo de redes de viales, pero ninguno satisface plenamente nuestra tesis. Se especifican además las tecnologías a utilizar y la metodología de desarrollo.

Capítulo 2: El capítulo dos expone la descripción de la solución del sistema, detalla las Historias de Usuarios obtenidas de la metodología XP y definidas en la fase de exploración, así como el tiempo estimado para el desarrollo de cada una de las Historia de usuarios. Además se describen las Iteraciones en la que se va a implementar el sistema.

Capítulo 3: El capítulo tres presenta la Construcción, Pruebas y resultados aplicadas y obtenidas de la implementación del algoritmo de reducción de grafo. Detallando así las tarjetas C.R.C para obtener una lógica del funcionamiento del sistema. Se especifican las Tareas de Ingeniería en las que se van a desglosar las Historias de usuarios para obtener un eficiente funcionamiento. Además se detallan las pruebas unitarias y las pruebas de aceptación pertenecientes al sistema.

1

CAPÍTULO

1. Fundamentación Teórica

En este capítulo se presentan los principales conceptos que pueden resultar de relevancia para el tema abordado en la tesis y que son necesarios para una buena comprensión del trabajo. Se realizará un análisis crítico del estado actual de las tecnologías que permiten realizar análisis de redes en SIG. Además de exponer las tecnologías a utilizar para el desarrollo del sistema.

1.1. Definiciones previas

Los Sistemas de Información Geográfica han ido evolucionando enormemente desde las décadas de los 80 en adelante. Uno de los principales retos a los que se enfrentan los SIG en la actualidad es por ejemplo, cómo llegar desde un punto *x* a uno *y*, o conocer el lugar donde está en estos momentos, etc.

1.1.1. Sistemas de Información Geográfica.

Los Sistemas de Información Geográfica (SIG) son un conjunto de herramientas que están destinadas para el trabajo con datos espaciales, dichas herramientas permiten la captura, almacenamiento, análisis, modificación y visualización de los datos espaciales en la cartografía.

A continuación se expones varias definiciones citadas por diferentes autores, estas son:

- Los Sistema de Información Geográfica (SIG) son un “conjunto de métodos, herramientas y datos que están diseñados para actuar coordinada y lógicamente para capturar, almacenar, analizar, transformar y presentar toda la información geográfica y de sus atributos con el fin de satisfacer múltiples propósitos. Los SIG son una tecnología que permiten gestionar y analizar la información espacial, que surgió como resultado de la necesidad de disponer rápidamente de información para resolver problemas y contestar a preguntas de modo inmediato”. **(7)**

- "Los Sistema de Información Geográfica (SIG) se definen como el conjunto de herramientas para reunir, introducir, almacenar, recuperar, transformar y cartografiar datos espaciales sobre el mundo real para un conjunto particular de objetivos". **(8)**
- "Sistemas de Información Geográfica es el conjunto de mapas de la misma porción del territorio, donde un lugar concreto tiene la misma localización en todos los mapas incluidos en el sistema de información, resultando posible realizar análisis de sus características espaciales y temáticas para obtener un mejor conocimiento de esa zona." **(1)**
- Los sistemas de información geográfica como los conjuntos de instrumentos y métodos especialmente dispuestos para capturar, almacenar, transformar y presentar información geográfica o territorial referenciada al mundo real. **(9)**

1.1.2. Grafo.

A continuación se exponen algunas definiciones necesarias para la comprensión de la solución sobre la teoría de grafos según Diestel **(10)**.

Definición de grafo

Un grafo es un par $G = (V, E)$ que satisface que $E \subseteq VxV$, por lo que los elementos de E son subconjuntos de pares de elementos de V. Para evitar ambigüedades en la notación, siempre se asume tácitamente que $V \cap E = \emptyset$. Los elementos de V son los vértices (nodos o puntos) del grafo G, los elementos de E son las aristas(líneas).

Dos vértices $v1, v2$ de un grafo G son adyacentes o vecinos, si $(v1, v2)$ es una arista de G.

Definición de camino

Se denomina camino desde el vértice v_i al vértice v_j en un grafo $G = (V, E)$ a la secuencia de vértices:

$CA = v_1, v_2, \ldots, v_k$, tal que $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k) \in E, vi = vj, vj = vk.$

Definición: Se denomina longitud de un camino a la suma de los costos de todas las aristas presentes en el mismo y se representa de la siguiente forma:

$$|CA| = \sum_{n=1}^{k-1} c_{n(n\mp1)}$$

En el caso de grafos no ponderados, la longitud del camino se puede calcular como la cantidad de nodos presentes en el mismo menos 1, o sea:

$|CA| = k - 1$.

Definición: El problema de la búsqueda de camino mínimo entre dos vértices v_t y v_j de un grafo consiste en encontrar un camino desde v_t hasta v_j tal que se minimice la longitud del mismo.

Definición: Los arcos que unen el mismo par de vértices serán denominados arcos múltiples. Según esta definición, los arcos múltiples de un grafo G no son más que los elementos repetidos en el multiconjunto E. **(11)**

Definición: Denominaremos lazo a una arista que une un vértice con él mismo. **(11)**

Definición: Si un grafo no tiene lazos ni arcos múltiples diremos que es un grafo simple. **(11)**

Definición: Sean $G = (V, E)$ un grafo sin lazos y v_1 un vértice de G. Llamaremos grado o valencia del vértice v_1 al número de aristas incidentes en v_1. En caso de que G sea un grafo con lazos, cada lazo aportará dos al grado del vértice al cual es incidente. **(11)**

Definición: Diremos que un camino C es simple sí, todos los vértices son diferentes. **(11)**

1.2. Análisis de las soluciones existentes.

Los SIG son herramientas que proporcionan funcionalidades para realizar análisis de redes, con ello la búsqueda de caminos mínimos. Para ello es muy frecuente que estos sistemas utilicen un Sistema Gestor de Base de Datos (SGBD) en el cual se almacene la cartografía en la que se desea realizar dicho análisis. En el marco de este trabajo se toma como SGBD el PostgreSQL. **(12)**

En la actualidad, el SGBD PostgreSQL es uno de los sistemas gestores de base de datos que más se utiliza en el desarrollo de software libre en el mundo, por su capacidad de manejar grandes cantidades de datos. El SGBD PostgreSQL cuenta con la librería PostGIS la cual le añade soporte para el trabajo con datos espaciales. Esta extensión tiene varias funciones implementadas para el trabajo con datos espaciales.

1.2.1. pgRouting.

Para el análisis de redes sobre cartografías almacenadas en el SGBD PostgreSQL, se cuenta con la extensión pgRouting **(5)**, que es muy eficiente para redes pequeñas o medianas y para cálculos sencillos **(13)**. La biblioteca pgRouting utiliza las potencialidades que brindan las consultas relacionales y espaciales, pero tiene que cargar el grafo completo en memoria por lo que con un grafo con millones de nodos, puede provocar un bajo rendimiento de la aplicación **(14)**; además pgRouting utiliza un modelo de grafo demasiado simple, poco escalable y no puede usarse en entorno multi-hilo. También presenta problemas en la búsqueda de caminos cuando el origen o el destino no coinciden con una intersección de calles debido a que busca la arista más cercana al punto, sin tener en cuenta si desde dicho punto se puede acceder o no a la calle que representa la arista **(15)**. También representa y analiza las redes de forma plana, lo que trae como consecuencia que cuando dos calles están a diferentes niveles el sistema no distingue si se intersectan o no, lo que puede traer problemas al realizar los diferentes análisis de redes. Estas deficiencias inciden directamente en el análisis de redes, lo cual se debe a que el modelo utilizado no es capaz de representar de forma natural las redes existentes en una cartografía.

1.2.2. gvSIG

A pesar de las potencialidades que puede brindar el SGBD PostgreSQL no todos los SIG lo utilizan. En Europa existe en la actualidad un auge por la creación de SIG. En España un grupo de programadores crearon el SIG de escritorio gvSIG **(16)**. La creación de este sistema estuvo sujeta a que anteriormente en España se utilizaba ArcGIS **(17)**, del cual solamente se explotaba un pequeño por ciento de sus funcionalidades. El sistema gvSIG es un software capaz de cubrir el 95% de las necesidades de sus técnicos de SIG, ahorrándose cientos de licencias de ESRI **(17)**.

Este sistema cuenta con un módulo de análisis de redes que permite cargar una topología, tomar una red desde fichero, etc. Este módulo le da soporte matemático al análisis de redes y el mismo define a un grafo como un conjunto de vértices o nodos unidos por enlaces llamados aristas. Los grafos se guardan internamente en base a listas de adyacencia para economizar memoria. Básicamente consiste en que cada nodo tiene una lista con las aristas que salen de ese nodo.

Para unificar algoritmos, se parte siempre de que el grafo en cuestión es un grafo dirigido. De ahí que, para simular los tramos de doble sentido, se generen 2 aristas por cada tramo **(18)**.

Teniendo en cuenta que el módulo de análisis de redes de gvSIG realiza el procesamiento en la memoria RAM, lo cual no es conveniente para grafos grandes, se anula la utilización de esta solución para resolver el problema planteado en el marco de esta tesis.

1.2.3. ArcGIS Network Analyst.

ArcGIS **(19)** es el nombre de un conjunto de productos de software en el campo de los SIG, producido y comercializado por el Enviromental Systems Research Institute (ESRI) **(20)**, bajo el nombre genérico ArcGIS se agrupan varias aplicaciones para la captura, edición, análisis, tratamiento, diseño, publicación e impresión de información geográfica **(21)**. Estas aplicaciones se engloban en familias temáticas como ArcGIS Server, para la publicación y gestión web, o ArcGIS Móvil para la captura y gestión de información en campo.

ArcGIS cuenta con la extensión ArcGIS Network Analyst para el análisis de redes. Este módulo como cualquier aplicación de ESRI o de cualquier software privativo **(22)**, no brinda al público información sobre su desarrollo, por lo que muchas de sus características y formas de implementar las funcionalidades no están disponibles para todo los usuarios de manera general, es algo que solamente el equipo de desarrollo conoce, por ejemplo los manuales de ArcGIS están en inglés y tienen licencia copyright por lo que no puede usarse sin pagarle por dicha licencia a ESRI **(17)**. En este caso no se puede conocer de qué forma implementa el módulo de análisis de redes por lo que no es posible utilizar este conocimiento para darle solución al problema planteado.

1.2.4. Directed Graph Library.

Otro de los SIG que se ha desarrollado a nivel mundial es el SIG GRASS el cual cuenta para el análisis de redes con la librería Directed Graph Library **(23)**. Ésta solo implementa los algoritmos de complejidad polinomial camino más corto, expansión en profundidad y expansión mínima, pero como esta librería aún se encuentra en desarrollo no incorpora el almacenamiento de grafo, es decir, los análisis los realiza en memoria interna. Esto no es conveniente para un grafo con un número grande de nodos y como uno de los objetivo de este trabajo es la búsqueda de caminos mínimos en un grafo grande, no se toma como solución al problema esta librería.

1.2.5. IDELabRoute.

IDELabRoute es una Librería para la gestión de grafos desarrollada en el Laboratorio de Infraestructuras de Datos Espaciales (IDELab) **(24)**. Esta es una librería que permite realizar análisis de grafos y para ello puede utilizar tres criterios para el manejo de la memoria **(25)**:

- AllInMemoryManager: En este gestor de memoria se carga el grafo completo en memoria, lo que no es conveniente para grafos grandes.
- AllInMemoryExternalSourceMemoryManager: Este gestor de memoria utiliza almacenamiento persistente, externo, pero como el otro tipo también carga el grafo completo en memoria, lo que tampoco es conveniente.
- BasicExternalSourceMemoryManager: Este tipo de gestor de memoria es el más óptimo de todos. Carga en memoria solo los elementos solicitados, pero esto lo hace a costa de un aumento de las peticiones, lo que no es conveniente ya que el aumento del número de peticiones tiene como consecuencia demoras en el tiempo de respuesta del sistema.

De acuerdo con la explicación anterior ninguno de los mecanismos de manejo de memoria de la librería IDELabRoute son eficientes para el trabajo con un grafo grande, por lo que no es conveniente la utilización de esta librería para el desarrollo de este trabajo.

1.3. Fundamentación de las tecnologías a utilizar.

A continuación se describen las herramientas que se utilizarán para el desarrollo del sistema propuesto.

1.3.1. Java.

Se escoge el lenguaje de programación Java para el desarrollo del sistema ya que es un lenguaje de código abierto. Al ser un lenguaje libre los usuarios no deben preocuparse por pagar los impuestos de las patentes de cada año **(26)**. Puede ser ejecutado en múltiples plataformas, permite crear programas modulares y códigos reutilizables **(27).**

1.3.2. Sistema Gestor de Base de Datos.

Un gestor de base de datos o sistema de gestión de base de datos (SGBD o DBMS) es un software que permite introducir, organizar y recuperar la información de las base de datos; en definitiva, administrarlas. Es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y

seguridad **(28)**. La selección de Java como lenguaje de programación para implementar el sistema propuesto también se debe a que el sistema de almacenamiento persistente para grafo que se utiliza en la implementación de la solución, es el NEO4J **(29)** y el mismo está implementado en Java.

1.3.3. PostgreSQL.

PostgreSQL es un Sistema Gestor de Base de Datos Objeto-Relacional (ORDBMS) basado en POSTGRES. **(30)** La versión 4.21, se desarrolló en la Universidad de California en el Departamento de Ciencias de la Computación de Berkeley.

El ORDBMS POST-GRES fue pionera en muchos conceptos, aunque sólo estuvo disponible en algunos sistemas de base de datos comerciales.

PostgreSQL es un descendiente de código abierto del código original de Berkeley. Soporta una gran parte de las consultas SQL estándar.

Debido a la libertad de la licencia, PostgreSQL puede ser usado, modificado y distribuido por todo el mundo libre de carga para cualquier propósito, ya sea privado, comercial o académico. **(31)**

El SGBD PostgreSQL cuenta con la extensión PostGIS la cual le añade soporte para el trabajo con datos espaciales. **(30)** Esta extensión cuenta con varias funciones implementadas para el trabajo con datos geográficos. Por esta razón se escogió el SGBD PostgreSQL para almacenar las cartografías que sirven como entrada para obtener el grafo que representa la red de viales de un mapa.

1.3.4. Sistema para almacenamiento persistente de grafo: Neo4j.

Para la gestión de los datos que se utilizan en el análisis de redes, se dispone del motor de persistencia embebido (MPE) Neo4j **(29)**. Para la implementación del sistema se seleccionó el MPE Neo4j ya que es un sistema de almacenamiento basado en disco, completamente transaccional que almacena datos estructurados en grafos más que en tablas. El MPE Neo4j, es uno de los gestores libres de éste tipo más utilizados a nivel mundial. El mismo cuenta con la implementación de varios algoritmos para el trabajo con grafos **(32)**.

1.3.5. Metodología ágil de desarrollo de software.

El proceso de desarrollo de software posee como principal propósito la eficiente producción de un software que cumpla con los requisitos establecidos por el cliente.

No existe un proceso de desarrollo de software que sea único y a la vez eficiente, por lo que no existe un proceso de desarrollo de software universal que sea efectivo para cualquier desarrollo de software. **(33)**

1.3.5.1. Programación Extrema.

Extreme Programming una metodología de desarrollo de software que promueven prácticas que son adaptativas en vez de predictivas, centradas en las personas o en los equipos, iterativas, orientadas hacia prestaciones y hacia la entrega, de comunicación intensiva y requieren que el negocio se involucre en forma directa. **(34)**

La metodología XP se centra en el trabajo en equipo, contribuye a lograr una buena comunicación entre el cliente y el equipo de desarrollo, trayendo consigo un mejor clima en el área de trabajo; instruye a sus desarrolladores pues una de sus principales tareas es el aprendizaje de los mismos. XP se fundamenta de una retroalimentación continua entre el equipo de trabajo y el cliente. **(35)**

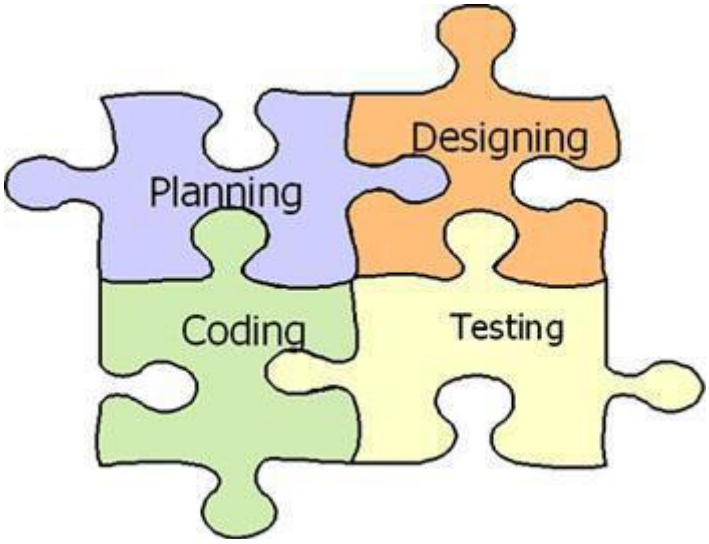


Figura 1: Fases de la Metodología XP

Las características esenciales de XP son las siguientes: historias de usuario (HU), roles, proceso y prácticas. **(36)**

HU: es la técnica utilizada para especificar las necesidades del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean funcionales o no funcionales. El tratamiento de las HU es muy dinámico y flexible. Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Roles:

- Programador: el programador escribe las pruebas unitarias y produce el código del sistema.
- Cliente: escribe las HU y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las HU y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.
- Encargado de pruebas: ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- Encargado de seguimiento: proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, con el objetivo de mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.
- Entrenador: es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas de XP y se siga el proceso correctamente.
- Consultor: es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto.
- Gestor: es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es la coordinación.

Proceso: el ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al principio.
- El ciclo de vida ideal de XP consiste de seis fases:
- Exploración.
- Planificación de entrega (*Release*).
- Iteraciones.
- Producción.
- Mantenimiento.
- Muerte del Proyecto.

Prácticas: la principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione.

El objetivo de XP es muy simple: la satisfacción del cliente. Esta metodología trata de dar al cliente el software que él necesita y en el momento que lo necesita. Por tanto, se debe responder muy rápido a las necesidades del cliente, incluso cuando los cambios sean al final de ciclo de la programación. XP se encarga de potenciar al máximo el trabajo en grupo. Tanto los jefes de proyecto, los clientes y los desarrolladores, son parte del equipo y están involucrados en el desarrollo de software.

XP mejora un proyecto de software de cuatro formas esenciales: comunicación, simplicidad, retroalimentación y coraje. Una de las características de la programación extrema es que es imposible prever todo antes de comenzar a programar; es imposible o si lo fuera es demasiado costoso e innecesario, ya que muchas veces se gasta demasiado tiempo y recursos en cambiar la documentación de la planificación para que se parezca al código. Para evitar esto, XP intenta implementar una forma de trabajo donde se adapte fácilmente a las circunstancias. **(35)**

Elementos que se tuvieron en cuenta la selección de la metodología XP.

Se llegó a la conclusión de que XP es la mejor opción para el desarrollo de la solución que se propone porque, si se analiza esta metodología, se observará rápidamente que es la que se adapta al trabajo en cuestión, ya que en XP existe un conjunto de prácticas que si se llevan a cabo se lograrán los objetivos trazados. Al existir dos personas para el desarrollo de este sistema, se puede implementar la práctica de Programación en pares, la misma permite que dos programadores se realicen pruebas entre sí en sus códigos. También el cliente se encuentra en todo momento en contacto con los desarrolladores del software.

Otras de las prácticas son las entregas pequeñas y frecuentes, la cual permite un control constante de las tareas y en cortos plazos de tiempo se pueden tener resultados. En XP no se implementa nada que no se necesite, es decir se minimiza la documentación.

Una de las prácticas más importante y que está muy ligada al proceso de formación actual de la universidad es la de Paso Sostenible. Aquí se plantea que en la semana

se debe trabajar solo 40 horas, lo que permite que el proyecto se adapte a las horas de producción que se necesitan tener por cada estudiante de la UCI.

Para tomar la decisión también se analizaron los roles que se necesitan para cada metodología y XP es la que se adapta a nuestras necesidades. XP motiva un conjunto de valores los cuales son muy necesarios para el trabajo en equipo, crea valores comunes que les permite al equipo trabajar por un objetivo común.

1.4. Conclusiones

En el presente capítulo se han abordado las tendencias y aspectos fundamentales de los diferentes sistemas que permiten realizar análisis de redes en la actualidad, desde un punto de vista crítico, arribando a la conclusión de la importancia de realizar la implementación de un algoritmo que sea capaz de reducir un grafo y realizar búsquedas de caminos mínimos en el grafo reducido de forma eficiente, en el análisis de las soluciones existentes se observa que estas tienen incorporada un módulo o biblioteca capaz de realizar el análisis de redes, pero en algunos de los casos no se cuenta con la información de cómo lo hacen, en otros la solución que brindan no es la más eficiente, por lo que es necesario llevar a cabo la implementación del sistema propuesto en el presente trabajo.

2

CAPÍTULO

2. Descripción de la solución propuesta.

En el presente capítulo se describen las principales características del sistema. Se hace mención a las fases de exploración y planificación, propias de la metodología ágil de desarrollo utilizada para la implementación del sistema. Se muestran las historias de usuarios que fueron escritas por el cliente y se definen las cuatro iteraciones en la que se va a implementar el sistema, construyéndose además, el Plan de Entrega del sistema.

2.1. Descripción de los procesos vinculados al campo de acción.

En la Universidad de la Ciencias Informáticas, en el proyecto SIG-Desktop del centro GEySED se lleva a cabo el proceso de desarrollo de un SIG. Una de las funcionalidades que brinda un SIG es la búsqueda de caminos mínimos, lo cual está vinculado con el tema de análisis de redes y con la misma se pueden resolver problemas, por ejemplo, de planificación de transporte, etc. En este proyecto no se tiene en cuenta la creación de un módulo o sistema especializado en el análisis de redes el cual puedan utilizar los SIG que se crean en este departamento y es por ello que utilizan como solución la librería pgRouting la cual no es eficiente para grafos grandes ya que no cuenta con soporte teórico y utiliza como modelo para realizar estos análisis la propia cartografía.

2.2. Propuesta de solución.

Un SIG para dar respuesta a algún tipo de análisis de ruta puede utilizar a propia cartografía como modelo para realizar dicho análisis, tal es el caso del uso pgRouting en sistemas como OpenStreetMap **(37)** otra forma sería la utilización de un modelo matemático equivalente a la red a analizar. En este trabajo se propone la creación de un sistema para el análisis de redes en los SIG que se desarrollan en el proyecto SIG-

Desktop del centro de desarrollo GEySED de la UCI donde se utiliza el modelo matemático grafo para realizar análisis de redes. Para el desarrollo del sistema se propone la creación de dos módulos, uno para la obtención del grafo que representa una red de viales y otro para la reducción de grafo y búsqueda de caminos mínimos sobre el grafo reducido.

En la Figura 2 se presenta los componentes principales del sistema desarrollado, a continuación se realiza una descripción de dichos componentes.

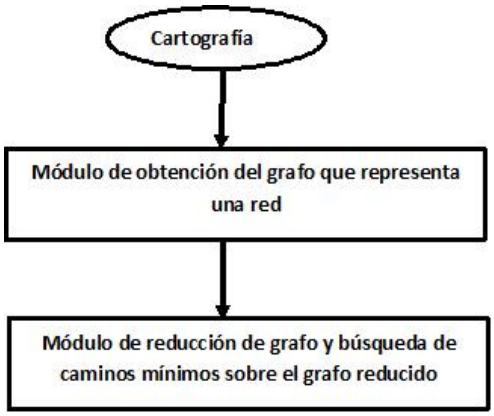


Figura 2: Estructura del sistema para el análisis de redes

Módulo de obtención de grafo:

Para el sistema desarrollado es importante tener el grafo que representa la red de viales de una cartografía ya que para manipular o analizar la información de un mapa, es más ventajoso hacerlo a través de un modelo matemático equivalente al mapa, en este caso un grafo, donde los objetos del mapa y sus relaciones representen los vértices y las aristas del grafo. Para ello en este módulo se implementa un algoritmo ávido que encuentra las intersecciones entre las calles de la red y los puntos iniciales y finales de cada calle, obteniendo así los nodos del grafo.

Para encontrar las aristas se toman por cada una de las calles de la red, los nodos que representan las intersecciones de dicha calle con las otras que conforman la red y se separan cada uno de estos nodos. Cada tramo obtenido pasa a ser una arista del grafo.

A continuación se muestra el algoritmo implementado para la obtención del grafo que representa una red modelada en una cartografía.

Algoritmo 1 Obtener un grafo que representa una red de viales a partir de un cartografía

Entrada: Una cartografía que contenga una capa de viales

Salida: Un grafo que representa la cartografía de entrada

1. Almacenar en la variable *calles* todas las calles existentes en la capa de viales de la cartografía
2. *listavertices*={}
3. *listaaristas*={}
4. **Para todo** $C_1 \in$ *calles* **hacer**

5. *auxiliar*={}
6. **Para todo** $C_2 \in$ *calles* **hacer**
7. *listapuntos*= DeterminarIntersecciones(C_1, C_2)
8. *auxiliar*.Adicionar(*listapuntos*)
9. **Para todo** *puntos* $\in$ *listapuntos* **hacer**
10. *listavertices.*Adicionar(*listapuntos*)
11. **Fin para**
12. **Fin para**
13. *listaaristas*=DeterminarAristas(C_1, *auxiliar*)
14. **Fin para**
15. **Retornar** CrearGrafo(*listavertices, listaaristas*)

Este primer algoritmo permite obtener el grafo que representa la red de viales de una cartografía, para esto es necesario obtener los nodos, que no son más que las intersecciones entre dos geometrías (líneas en este caso) y luego calcular las aristas que unen los nodos encontrados.

Para determinar las aristas se implementó el siguiente algoritmo:

Algoritmo 2 Determinar las aristas del grafo que representa la red

Entrada: Una calle *c* y una lista de puntos *puntos*

Salida: Un conjunto de aristas vinculadas a la red modelada en la cartografía

1. *aristas*={}
2. Calcular la distancia desde el primer punto de la calle *c* hasta cada punto de la lista *puntos* haciendo uso de la geometría *c*
3. Ordenar las lista de los puntos según la distancia calculada
4. *puntos*.Insertar(0, *c[0]*) {Insertar en la primera posición de la lista de puntos el primer punto de la calle que se analiza}
5. *puntos*.Adicionar(*c[cantidadpuntos]*){Adicionar al final de la lista de puntos, el último punto *c*}
6. **Para** *i=0* hasta *i<longitud(puntos)*-1 **hacer**
7. *a*=ExtraerGeometría(*c, puntos[i], puntos[i+1]*) {Devuelve la geometría (una línea) que une dos puntos que están sobre la calle *c*}
8. *aristas.*Adicionar(*puntos[i], puntos[i+1], len(a), a*)
9. **Fin para**
10. **Retornar** *aristas*

Este módulo facilita, a partir de la red de viales de una cartografía que se encuentra almacenada en una base de datos PostgreSQL, un modelo matemático equivalente a la red de viales de un mapa, en este caso un grafo, el cual será almacenado en el motor de persistencia para almacenamiento de grafos Neo4j. Es importante señalar que la cartografía que es entrada de este módulo debe estar almacenada en una base de datos PostgreSQL. Para el desarrollo de éste módulo se utilizó la herramienta shp2psql, la misma permite exportar una cartografía en formato shape **(22)** a PostgreSQL. También se utilizaron las funciones que brinda la librería PostGIS, la cuál es la encargada de darle soporte para el trabajo con datos espaciales al SGBD PostgreSQL.

El sistema QGIS **(38)** es el sistema que se utiliza en el proyecto SIG-Desktop como base para desarrollar una plataforma de escritorio, donde posteriormente se desarrollará personalizaciones para determinados clientes. El QGIS cuenta con una funcionalidad para obtener las intersecciones de las calles de una cartografía. Esta funcionalidad obtiene dichas intersecciones de forma repetida, lo cual se debe a que cuando se tiene una red de viales, por ejemplo la Figura 3, al calcular las intersecciones se tiene lo siguiente:

- Cuando se calcula las intersecciones entre las calles ***A*** y ***1***, se obtiene como resultado el punto ***(A, 1).***

- Cuando se calcula las intersecciones entre las calles ***A*** y ***2***, se obtiene como resultado el punto ***(A, 2).***

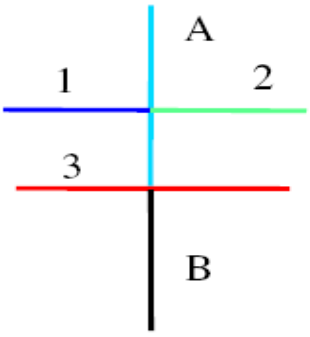


Figura 3: Ejemplo de red de viales

Los puntos ***(A,1)*** y ***(A,2)*** representan el mismo objeto geográfico, que coincide con la intersección de las calles ***A, 1*** y ***2***, en este caso se puede afirmar, que desde el punto de vista del análisis de la red, se debe tener en cuenta como una sola intersección,

esto aumenta su importancia cuando el objetivo es modelar la red (presente en la cartografía) a través de un grafo para realizar análisis posteriores.

Lo mismo ocurre con la intersección entre las calles ***A, B*** y ***3***. Debido a esta situación, cuando se calcula las intersecciones de una calle con el resto de las mismas, hay que buscar si el punto de intersección resultante ya fue encontrado anteriormente, para evitar puntos (vértices) repetidos.

Módulo de reducción de grafos y búsqueda de camino mínimo sobre el grafo reducido:

Partiendo de que en una red de viales pueden existir miles de nodos y aristas, es necesario, para optimizar el tiempo de respuesta del sistema encontrar una forma de reducir el grafo que representa a la red de viales de un mapa. Para ello se implementa el algoritmo de reducción de grafo sin pérdida de información, definido en el artículo "*Aplicación de las gramáticas de grafo en Sistemas de Información Geográfica*" **(39)**. Este módulo se encarga de tomar un grafo almacenado en el motor de persistencia para almacenamiento de grafos Neo4j y le aplica el número de reducciones especificadas según las necesidades del usuario. Por ejemplo, si en una cartografía que representa la red de viales de Cuba, se quiere saber cuál es el camino mínimo para llegar a la provincia Santiago de Cuba tomando como partida la provincia de la Habana, cualquier algoritmo de grafo para la búsqueda de camino mínimo debe analizar todos los objetos del mapa (los nodos del grafo que representa la red de viales del mapa). Sin embargo, si se aplica una reducción al grafo que representa la red de viales, al realizar una búsqueda de camino mínimo sobre el grafo reducido, la cantidad de nodos de este va a ser menor que la cantidad de nodos del grafo sin reducir. Esto es posible ya que en una determinada escala del mapa, existen objetos que pueden perder importancia o simplemente no son visibles. La implementación de este algoritmo es importante ya que permitiría realizar análisis de redes en dependencia de la escala en la que se muestre un determinado mapa. La Figura 4 muestra un ejemplo de grafo que representa la red de viales de Cuba después de aplicar una reducción agrupando los vértices que pertenecen a una misma provincia.

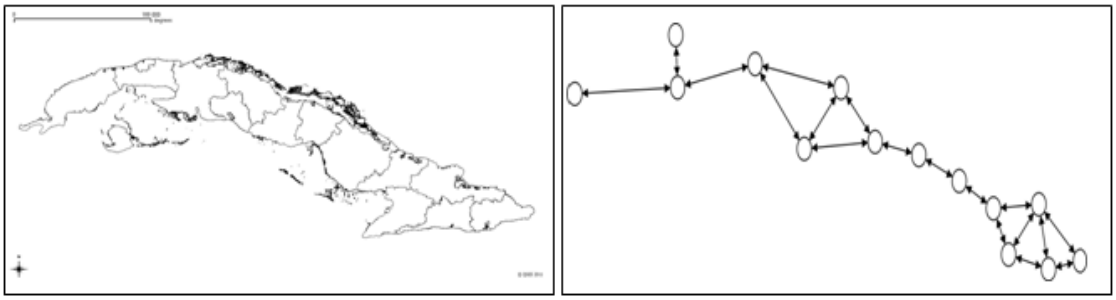


Figura 4: Representación de la red de viales de Cuba una vez aplicada una reducción por provincias

En este módulo también se propone la implementación de un algoritmo de búsqueda de caminos mínimos sobre un grafo al que se le haya aplicado una o más reducciones, el cual será más eficiente ya que se trabaja con un grafo con una menor cantidad de nodos. Para ello se realiza una modificación del algoritmo de Dijkstra, de forma tal que permita realizar búsquedas de caminos mínimos sobre un grafo reducido. Además se implementa la función ***f*** la cual determina el menor costo del camino desde un nodo ***A*** a un nodo ***C*** pasando por ***B*** siendo ***B*** un nodo reducido.

2.3. Personal relacionado con el sistema.

Para determinar al personal relacionado con el sistema se hace referencia, a todas aquellas personas que se encuentren de una forma u otra vinculada con el sistema, ya sean las personas que interactúan en el proceso de desarrollo, así como las que interactúan con el mismo.

En la tabla 1 se muestra el personal relacionado con el sistema.

Tabla 1: Personal relacionado con el sistema

Personal relacionado con el sistema	Justificación
Desarrolladores	Son las personas que se encargan de implementar todas las características definidas por el cliente para lograr el sistema.
Rol de usuarios	Son todas aquellas personas que interactúan con los contenidos generados por los desarrolladores.

2.4. Fase I: Exploración

La metodología XP en su primera la fase Exploración, define el alcance general del sistema ya que los clientes describen las historias de usuarios. Esta fase permite que

el equipo de desarrollo se familiarice con las tecnologías y herramientas que se utilizarán para el desarrollo del sistema. Se realizan estimaciones que se consideran como estimaciones primarias ya que las mismas podrían variar al analizar con más detalles cada iteración. Esta fase demora pocas semanas o pocos meses, dependiendo de la experiencia de los programadores con las tecnologías de desarrollo **(39)**.

2.4.1. Historias de Usuarios.

Las Historias de Usuarios (HU) son escritas por los clientes, aunque los desarrolladores pueden brindar su ayuda en la identificación de las mismas. Las descripciones de las HU son cortas y escritas en lenguaje del usuario. Se utilizan en la parte de prueba para verificar que el sistema cumpla con las necesidades de los usuarios. El tiempo ideal para el desarrollo de las historias de usuarios es 1 y 3 semanas y para esto cuando llega el momento de llevar a cabo la implementación de las mismas se reúnen el cliente y el equipo de desarrollo para concretar y detallar lo que tiene que hacer dicha historia de usuario.

A continuación como resultado de la Fase de exploración se exponen las historias de usuarios identificadas para lograr el desarrollo de nuestro sistema.

Historias de Usuarios	
Número: 1	**Nombre**: Reducir Grafo
Usuario: Cliente	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo**: Alto
Puntos de Estimación: 3	**Iteración Asignada**: 1
Descripción: Permite reducir la cantidad de nodos y aristas de un grafo teniendo en cuenta que los nodos cumplan con una propiedad determinada.	
Observaciones: Al reducir el grafo disminuye el tiempo de respuesta del algoritmo de búsqueda de caminos mínimos.	

Tabla 2: HU Reducir Grafo

Historias de Usuarios	
Número: 2	**Nombre**: Gestionar fichero de regla
Usuario: Cliente	
Prioridad en Negocio: Media	**Riesgo en Desarrollo**: Medio

Puntos de Estimación: 1	**Iteración Asignada**: 2
Descripción: Permite cargar o almacenar una regla de reescritura en o desde un fichero.	
Observaciones: Una regla de reescritura de grafo es aplicada a un grafo para sustituir un subgrafo por otro. Una regla de reescritura, provee información de empotrado, la cual es usada para conocer la forma de conectar el grafo reducido con el resto del grafo.	

Tabla 3: HU Gestionar ficheros de regla

Historias de Usuarios	
Número: 3	**Nombre**: Gestionar fichero de partición.
Usuario: Cliente	
Prioridad en Negocio: Medio	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 2
Descripción: Permite cargar o almacenar una partición en o desde un fichero.	
Observaciones: Antes de aplicar un algoritmo de reducción a un grafo, es necesario primeramente particionar dicho grafo. Una partición contiene un conjunto de clases de equivalencia, las cuales agrupan los vértices que cumplen con la propiedad especificada para la reducción.	

Tabla 4: HU Gestionar ficheros de partición

Historias de Usuarios	
Número: 4	**Nombre**: Crear información de empotrado
Usuario: Cliente	
Prioridad en Negocio: Medio	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 2
Descripción: Permite tener la información para sustituir un subgrafo por otro.	
Observaciones: Cuando se crea una regla de reescritura, ésta genera información para sustituir un grafo por otro, dicha información contiene el peso de las arista.	

Tabla 5: HU Crear información

Historias de Usuarios	
Número: 5	**Nombre**: Búsqueda de caminos mínimos.
Usuario: Cliente	

Prioridad en Negocio: Media	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 2	**Iteración Asignada**: 3
Descripción: Permite determinar el camino más corto entre dos puntos.	
Observaciones: Cuando se aplique la reducción del grafo se realizará la búsqueda de caminos mínimos.	

Tabla 6: HU Búsqueda de caminos mínimos

Historias de Usuarios	
Número: 6	**Nombre**: Brindar servicio de búsqueda de caminos mínimos.
Usuario: Cliente	
Prioridad en Negocio: Media	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 3
Descripción: Permite la integración del sistema con los SIG.	
Observaciones:	

Tabla 7: HU Crear servicio Web

Historias de Usuarios	
Número: 7	**Nombre**: Generar grafo desde cartografía.
Usuario: Cliente	
Prioridad en Negocio: Baja	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 4
Descripción: Permite obtener el grafo que corresponde a la capa de viales de una cartografía.	
Observaciones: Es necesario obtener el grafo que representa la capa de viales para poder aplicar el algoritmo de reducción.	

Tabla 8: HU Generar grafo desde cartografía.

Historias de Usuarios	
Número: 8	**Nombre**: Guardar grafo en fichero .dot.
Usuario: Cliente	
Prioridad en Negocio: Baja	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 4
Descripción: Permite almacenar un grafo en un fichero con formato .dot.	

Observaciones: Cuando se genere el grafo que representa la capa de viales de una cartografía, es necesario almacenarlo en un fichero con soporte da almacenamiento para grafo, es este caso .dot.

Tabla 9: HU Guardar grafo en fichero .dot.

Historias de Usuarios	
Número: 9	**Nombre**: Cargar grafo de fichero .dot a base de datos neo4J
Usuario: Cliente	
Prioridad en Negocio: Baja	**Riesgo en Desarrollo**: Medio
Puntos de Estimación: 1	**Iteración Asignada**: 4
Descripción: Permite cargar el grafo que se encuentra en un fichero .dot para una base de datos neo4j.	
Observaciones: Todo el trabajo con el grafo que representa la capa de viales se realiza en la base de datos.	

Tabla 10: HU Cargar grafo a BD Neo4J.

2.5. Fase II: Planificación.

En la Fase de planificación, el cliente define la prioridad de cada Historia de usuario que el mismo haya descrito en la fase anterior, luego de esto los programadores realizan una estimación del esfuerzo necesario para cada Historia de Usuario. Se determina un cronograma en conjunto con los clientes y programadores, según los acuerdos tomados.

El equipo de programadores mantienen un registro de la "velocidad de desarrollo", en la que llevan un control de la suma de todos los puntos correspondiente a las HU que fueron determinadas en la última iteración. Este registro de velocidad es utilizada para determinar el tiempo que se llevará implementar un conjunto de HU.

La velocidad del proyecto es de gran utilidad para establecer la cantidad de HU que se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementarlas.

2.5.1. Estimación de esfuerzos por Historias de usuarios.

La estimación de esfuerzo se expresa utilizando como medida el punto. Un punto se considera como una semana ideal de trabajo, donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción. Esta estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario.

Para el logro del sistema se ha realizado la estimación de esfuerzo por historias de usuario según el orden de prioridad:

Historias de Usuarios	Puntos de estimación
Reducir grafo	3
Gestionar fichero de regla	1
Gestionar fichero de partición	1
Crear información de empotrado	1
Búsqueda de caminos mínimos	2
Brindar servicio de búsqueda de caminos mínimos.	1
Generar grafo desde cartografía	1
Guardar grafo en fichero .dot	1
Cargar grafo a base de datos Neo4J	1

Tabla11: Estimación de esfuerzos por HU.

2.5.2. Plan de iteraciones

La Fase de planificación, contiene las iteraciones del sistema antes de que el mismo sea entregado. Al estar descritas las Historias de usuarios y los puntos de estimación de cada una de ellas, se procede a definir el Plan de iteraciones, que no es más que las partes en las que se dividirá la implementación. Hay que tener presente que el cliente es quien decide en todo momento las HU que serán implementadas en cada iteración. Todo proyecto que siga la metodología XP se ha de dividir en iteraciones de aproximadamente 3 semanas de duración. **(40)**

Se determina realizar el sistema en cuatros iteraciones en las que cada una posee una duración de tres semanas, las cuales quedan descritas a continuación:

Primera Iteración

En esta iteración tiene como objetivo darle cumplimiento a la HU # 1 que se define con el nombre de Reducir grafo, quedando implementada dicha HU, según la descripción realizada por el cliente. Igualmente se tendrá la primera versión de pruebas.

Segunda Iteración

El objetivo de esta iteración es implementar las siguientes Historias de Usuarios:

- HU #2 Gestionar fichero de regla.

- HU #3 Gestionar fichero de partición.
- HU #4 Crear información de empotrado.

Después de lograr la implementación de estas HU, las versiones de pruebas anteriores, junto a estas se han de mostrar al cliente.

Tercera Iteración

En esta iteración se implementará la HU #5 Búsqueda de caminos mínimos, la cual tiene como objetivo permitir realizar búsquedas de caminos mínimos y la HU #6 Brindar servicio de búsqueda de caminos mínimos, que mediante esta historia de usuario, el cliente puede acceder a las funcionalidad búsqueda de caminos mínimos. De igual manera se procede a desarrollar las pruebas al concluir la iteración.

Cuarta Iteración

En esta última iteración del sistema, se le darán cumplimiento a las siguientes Historias de usuarios:

- HU # 7 Generar grafo desde cartografía.
- HU #8 Guardar grafo en fichero .dot.
- HU #9 Cargar grafo a Base de Datos Neo4J.

Estas Historias de usuarios son integradas con el resultado de las iteraciones antes descritas y como beneficio de esta integración se obtendrá el Sistema para la reducción de grafos y búsqueda de camino mínimo.

2.5.3. Plan de duración de las iteraciones.

Para lograr el desarrollo de un proyecto, la metodología de desarrollo XP, describe un ciclo de vida, en el cual se crea el plan de duración para cada una de las iteraciones que se llevan a cabo para el desarrollo del sistema. Este plan tiene como objetivo fundamental mostrar el orden en que las Historias de usuarios han de ser implementadas, así como la duración de cada iteración definida y descrita anteriormente.

A continuación se muestra mediante una tabla el Plan de duración de las iteraciones:

Iteraciones	Orden de las Historias de Usuarios a implementar	Duración total de las iteraciones

Iteración 1	1. Reducir grafo	3 semanas
Iteración 2	1. Gestionar fichero de regla 2. Gestionar fichero de partición 3. Crear información de empotrado	3 semanas
Iteración 3	1. Búsqueda de caminos mínimos 2. Brindar servicio de búsqueda de caminos mínimos.	3 semanas
Iteración 4	1. Generar grafo desde cartografía. 2. Guardar grafo en fichero .dot. 3. Cargar grafo a base de datos Neo4J.	3 semanas

Tabla 12: Plan de duración de las iteraciones

2.6. Integración con otros sistemas

El sistema resultante de este trabajo será capaz de integrarse con los SIG que se desarrollan en el departamento de Geoinformática del centro GEySED. Las funcionalidades implementadas en el sistema propuesto se podrán utilizar desde otros sistemas a través de un servicio web para ello se creó un contrato en formato WSDL **(41)** en el cual se especifica la forma en la que se puede acceder al servicio web desarrollado.

2.7. Conclusiones.

Se ha abordado sobre la planificación del sistema concentrándose en la Historias de usuarios, plan de entrega, finalmente se arribaron a las siguientes conclusiones: el sistema deberá desarrollarse en cuatro iteraciones, además el sistema cuenta con un total de nueve HU, las cuales serán implementadas en cuatro iteraciones.

3

CAPÍTULO

3. Construcción de la solución propuesta.

La metodología XP, cuenta con la realización del diseño del sistema, para esto es necesaria la utilización de las Tarjetas C.R.C. Además XP plantea que la implementación del sistema debe realizarse de forma iterativa, logrando obtener al finalizar cada iteración un producto final, el cual es mostrado al cliente y aprobado por el mismo.

En el presente capítulo se detallan las cuatros iteraciones que se han de llevar a cabo durante la etapa de construcción del sistema. Además quedan plasmadas las tarjetas C.R.C y las tareas de ingeniería, las cuales constituyen pasos indispensables a seguir para la construcción del sistema.

3.1. Diseño de la solución propuesta.

La metodología XP, como metodología seleccionada para el desarrollo del sistema, utiliza técnicas como las llamadas Tarjetas CRC (Contenido, Responsabilidad, Colaboración), para lograr entender el diseño del sistema. **(42)**

3.1.1. Estructura del sistema.

Para lograr un mejor entendimiento del sistema se han de definir la estructura de los dos módulos con los que cuenta el sistema. Definiendo los paquetes que posee cada uno de estos módulos, así como las responsabilidades que han de tener cada uno de estos paquetes.

Para el módulo de Obtención de obtención de grafo, se define la siguiente estructura:

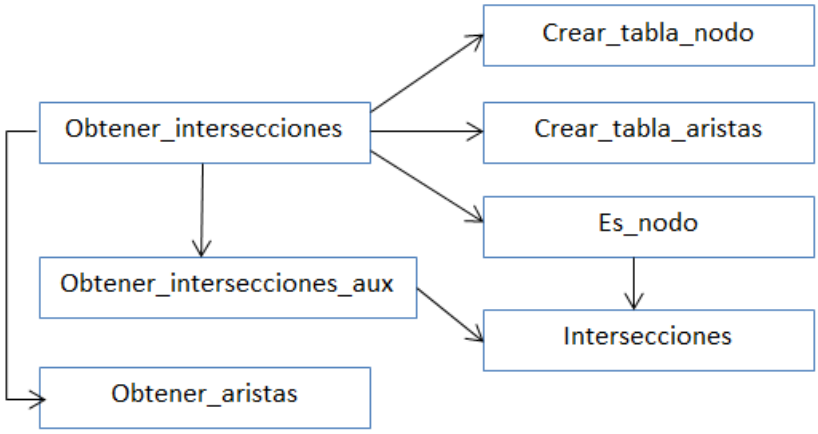


Figura 5: Estructura del módulo de obtención del grafo

Para el módulo de reducción de grafos y búsqueda de camino mínimo sobre el grafo reducido, se define la siguiente estructura:

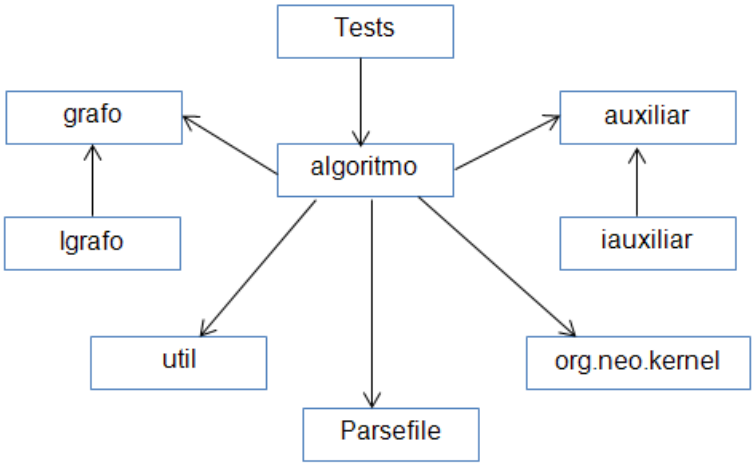


Figura 6: Estructura del Módulo de reducción de grafo

Con una descripción detallada de los paquetes cualquier desarrollador con un conocimiento del lenguaje de Java puede asumir el mantenimiento y seguimiento del sistema. Este módulo cuenta con nueves paquetes creados para el análisis de redes.

El paquete ***algoritmo*** tiene la clase con la implementación de los algoritmos para el trabajo con los grafos. Aquí se implementa el algoritmo de reducción de grafo, el de búsqueda de caminos mínimo, etc. (Ver Anexo 1)

El paquete **iauxilar** cuenta con las interfaces que sirven da apoyo para la implementación del sistema. El mismo se detalla en el Anexo 5

El paquete **auxiliar** es el encargado de implementar las interfaces definidas en el Anexo 5. Este paquete cuenta con las clases que sirven de apoyo al sistema. El mismo se detalla en el Anexo 2

El paquete **configuración** cuenta con los ficheros de configuración del sistema. El mismo se detalla en el Anexo 3

El paquete **grafo** es el encargado de implementar las clases que modelan las estructuras de datos del modelo utilizado, en este caso un grafo. El mismo se detalla en el Anexo 4

El paquete **Igrafo** contiene las interfaces necesarias que hay que implementar para modelar el modelo matemático utilizado en el sistema, en este caso un grafo. El cual se detalla en el Anexo 6

En el paquete **Relaciones** es donde se debe almacenar las relaciones de equivalencia que hay que implementar para aplicar una reducción a un grafo. El mismo se detalla en el Anexo 7

El paquete **Tests** cuenta con la implementación de las pruebas realizadas en el sistema. El mismo se detalla en el Anexo 8

El paquete **util** cuenta con un conjunto de clases que son de utilidad al sistema. El mismo se detalla en el Anexo 9

3.1.2. Tarjetas CRC.

Las Tarjetas CRC son una técnica para la representación de los sistemas, constituyen un puente de comunicación entre los diferentes participantes.

Las Tarjetas CRC se dividen en tres secciones que contienen del nombre de la clase, sus responsabilidades y colaboradores.

- Nombre de las clases: Se pone el nombre de la clase a la que se está describiendo.
- Responsabilidades: Funciones que contiene implementada la clase.
- Colaboradores: Son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades.

A continuación se muestras las tarjetas C.R.C correspondiente al desarrollo del sistema. En el Anexo 14 se describe la explicación de cada método de cada clase que se representan en las tarjetas C.R.C.

Grafo	
Responsabilidades	**Colaboraciones**
Crear Nodo	Arista
Buscar Nodo	Nodo Persistente
getNodoPersistenteById	Grafo Reducido
crear Arista	
getArista	
Cantidad Aristas	
Cantidad Nodos	
Dijkstra	
getIteratorNodes	

Tabla 13: Tarjeta C.R.C Grafo

Arista	
Responsabilidades	**Colaboraciones**
getNodoInicial	
getNodoFinal	
getCosto	
setCosto	
getNombre	
setNombre	
equals	
hashCode	
toString	

Tabla 14: Tarjeta C.R.C Arista

Nodo Persistente	
Responsabilidades	**Colaboraciones**
Equals	Etiqueta
hashCode	
toString	
getNombre	
setNombre	
getAdyacentes	

setEtiqueta getEtiqueta getArista compareTo	

Tabla 15: Tarjeta C.R.C Nodo Persistente

Grafo Reducido	
Responsabilidades	**Colaboraciones**
getNiveles getCantidadNiveles addRegla addNivel crearNodoComplejo toString	Nivel Nodo Complejo

Tabla 16: Tarjeta C.R.C Grafo Reducido

Nivel	
Responsabilidades	**Colaboraciones**
getReglas addRegla setNivel getNivel toString	

Tabla 17: Tarjeta C.R.C Nivel

Etiqueta	
Responsabilidades	**Colaboraciones**
getName getValue setValue toString	

Tabla 18: Tarjeta C.R.C Etiqueta

Nodo Complejo	
Responsabilidades	**Colaboraciones**
getRegla setRegla	Etiqueta Regla de Reescritura

Tabla 19: Tarjeta C.R.C Nodo Complejo

Regla de Reescritura	
Responsabilidades	**Colaboraciones**
getIn setSubGrafo getSubGrafo getInToString getOutToString getOut setIn setOut	

Tabla 20: Tarjeta C.R.C Regla de Reescritura

Información de Empotrado	
Responsabilidades	**Colaboraciones**
getC getU getV getVp setC setU setV setVp toString	

Tabla 21: Tarjeta C.R.C Información de Empotrado

Partición	
Responsabilidades	**Colaboraciones**
Crear Particion getElementos ImprimirParticion isDisjoin	Clase Equivalente

Tabla 22: Tarjeta C.R.C Clase Equivalente

Clase Equivalente	
Responsabilidades	**Colaboraciones**
getEtiqueta adicionar	Nodo Persistente

setEtiqueta getElementos toString compareTo	

Tabla 23: Tarjeta C.R.C Clase Equivalente

Algoritmo Reducción	
Responsabilidades	**Colaboraciones**
BCM Fx Reducir Grafo	Arista Nodo Persistente Nodo Complejo Grafo Reducido Grafo Información de Empotrado Regla de Reescritura Partición

Tabla 24: Tarjeta C.R.C Algoritmo Reducción

3.2. Desarrollo de las Iteraciones

En la fase de planificación de la metodología XP se detallaron cada una de las HU correspondientes a cada iteración según la selección del cliente. Durante el desarrollo de las iteraciones se va realizando una revisión del plan de iteraciones ya que el mismo puede o no ser modificado. Como parte de este plan se crean tareas para ayudar a organizar la implementación exitosa de la HU.

Estas tareas en la que se descomponen las HU son las llamadas tareas de ingeniería, que son asignadas a una persona perteneciente al equipo de desarrollo, que el mismo sea quien responda por la implementación asignada. Estas tareas son escritas por los programados y no por el cliente, ya que son para uso estricto de los programadores.

La planificación que se llevó a cabo para el desarrollo del sistema, está compuesta por cuatro iteraciones, permitiendo que al final de la última iteración se logre un producto con todas las restricciones y características deseadas por el cliente. A continuación se detallan cada una de las tareas de la ingeniería por iteraciones.

Tareas de la Ingeniería

Para llevar a cabo las tareas de la ingeniería se cuenta con una tabla la cual permite definir cada una de las actividades en las que estarán asociadas cada historia de usuario. A continuación se detallan las tareas asignadas por cada una de las iteraciones.

3.2.1. Primera Iteración.

En esta iteración se implementa la HU número 1 con el objetivo de lograr la reducción de un grafo.

Historias de Usuarios	Estimación	Real
Reducir grafo	3 semanas	3 semanas

Tabla 25: HU abordadas en la primera iteración.

A continuación se muestra mediante tablas las tareas de ingeniería en las que la HU mencionada anteriormente fue desglosada, para lograr un mejor funcionamiento del sistema.

Tareas de la Ingeniería	
No. De la tarea: 1	**No. De la HU:** 1
Nombre de la tarea: Crear aristas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Fecha inicio: 01/12/2010	**Fecha fin:** 14/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en seleccionar una arista representativa entre cada nodo del grafo reducido y la adiciona a dicho grafo.	

Tabla 26: Tarea de Ingeniería #1 de la HU #1.

Tareas de la Ingeniería	
No. De la tarea: 2	**No. De la HU:** 1
Nombre de la tarea: Obtener Nodos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Fecha inicio: 16/01/2011	**Fecha fin:** 18/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en obtener un nodo del grafo original.	

Tabla 27: Tarea de la Ingeniería #2 de la HU #1.

Tareas de la Ingeniería

No. De la tarea: 3	**No. De la HU:** 1
Nombre de la tarea: Obtener Aristas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Fecha inicio: 18/01/2011	**Fecha fin:** 20/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en obtener una arista del grafo original.	

Tabla 28: Tarea de la Ingeniería #2 de la HU #1.

Tareas de la Ingeniería	
No. De la tarea: 4	**No. De la HU:** 1
Nombre de la tarea: Crear Partición	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Fecha inicio: 21/01/2011	**Fecha fin:** 25/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en crear una partición dado una determinada propiedad.	

Tabla 29: Tarea de la Ingeniería #2 de la HU #1.

3.2.2. Segunda Iteración.

En esta iteración se implementaron las Historias de usuarios 2,3 y 4:

Historias de Usuarios	Estimación	Real
Gestionar fichero de regla	1 semanas	1 semanas
Gestionar fichero de partición	1 semanas	1 semanas
Crear información de empotrado	1 semanas	1 semanas
Total	3 semanas	3 semanas

Tabla 30: HU abordadas en la segunda iteración.

A continuación se muestran las tareas de ingeniería en que las HU han sido separadas para lograr un eficiente y rápido funcionamiento del sistema.

Tareas de la Ingeniería	
No. De la tarea: 1	**No. De la HU:** 2

Nombre de la tarea: Guardar regla	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 26/01/2011	**Fecha fin:** 27/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en guardar en un fichero la información de una regla de reescritura.	

Tabla 31: Tarea de la Ingeniería #1 de la HU # 2

Tareas de la Ingeniería	
No. De la tarea: 2	**No. De la HU:** 2
Nombre de la tarea: Cargar regla	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 28/01/2011	**Fecha fin:** 31/01/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en cargar la información de una regla de reescritura que se encuentra en un fichero, para poder obtener el grafo original a partir del grafo reducido.	

Tabla 32: Tarea de la Ingeniería #2 de la HU #2

Tareas de la Ingeniería	
No. De la tarea: 3	**No. De la HU:** 3
Nombre de la tarea: Guardar partición	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 01/02/2011	**Fecha fin:** 02/02/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en guardar en un fichero la información de una partición.	

Tabla 33: Tarea de la Ingeniería #3 de la HU #3

Tareas de la Ingeniería	
No. De la tarea: 4	**No. De la HU:** 3
Nombre de la tarea: Cargar partición	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 03/02/2011	**Fecha fin:** 05/02/2011

Programador responsable: Yasnary González Pérez
Descripción: Esta tarea consiste en cargar la información de una partición que se encuentra en un fichero, para poder aplicar el algoritmo de reducción de grafos.

Tabla 34: Tarea de la Ingeniería #4 de la HU #3.

Tareas de la Ingeniería	
No. De la tarea: 5	**No. De la HU:** 4
Nombre de la tarea: Crear Información de empotrado	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 06/02/2011	**Fecha fin:** 10/02/2011
Programador responsable: Yasnary González Pérez	
Descripción: Esta tarea consiste en crear la información de empotrado.	

Tabla 35: Tarea de la Ingeniería #5 de la HU #4.

3.2.3. Tercera Iteración.

En esta iteración se implementa la Historia de Usuario perteneciente al número 5, con el objetivo de lograr realizar búsquedas de caminos mínimos:

Historias de Usuarios	Estimación	Real
Búsqueda de caminos mínimos	2 semanas	2 semanas
Brindar servicio de búsqueda de caminos mínimos.	1 semana	1 semana

Tabla 36: HU abordadas en la tercera iteración.

Tareas de la Ingeniería	
No. De la tarea: 1	**No. De la HU:** 5
Nombre de la tarea: Búsqueda de caminos mínimos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 15/02/2011	**Fecha fin:** 29/05/2011
Programador responsable: Jose Angel Inda Herrera	
Descripción: Esta tarea consiste en implementar la búsqueda de caminos mínimos después de aplicar el algoritmo de reducción de grafo. Para ello es necesario implementar una función fd la es necesaria para obtener las distancias entre los nodos.	

Tabla 37: Tarea de la Ingeniería #1 de la HU #5

Tareas de la Ingeniería	
No. De la tarea: 2	**No. De la HU:** 6
Nombre de la tarea: Brindar servicio de búsqueda de caminos mínimos.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 29/05/2011	**Fecha fin:** 02/06/2011
Programador responsable: Jose Angel Inda Herrera	
Descripción: Esta tarea consiste en implementar un servicio web que le permite al sistema la integración con los SIG.	

Tabla 38: Tarea de la Ingeniería #2 de la HU #6

3.2.4. Cuarta Iteración.

Para esta última iteración se implementan las Historias de usuarios correspondientes a los números 6,7 y 8. Con el objetivo de que el sistema sea capaz de cargar el grafo de la Base de Datos Neo4J., genere el grafo. A continuación se muestran mediante una tabla las HU correspondientes a esta última iteración.

Historias de Usuarios	Estimación	Real
Generar grafo desde cartografía	1 semanas	1 semanas
Guardar grafo en fichero .dot.	1 semanas	1 semanas
Cargar grafo a base de datos Neo4J.	1 semanas	1 semanas
Total	3 semanas	3 semanas

Tabla 39: HU abordadas en la cuarta iteración

A continuación se muestran las tareas de ingeniería en las que fueron desplegadas las HU:

Tareas de la Ingeniería	
No. De la tarea: 1	**No. De la HU:** 7
Nombre de la tarea: Generar grafo desde cartografía	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio:01/03/2011	**Fecha fin:** 05/03/2011

Programador responsable: Jose Angel Inda Herrera
Descripción: Esta tarea consiste en obtener el grafo que representa la capa de viales de una cartografía.

Tabla 40: Tarea de la Ingeniería #1 de la HU #7

Tareas de la Ingeniería	
No. De la tarea: 2	**No. De la HU:** 8
Nombre de la tarea: Guardar grafo en fichero .dot.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 06/03/2011	**Fecha fin:** 10/03/2011
Programador responsable: Jose Angel Inda Herrera	
Descripción: Esta tarea consiste en guardar en un fichero .dot el grafo que se obtiene al generar el grafo que representa la capa de viales de la cartografía.	

Tabla 41: Tarea de la Ingeniería #2 de la HU #8

Tareas de la Ingeniería	
No. De la tarea: 3	**No. De la HU:** 9
Nombre de la tarea: Cargar grafo a Base de Datos Neo4J.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 12/03/2011	**Fecha fin:** 25/03/2011
Programador responsable: Jose Angel Inda Herrera	
Descripción: Esta tarea consiste en cargar un grafo que se encuentra en un fichero con formato .dot para el motor de persistencia para almacenamiento de grafos Neo4j.	

Tabla 42: Tarea de la Ingeniería #3 de la HU #9

3.3. Pruebas.

Una de las características principales de XP, es su fuerte énfasis en las pruebas, comprobando el funcionamiento del sistema. Las pruebas son un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente. Por esta razón se debe definir en el proceso de la ingeniería del software **(43)**. Las pruebas contribuyen a elevar la calidad de los productos desarrollados y a la seguridad de los programadores a la hora de introducir cambios o modificaciones.

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática y las pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente **(44)**.

3.3.1. Pruebas unitarias.

Las pruebas unitarias según la metodología XP, se utilizan para verificar el funcionamiento del sistema; ya que al ser la programación en dúo, uno de los programadores va programando mientras que el otro va buscando errores e ideando posibles mejoras. Una prueba unitarias es la verificación de un módulo (unidad de código) determinado dentro de un sistema. Son llevadas a cabo por los programadores de cada módulo. Las pruebas unitarias poseen gran beneficio ya que mediante ellas, se puede asegurar que un determinado módulo cumpla con un comportamiento esperado.

Las pruebas unitarias se consideran una actividad fundamental de XP. Brindan una visión de lo que se quiere realizar. Además demuestra que lo implementado es lo que se pensaba desde el principio. **(45)**

A continuación se exponen algunas utilidades de este tipo de pruebas:

- Ayudan a producir un código de mayor calidad.
- Detección rápida de errores cuando se programan nuevas funcionalidades o se realizan cambios en el código.
- Sirven como pequeña fuente de documentación sobre qué es lo que se espera que haga el código.
- Realizar pruebas unitarias obliga al programador a escribir el código en pequeñas porciones.

Prueba unitaria reducción de grafo

Entrada: Grafo almacenado en el motor de persistencia Neo4j (grafo Tabla 43).

Se sabe que la cantidad de nodos del grafo reducido resultante de la prueba es 4, por lo que al invocar la función *cantidadNodos()* del grafo reducido se obtiene la cantidad de 4 nodos. También se invoca la función *cantidadAristas()* en el grafo reducido se obtienen 8 aristas y al comparar con la cantidad de aristas esperadas da la misma cantidad.

3.3.2. Pruebas de aceptación.

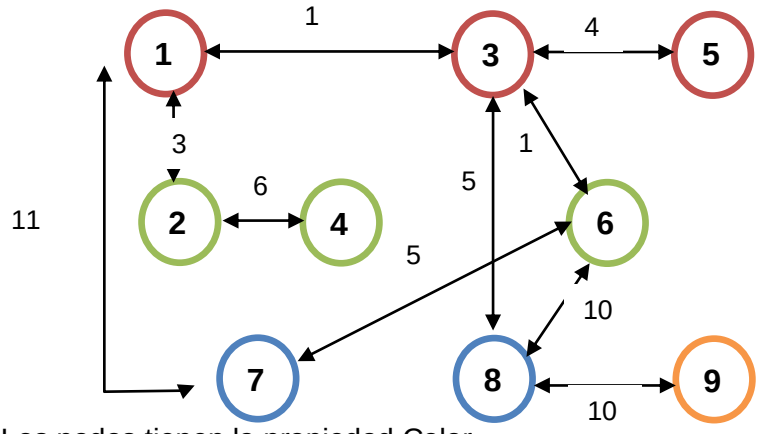
Las pruebas de aceptación no son más que validaciones que se le realiza al sistema para ver si cumple con el funcionamiento esperado. Estas pruebas son definidas por el usuario del sistema y preparadas por el equipo de trabajo. Tienen una importancia crítica para el éxito de una iteración.

Las pruebas de aceptación son pruebas de caja negra que se realizan partiendo de las Historias de usuario **(44)**, estas pueden tener varias pruebas hasta que quede asegurada la aceptación del sistema. Los clientes son los responsables de que los resultados de estas pruebas sean correctos. En caso de que fallen varias pruebas deben de indicar el orden en que las mismas han de ser resueltas. Una Historia de usuario no puede indicarse resuelta hasta que la misma no pase correctamente por las pruebas de aceptación.

El objetivo específico de esta prueba es garantizar que los requerimientos han sido cumplidos y que el sistema ha sido aceptable. **(46)**

A continuación se muestran las pruebas de aceptación para cada Historia de usuario y definidas en iteraciones, para lograr una mayor organización:

3.3.2.1. Iteración 1

Caso de Prueba de Aceptación	
Código: HU1_P1	**HU:** 1
Nombre de la prueba: Comprobar reducción de grafo	
Descripción: Se realizó una prueba con un grafo como el que se muestra a continuación  Los nodos tienen la propiedad Color Rojo (1,3,5) Verde (2,4,6)	

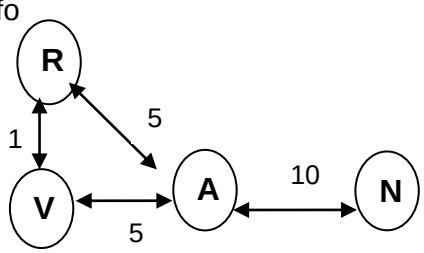
Azul (7,8) Naranja (9)
Condiciones de ejecución: Se debe tener almacenado el grafo en la BDOG Neo4J.
Entradas/ Pasos de ejecución: Al aplicar una reducción por color al grafo entrado se obtuvo el siguiente grafo 
Resultados esperados: Se muestra el grafo reducido por color sin haber generado ningún error.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 43: Prueba del módulo de reducción de grafo y búsqueda de caminos mínimos

Caso de Prueba de Aceptación
Nombre de la prueba: Comprobar obtención del grafo
Descripción: Se realizó una prueba con la red de viales de una cartografía la cual se puede observar en el Anexo 10, permitió además realizar una comparación con el sistema propuesto y el QGIS.
Condiciones de ejecución: Se deben haber almacenada la cartografía en el SGBD PostgreSQL.
Entradas/ Pasos de ejecución: Al realizar una comparación entre el sistema propuesto y el QGIS se obtuvo lo siguiente:

	cantNodos	cantAristas	cantNodosRepetidos
QGIS	1457	0	1189
Sistema	268	360	0

Resultados esperados:

Se obtuvo el grafo sin haber generado ningún error.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 44: Prueba del módulo de obtención

Caso de Prueba de Aceptación	
Código: HU3_P2	**HU:** 3
Nombre de la prueba: Guardar fichero de partición.	
Descripción: Evaluar la funcionalidad guardar fichero de partición.	
Condiciones de ejecución: Se debe tener almacenado el grafo en la BDOG Neo4J.	
Entradas/ Pasos de ejecución: Comprobar que para cada una de las particiones del grafo se cree un fichero correspondiente.	
Resultados esperados: Se guarda el fichero de partición sin haber generado ningún error.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 45: Prueba de Aceptación HU3_P2

Caso de Prueba de Aceptación	
Código: HU4_P1	**HU:** 4
Nombre de la prueba: Crear información de empotrado	
Descripción: Evaluar la funcionalidad crear información de empotrado.	
Condiciones de ejecución: Se debe tener las particiones del grafo creadas.	
Entradas/ Pasos de ejecución: Comprobar la generación de la información de empotrado para cada uno de los ficheros.	
Resultados esperados: Se crea la información de empotrado sin haber generado ningún error.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 46: Prueba de Aceptación HU4_P1

Caso de Prueba de Aceptación	
Código: HU5_P1	**HU:** 5
Nombre de la prueba: Búsqueda de caminos mínimos.	
Descripción: Evaluar la funcionalidad búsqueda de caminos mínimos.	
Condiciones de ejecución: Se debe haber aplicado la funcionalidad de reducción de grafo.	
Entradas/ Pasos de ejecución: Se debe entrar dos vértices de entrada, para buscar luego el camino mínimo entre los dos vértices entrados.	
Resultados esperados: Se muestran los puntos que conforman el camino mínimo sin haber generado ningún error.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 47: Prueba de Aceptación HU5_P1

Caso de Prueba de Aceptación	
Código: HU6_P1	**HU:** 6
Nombre de la prueba: Generar grafo desde cartografía.	
Descripción: Evaluar la funcionalidad generar grafo desde cartografía.	
Condiciones de ejecución: Solo se convertirá la cartografía a grafo si la misma se encuentra almacenada en la base de datos PostgreSQL	
Entradas/ Pasos de ejecución: Comprobar la generación del fichero con el grafo almacenado.	
Resultados esperados: Se genera la cartografía en un modelo matemático en este caso un grafo.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 48: Prueba de Aceptación HU6_P1

Caso de Prueba de Aceptación	
Código: HU8_P1	**HU:** 8

Nombre de la prueba: Cargar grafo a base de datos Neo4J
Descripción: Evaluar la funcionalidad cargar grafo a base de datos Neo4J
Condiciones de ejecución: El grafo debe estar almacenado en fichero .dot
Entradas/ Pasos de ejecución: Comprobar que el grafo que se encuentra almacenado en la BD Neo4J tiene la misma cantidad de aristas y de nodos que el que se encuentra en la BD PostgreSQL.
Resultados esperados: Se carga el grafo a BD Neo4J sin haber generado ningún error.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 49: Prueba de Aceptación HU8_P1

3.4 Experimentos

Se realizaron experimentos sobre los grafos G_1 (Anexo 11) y G_2 (Anexo 12) y se arrojó como resultado lo siguiente:

Experimento sobre el grafo G_1:

Al grafo G_1, el cual tiene 270 nodos, se le aplica una reducción por una determinada relación de equivalencia. Luego de esto se obtiene el grafo reducido GR_1 con 21 nodos. Al buscar el camino mínimo entre los nodos n1 (nodo con el id # 135) y n2 (nodo con el id # 166) sobre el grafo original G_1 utilizando para ello la implementación del algoritmo Dijkstra, se obtiene lo siguiente:

- Cantidad de nodos visitados: 262.
- Tiempo de ejecución del algoritmo Dijkstra: 553 milisegundos.
- Costo total del camino: 38220.09714 u.

Al realizar la misma búsqueda de camino mínimo sobre el grafo reducido GR_1 utilizando la implementación del mismo algoritmo Dijkstra al cual se le realizó una transformación para utilizar la función ***Fx*** que calcula el costo de pasar por un nodo reducido, se obtiene:

- Cantidad de nodos visitados: 20.
- Tiempo de ejecución del algoritmo Dijkstra: 552 milisegundos.
- Tiempo de la expansión[1]: 411 milisegundos.
- Costo total del camino: 38220.09714 u.

[1] Es el tiempo que tarda en ejecutarse una expansión de un nodo que lo necesite para la búsqueda de camino mínimo sobre le grafo reducido. El tiempo de expansión + el tiempo de ejecución del algoritmo Dijkstra da como resultado el tiempo total de la búsqueda.

Experimento sobre el grafo G_2:

Al grafo ***G_2***, el cual tiene 35 nodos, se le aplica una reducción por una determinada relación de equivalencia. Luego de esto se obtiene el grafo reducido ***GR_2*** con 27 nodos. Al buscar el camino mínimo entre los nodos n1 (nodo con el nombre n3) y n2 (nodo con el nombre n34) sobre el grafo original ***G_2*** utilizando para ello la implementación del algoritmo Dijkstra, se obtiene lo siguiente:

- Cantidad de nodos visitados: 35.
- Tiempo de ejecución del algoritmo Dijkstra: 63 milisegundos.
- Costo total del camino: 9.0 u.

Al realizar la misma búsqueda de camino mínimo sobre el grafo reducido ***GR_2*** utilizando la implementación del mismo algoritmo Dijkstra al cual se le realizó una transformación para utilizar la función ***Fx*** que calcula el costo de pasar por un nodo reducido, se obtiene:

- Cantidad de nodos visitados: 26.
- Tiempo de ejecución del algoritmo Dijkstra: 299 milisegundos.
- Tiempo de la expansión: 43 milisegundos.
- Costo total del camino: 9.0 u.

Es importante tener en cuenta que en el tiempo de la búsqueda de camino mínimo sobre el grafo reducido está incluido el tiempo de acceder a los ficheros de las reglas de reescritura y el fichero que tiene los valores de la función ***Fx***. Este tiempo de acceso es considerable si se tiene en cuenta que en los ficheros de las reglas está la información de todos los nodos del grafo original y en el fichero de la función ***Fx*** está el costo de pasar por cada uno de los nodos reducidos. En ambos casos hay que realizar una búsqueda sobre estos ficheros, es decir, en el peor de los casos en que se esté localizando un nodo determinado, hay que recorrer el fichero de la regla completo o si se busca el costo de pasar por un nodo reducido determinado hay que recorrer el fichero de la función ***Fx*** completo.

3.5 Conclusiones

En este capítulo se elaboran las Tarjetas C.R.C. Se especifican las tareas necesarias para darle cumplimiento a la descripción de las HU. Se plasmaron las pruebas unitarias y se precisaron las pruebas de aceptación que le brindaran al cliente, conformidad y seguridad ante las funcionalidades del sistema. De esta manera se concluye que gracias a las pruebas realizadas el análisis de redes que se implementa en el sistema propuesto es más eficiente que el que brinda el SIG QGIS. Además de que QGIS devuelve información repetida. También con los experimentos se

comprueba la eficiencia de realizar una búsqueda de camino mínimo sobre un grafo reducido.

Conclusiones

En el presente trabajo se desarrolló un sistema para el análisis de redes en los SIG, el sistema está compuesto por dos módulos, uno para la obtención del grafo que representa una red de viales y el otro para la reducción de grafo y búsqueda de caminos mínimos sobre el grafo reducido de forma eficiente.

Finalmente se arribó a las siguientes conclusiones:

1 El sistema se desarrolló haciendo uso de herramientas libres, el mismo contribuye a la soberanía tecnológica.

2 Al aplicar una reducción a un grafo para realizar búsquedas de caminos mínimos sobre el grafo reducido se obtiene una respuesta eficiente cuando las redes son grandes.

3 Al obtener un grafo reducido se puede realizar análisis de redes a diferentes escalas del mapa.

4 El módulo de obtención de grafo obtiene los nodos y las aristas del grafo que representa una red de formas más eficiente que el SIG de escritorio QGIS.

Recomendaciones.

Una vez concluido el desarrollo de este trabajo se recomienda para la mejora del sistema:

- Es necesario desarrollar un sistema que cuente con almacenamiento persistente eficiente para la función ***Fx*** de forma tal que el tiempo de búsqueda de un valor en esta función sea el mínimo.
- Es necesario desarrollar un sistema para almacenamiento persistente de grafo ya que los sistemas o bases de datos para grafos actuales, carecen de soporte teórico o científico para el manejo de estas estructuras. Esto trae como consecuencia que estos sistemas no soporten algunos de los requisitos que se deben cumplir al modelar una estructura como esta, en este caso un grafo, por ejemplo, el neo4j no permite relaciones entre un mismo nodo y en un grafo pueden existir lazos.

En el módulo de obtención de grafo es necesario:

- Disminuir la complejidad del algoritmo presentado utilizando diversas técnicas de geometría computacional.
- Definir un sistema de actualización del grafo que representa una cartografía de forma tal que cuando se modifique la cartografía no sea necesario general el grafo completamente, sino solamente actualizarlo.

Referencias Bibliográficas.

1. Bosque Sendra, J. *Sistemas de Información Geográfica.* Madrid : Ediciones Rialp, S.A., 1992.

2. *Servicio de mapas temáticos.* Rodríguez Torres, A. Rodríguez Puente. s.l. : Revista internacional de ciencias de la tierra, (139):36–39. (139):36–39.

3. Postgresql. [En línea] 2011. http://www.postgresql.org/..

4. PostGIS. [En línea] http://postgis.refractions.net..

5. project, #1_pgRounting. [En línea] http://www.postlbs.org/.

6. *Idelabroute: Librería para la.* F. Campos, J.P. de Castro, and R. García. s.l. : In IV Jornadas SIG Libre, 2010.

7. Rodríguez, N. Sanabria., R. D. M. LOS SISTEMAS DE INFORMACION GEOGRAFICA. [En línea] 1998. http://www.humboldt.org.co/humboldt/mostrarpagina.php?codpage=70001#top.

8. Burrough, P,. *Principles of Geographical Information Systems for Land Resources Assessment.* Oxford : Oxford University Press, 1998.

9. Calvo Melero, M. *Sistemas de Información Geográfica Digitales.* Vitoria : Instituto Vasco de Administración Pública, 1993. ISBN 84-7777-101-4.

10. Diestel, Reinhard. *Graph Theory.* s.l. : Springer-Verlag, 1997.

11. Garrido, Luis Garcia. Comprobar las posibilidades que el Método de Grafos. [En línea] 1990. www.ucm.es/BUCM/tesis/edu/ucm-t26700.pdf.

12. *Tutorial de PostgreSQL.* PostgreSQL, El equipo de desarrollo de. s.l. : Thomas Lockhart.

13. project, #2_pgRouting. [En línea] http://pgrouting.postlbs.org/wiki/LoadingtheCode5.

14. project, #3_pgRouting. [En línea] http://pgrouting.postlbs.org/discussion/topic/353.

15. Campos Gutiérrez, F, De Castro Fernández, J.P. y García Martín, R. *IDELabRoute: Librería para la gestión de grafos escalable.* Instituto Geográfico Nacional y la Universidad de Valladolid : s.n., 2009.

16. *GUÍA PARA EL SOPORTE DE SISTEMAS DE INFORMACIÓN GEOGRÁFICA EN SERVIDORES POSTGRESQL PARA UBUNTU.* Ing. Yisel Barrabia Legrá, Ing. Yudisney Vazquez Ortíz, Ing. Aliuska Sánchez Ibarria, Lisleydi Mier Pierre, Jorge Luis Rodríguez Armenteros. Ciudad de la Habana : s.n., 2010.

17. Becchi, Lorenzo. *Entrevista LorenzoBecchi.* 2009.

18. Módulo de análisis de redes.gvSIG. [En línea] http://www.gvsig.org/web/docdev/docs/desarrollo/plugins/redes/components/core/descripcion/.

19. ArcGis. [En línea] http://www.esri.com/software/arcgis/index.html.

20. ESRI. [En línea] http://www.esri.com/.

21. Institute, Environmental Systems Research. ArcView Spatial Analyst. [En línea] 2004. http://www.rockware.com/product/featuresLobby.php?id=200&category=476.

22. Equipo de desarrollo. ESRI Shapefile Technical Description. [En línea] julio de 1998. http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf..

23. GRASS GIS. *GRASS Development Team.* [En línea] ftp://ftp.rz.uni-wuerzburg.de/pub/other/grass/dglib/index.html.

24. *IDELabRoute: Librería para la gestión de grafos escalable.* Campos Gutiérrez F., Castro Fernández J. P., García Martín R. Ciudad de la Habana : IV Jornadas SIG Libre, 2010.

25. Campos Gutiérrez, F, De Castro Fernández, J.P. y García Martín, R. *IDELabRoute: Librería para la gestión de grafos escalable.* s.l. : Instituto Geográfico Nacional y la Universidad de Valladolid, 2009.

26. Javier García de Jalón, J.I.R., Iñigo Mingo, Jesús Calleja. *Aprenda Java como si estuviera en primero.* San Sebastian : s.n., 2000.

27. —. *Aprenda Java como si estuviera en primero.* 2000.

28. Error500.net. Sistema Gestor de base de datos SGBD. [En línea] 2004. http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php..

29. Neo4j and the Benefits of data locality. [En línea] 2010. http://neo4j.org/.

30. Pescos, Daniel. *PostgreSQL vs. MySQL.* 2009.

31. Development Group, The PostgreSQL Global. *PostgreSQL 8.1.0 Documentation.* 2005.

32. Toby Segaran, Jeff Hammerbache. *The Stories Behind Elegant Data Solutions.*

33. Pressman, R.S. Ingeniería del software. [En línea] 2005. [Citado el: 10 de Enero de 2011.]

34. Beck, K. *Extreme Programming Explained. Embrace Change.* s.l. : Pearson Education, 1999.

35. Beck, F.M.Y.K. *Planeando en Programación Extrema.* 2000.

36. Joskowicz, José. Reglas y Prácticas en eXtreme Programming. [En línea] 2002. http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf..

37. Openstreetmap project. [En línea] 2011. http://wiki.openstreetmap.org/wiki/Main_Page.

38. Gary E. Sherman Tim Sutton. Quantum GIS. [En línea] 2011. http://www.qgis.org..

39. *Aplicación de las gramáticas de grafo en Sistemas de Información Geográfica.* Puente, Rafael Rodríguez. Ciudad de La Habana : s.n.

40. Escribano, Gerardo Fernández. *Introducción a Extreme Programming.* 2002.

41. Planning and Running an XP Iteration. [En línea] 2005. http://martinfowler.com/articles/planningXpIteration.html ..

42. C., Benjamín González. WSDL para la documentación de Servicios Web. [En línea] 2004. . http://www.desarrolloweb.com/articulos/1581.php..

43. Carmen. Metodologías ágiles para el desarrollo de software. [En línea] 2008. [Citado el: 16 de Febrero de 2011.] http://www.willydev.net/descargas/masyxp.pdf..

44. Pressman, R. S. *Ingeniería del software.* Ciudad Habana : s.n., 2005.

45. Allende, R. *Desarrollo de Portales y Extranet con Plone,.* 2006.

46. Peñalver, Gladys Marsi. *Metodología Ágil para proyectos de software libre.* 2008.

47. Beck, Kent. *Extreme Programming.* 2000.

48. Heng Tao Shen, Jian Pei,M. Tamer Ăzsu. *Web-age Information Management.* China : s.n., Julio 2010.

49. Universidad, Morón. *Metodologías Ágiles.* s.l. : Morón Universidad.

50. Nodarse, Yoandry Lazo. *Que se utilizan para el analisis de redes.* Abril de 2010.

51. *"ENFOQUE DE HIPERGRAFO EN SISTEMAS DE INFORMACIÓN GEOGRÁFICA PARA LA MODELACIÓN DE REDES,".* Puente, R. Rodríguez. VII Congreso Internacional Geomática : s.n., 2011.

52. ESRI Shapefile Technical Description. [En línea] julio de 1998. http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf..

53. Sitio Oficial de Neo4j. *Sitio Oficial de Neo4j.* [En línea] 2011. http://neo4j.org/.