

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Neo4Qgis: Conexión del software QGIS y una base de datos geográfica orientada a grafos

Cristhian Fabian Forero
David Esteban Moreno
Wilmar Fernando Pineda

Universidad Distrital Francisco José de Caldas
Especialización en Sistemas de Información Geográfica (SIG)
Bogotá D.C, Colombia
2016

Neo4Qgis: Conexión del software QGIS y una base de datos geográfica orientada a grafos

Cristhian Fabian Forero
David Esteban Moreno
Wilmar Fernando Pineda

Proyecto de grado en modalidad monografía presentado como requisito parcial para optar título de:
Especialistas en Sistemas de Información Geográfica

Director (a):
Salomón Ramírez

Línea de trabajo:
Ingeniería de software

Universidad Distrital Francisco José de Caldas
Ingeniería Catastral y Geodesia
Bogotá D.C, Colombia
2016

Agradecimientos

A nuestras familias…

Resumen

El propósito de este estudio se centra en análisis, diseño e implementación de una herramienta para la gestión de información geográfica en bases de datos orientadas a grafos desde un cliente SIG, la metodología de desarrollo se basó en OpenUP, se realizó la identificación de los requerimientos funcionales y no funcionales que satisficieran las necesidades de información geográfica en bases de datos orientadas a grafos desde el cliente SIG, así mismo se definió y estableció la arquitectura que permitió satisfacer los requerimientos funcionales y no funcionales y se realizó la implementación de la herramienta que permitió la conexión y gestión de información geográfica por medio de un plugin en el software QGIS con la base de datos de Neo4j. Finalmente la evaluación del plugin se desarrolló teniendo en cuenta la norma ISO-9126, donde los resultados fueron satisfactorios en un 90 por ciento.

Palabras clave: Base de datos, grafos, plugin, QGIS, SIG, Neo4j.

Abstract

This study is focused on analysis, design and implementation of a tool for managing geographic information in graphs oriented databases from a GIS client. The software development methodology was based on OpenUP, it was made the identification of functional and nonfunctional requirements that met the needs of geographic information in graphs oriented databases from the GIS client, also it was defined and established the architecture that allowed meet the functional and nonfunctional requirements and it was made the implementation of the tool which allowed to perform the connection with Neo4j database and the management of geographic information through a plugin in QGIS software. Finally, the plugin evaluation was developed taking into account the ISO-9126 standard, where the results were satisfactory by 90 percent.

Keywords: Database, graphs, plugin, QGIS, GIS, Neo4j.

Contenido

Pág.

Resumen VII

Contenido VIII

Lista de figuras IX

Lista de tablas X

1. Introducción 11

2. Planteamiento del problema 12

3. Objetivos 14

4. Justificación 15

5. Alcance 16

6. Metodología 17

7. Resultados 21

7.1 Requisitos 21
7.2 Arquitectura 23
7.2.1 Modelo de dominio 23
7.2.2 Modelo de Interacción 24
7.2.3 Diagrama de clases 24
7.2.4 Vista de alto nivel 25
7.2.5 Vista de Despliegue 26
7.2.6 Vista de implementación 27
7.3 Implementación 27
7.3.1 Creación de la conexión de Neo4J con Python 27
7.3.2 Creación de la clase dibujar 29
7.3.3 Visualización de la interfaz 29
7.4 Pruebas 30

Conclusiones y recomendaciones 33

Conclusiones 33
Recomendaciones 34

Lista de figuras

Pág.

Figura 6-1: Ciclo de Vida 17
Figura 7-1: Diagrama casos de uso. Fuente propia. 22
Figura 7-2: Modelo de dominio. Fuente propia. 23
Figura 7-3: Diagramas de secuencias. Fuente propia. 24
Figura 7-4: Diagrama de clases. Fuente propia. 25
Figura 7-5: Vista de alto nivel. Fuente propia. 26
Figura 7-6: Vista Despliegue. Fuente propia. 26
Figura 7-7: Vista de Implementación. Fuente propia. 27
Figura 7-8: Líneas de código usadas en la conexión. 28
Figura 7-9: Presentación de los puntos cargados en Neo4j. Fuente propia. 28
Figura 7-10: Líneas de código usadas en la conexión. 29
Figura 7-11: Visualización de la interfaz gráfica. Fuente propia 30
Figura 7-12: Prueba de posibles errores. Fuente propia 32

Lista de tablas

Pág.

Tabla 6-1: Metodología de evaluación..... 19

Tabla 7-1: Resultado de la evaluación por los usuarios..... 31

1. Introducción

Vivimos en un mundo conectado, no hay piezas aisladas de información, sólo una base de datos que abarque las relaciones como un aspecto fundamental de su modelo de datos, es capaz de almacenar conexiones de procesos y de consulta de manera eficiente (Hunger, Boyd, & Lyon, 2016).

Los requerimientos de usuarios han ido creciendo y especializándose, la información que se gestiona a través de los sistemas informáticos que soportan dichos requerimientos ha evolucionado, pasando de cúmulos de información con una estructura clara y definida, ya sea de tipo entidad - relación u objeto - relacional, a un cúmulos de información provenientes de diferentes fuentes de datos, con diferentes formatos y estructura, los cuales requieren ser almacenados usando estrategias de persistencia que no cumplen el esquema entidad - relación (Ramírez & Herrera, 2013).

Las bases de datos relacionales todavía tienen sus casos de uso perfecto. Sin embargo, para aquellos casos en que se necesita una solución diferente, esperamos apoyar y reconocer cuándo y cómo el uso de una base de datos orientada a grafos para hacer frente a los nuevos retos.

Una base de datos de grafos es un sistema de gestión de bases de datos que permite crear, leer, actualizar y eliminar (CRUD) trabajando en un modelo de datos gráficos. Las bases de datos de grafos se construyen generalmente para su uso con sistemas transaccionales (OLTP). En consecuencia, normalmente son optimizados para el rendimiento transaccional, y diseñado con la integridad de las transacciones y la disponibilidad operativa en mente. (Hunger, Boyd, & Lyon, 2016).

El mundo de hoy ya no es impulsado por los datos, es impulsado por las conexiones entre ellos, los grafos son la forma más eficiente e intuitiva de trabajar con datos, imitando la interconexión de las ideas en la mente humana (Neo4j, 2016).

2. Planteamiento del problema

Desde hace años estamos asistiendo al crecimiento exponencial del uso de las tecnologías de la información y la comunicación en todos los ámbitos de la sociedad. Estas tecnologías, en su mayor parte, han facilitado el desarrollo de la sociedad y dotado de herramientas para afrontar problemas que, hasta entonces, no tenían solución con los medios existentes en ese momento (Araque, 2012)

Dada la alta demanda requerida al momento de almacenar información espacial y la gran capacidad que se necesita en las aplicaciones informáticas cuando realizan consultas sobre ellas, se ha generado la búsqueda de nuevas herramientas tecnológicas que ayuden a tener una mejor gestión de los datos (Washington, 2014), en esta medida los requerimientos de usuarios han ido creciendo y especializándose, la información que se gestiona a través los sistemas informáticos que soportan dichos requerimientos ha evolucionado (Ramírez & Herrera, 2013), y concordando con Hunger et al. (2016) las necesidades del usuario y las aplicaciones de hoy en día están demandando más datos, agilidad y velocidad.

Las bases de datos orientadas a grafos constituyen una alternativa eficiente y adecuada en el manejo de la información, proporcionando un importante valor agregado en cuanto a rendimiento, agilidad, flexibilidad y escalabilidad del sistema (Neo4j, 2016) (Washington, 2014). Actualmente, el paradigma tradicional de gestión corresponde a las bases de datos relacionales (RDBMS), las cuales se encuentran altamente distribuidos e implementados en las organizaciones que requieren el manejo de su información, sin embargo, estos sistemas al requerir un elevado nivel de procesamiento para manejar las relaciones, ralentizan el tiempo de respuesta y demandan altos recursos computacionales (González & Cáceres, 2013; Vicknair et al., 2010; Ramírez & Herrera, 2013).

Por su parte, los sistemas de bases de datos orientados a grafos, manejan tanto las entidades como las relaciones en un solo conjunto de datos, permitiendo a los motores de consulta recorrer el sistema en por medio de grafos únicamente a través de las relaciones y entidades de interés, así como permitiendo la escalabilidad y crecimiento del sistema de una forma ordenada y flexible, gracias al fácil control de la estructura y la baja presencia de problemas relacionados con la migración de datos.

En este sentido los fenómenos espaciales pueden ser modelados de forma más eficaz y eficiente en las bases de datos orientadas a grafos que en los modelos relacionales (Ramírez & Herrera, 2013; Vicknair et al., 2010). Estas bases de datos orientadas a grafos ofrecen una velocidad de procesamiento, mejor capacidad de escalabilidad, entre otras características. Es por esto que es de interés para los investigadores y profesionales involucrados en el modelamiento espacial, contar con las herramientas que proporciona una base de datos orientada a grafos, y las posibilidades de visualización que ofrece un programa dedicado al manejo de los sistemas de información geográfica.

Por lo tanto, si fuera posible representar los datos como objetos y sus relaciones con otros objetos como un solo conjunto de datos, quizás sea posible generar patrones de conocimiento que permitan comprender dichos conjuntos de datos. Para tal efecto se consideran que las bases de datos orientadas a grafos son lo suficientemente flexibles y poderosas para representar estos conjuntos de datos geográficos (Taverner, 2011; Washington, 2014).

3. Objetivos

Objetivo general

- Análisis, diseño e implementación de una herramienta para la gestión de información geográfica en bases de datos orientadas a grafos desde un cliente SIG de escritorio.

Objetivos específicos

- Identificación de los requerimientos funcionales y no funcionales que satisfagan las necesidades de información geográfica en bases de datos orientadas a grafos desde un cliente SIG.

- Definir y establecer la arquitectura que permita cumplir los requerimientos funcionales y no funcionales.

- Implementar una herramienta que permita la gestión de información geográfica en bases de datos orientadas a grafos desde un cliente SIG.

4. Justificación

Gracias a los avances de la tecnología, cada día se puede tener información rápida y oportuna sobre cualquier área del conocimiento humano. De acuerdo a Martínez y Michel (2007), las bases de datos relacionales tienen una amplia aceptación y han demostrado su efectividad. Sin embargo, actualmente la manipulación de grandes volúmenes de datos geográficos resulta ser algo complejo, por lo cual se han implementado otras estrategias para manejar información geográfica.

Las bases de datos orientadas a grafos constituyen una alternativa eficiente y adecuada en el manejo de la información, permitiendo operar tanto las entidades como las relaciones en un solo conjunto de datos, existen casos de uso cuyos patrones de datos presentan una mejora de rendimiento en consultas cuando se implementan en un modelo orientado a grafos, y cuya latencia es mucho menor en comparación con el procesamiento por lotes de las funciones colectivas de otros tipos de base de datos (Robinson, Jim, & Eifrem, 2015). A parte de la ventaja de rendimiento, las bases de datos orientadas a grafos ofrecen un modelo de datos muy flexible que se adapta al constante cambio de requerimientos o entregas iterativas que concuerdan con las prácticas actuales de desarrollo de software ágil.

En concordancia con esto, la información espacial puede ser modelada de una forma más natural y coherente con el uso de bases de datos orientadas a grafos en contraste con los tradicionales sistemas de bases de datos relacionales. La forma original en que las BDOG fueron concebidas, se produjo en un contexto espacial (Taverner, 2011). Todo lo relativo a rutas, telecomunicaciones, logística, entre otros, puede apoyarse en aspectos que van desde las relaciones ponderadas y dirigidas simples a través de índices espaciales, hasta herramientas que permiten la representación de datos multidimensionales, utilizando estructuras de datos de control de árboles. Actualmente no existen herramientas de sistemas de información geográfica que contemplen la integración con motores de bases de datos orientados a grafos.

Es por esto que desarrollar una herramienta que permita la integración de motores de bases de datos orientados a grafos con software para el manejo de Sistemas de Información Geográfica, es un aporte de suma importancia, tanto en la implementación de los avances

actuales en cuanto a consulta y representación, así como en la investigación y desarrollo nuevos casos de uso y metodologías que permitan explorar y aprovechar los recursos que estas herramientas ofrecen.

5. Alcance

El presente proyecto es un paso inicial para el aprovechamiento de las bases de datos orientadas a grafos desde un ámbito espacial. En este sentido, el trabajo tiene como eje fundamental el análisis, diseño e implementación de una herramienta que permita la gestión de información geográfica contenida en bases de datos orientadas a grafos desde un cliente SIG de escritorio.

La herramienta debe permitir establecer una conexión entre una base de datos orientada a grafos y un software para el manejo de sistemas de información geográfica. Esta conexión debe realizar la importación de objetos geográficos con geometría tipo punto. También debe proveer los medios necesarios para visualizar dicha información de forma espacial valiéndose de las herramientas que el cliente SIG proporciona, esto permite a los usuarios de datos geográficos en general tener un vínculo con las bases de datos orientadas a grafos, así mismo los usuarios podrán encontrar soluciones óptimas y apropiadas de acuerdo a las necesidades nacientes en el mundo.

De esta forma se abre el camino para nuevos proyectos enfocados en el aprovechamiento del gran potencial espacial que las bases de datos orientadas a grafos ofrecen, pues su concepción original surgió bajo un ámbito geográfico, y estas representan la forma más natural en que los datos espaciales pueden ser manejados (Taverner, 2011).

Teniendo en cuenta lo dicho anteriormente se desarrolla bajo estos conceptos la idea del alcance para este proyecto, el cual expuesto de la forma más sencilla posible es la iniciar interacción a las bases de datos espaciales orientadas a grafos por medio de un sistema SIG de escritorio. Más específicamente el alcance primario del proyecto es establecer la conexión entre un software SIG de escritorio y un sistema gestor de bases de datos orientadas a grafos, después de establecer dicha conexión identificar los objetos espaciales contenidos en la base de datos orientada a grafos y posteriormente representarlos de forma gráfica en la herramienta SIG de escritorio.

6. Metodología

Con el fin de diseñar un aplicativo capaz de conectar una base de datos no relacional orientada a grafos con un software para el manejo de sistemas de información geográfica permitiendo la lectura y visualización de información contenida en dicha base de datos, se adoptó una metodología de desarrollo de software basada en OpenUP.

OpenUP es un proceso de desarrollo de software mínimo y suficiente, lo que significa que sólo el contenido fundamental se incluye. Sin embargo, OpenUP es completo en el sentido que puede ser manifiesto como todo un proceso para construir un sistema. Para hacer frente a las necesidades que no están cubiertas en su contenido, OpenUP es extensible para ser utilizado como base sobre la cual el contenido de proceso se puede añadir o adaptarse según sea necesario (Baduino, 2007).

Es posible distinguir dos dimensiones distintas pero complementarias: el método y el proceso. El primero hace referencia a la definición de los elementos del mismo (artefactos, tareas, roles y lineamientos), mientras que el proceso define la aplicación de estos elementos en un periodo de tiempo.

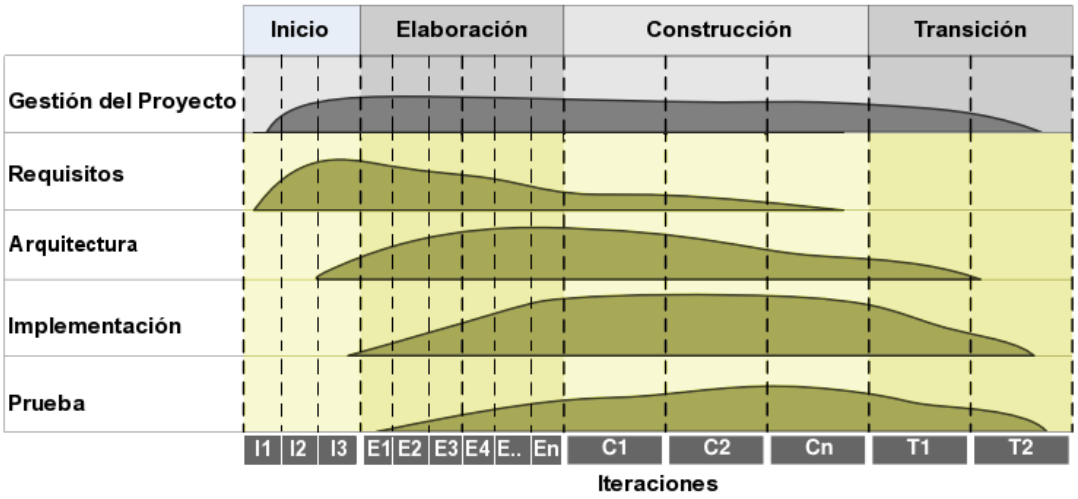


Figura 6-1: Ciclo de Vida, Tomado de OpenUP, (2016)

Esta metodología se centra en la arquitectura de forma temprana para reducir el riesgo y optimizar el desarrollo. De esta forma, en la fase de inicio fueron esbozadas las arquitecturas candidatas y se preparó un plan de proyecto. Esto se realizó con base en una investigación sobre los motores de bases de datos NoSQL y su implementación espacial, permitiendo así la identificación de oportunidades de mejora y estableciendo requerimientos que hicieran posible sacar provecho de las ventajas en el análisis espacial que las bases de datos orientadas a grafos ofrecen.

Posteriormente, en la fase de elaboración se definieron las reglas de negocio, los actores y se establecieron los requerimientos funcionales y no funcionales con base en los cuales fueron planteados los casos de uso. También se detalló en esta etapa la arquitectura estable de la herramienta, estableciendo los diagramas de contexto, de componentes, de despliegue y la vista lógica, en esta etapa fueron finalizados todos los elementos del método de requisitos.

Luego de esto, se realizó la fase de construcción, en la cual el sistema fue desarrollado con base en lo especificado en la fase de elaboración. Las características de la herramienta son implementadas en forma de iteraciones pequeñas y limitadas en el tiempo. Estas iteraciones estuvieron orientadas básicamente al establecimiento de la conexión entre la base de datos orientada a grafos y el software para manejo de sistemas de información geográfica, la definición de objetos espaciales en la base de datos y la invocación de estos elementos por parte del cliente, en esta etapa fueron finalizados todos los elementos del método de implementación.

Finalmente, en la fase de transición la herramienta fue liberada para los usuarios finales, lo cual incluyó un proceso de capacitación y permitió una retroalimentación por parte de estos. En esta fase se realizaron las pruebas finales teniendo en cuenta parámetros de calidad de la norma ISO/IEC 9126, con base en los objetivos del proyecto. El enfoque para la evaluación es cuantitativo, puesto que permite adquirir conocimientos fundamentales de cada una de las herramientas además de ver su funcionalidad de una manera imparcial, recogiendo y analizando los datos mediante conceptos y variables (González & Cáceres, 2013). Por medio de esta metodología se puede realizar la relación entre variables cuantificadas, lo cual posibilita hacer un análisis que permite conocer la evaluación del plugin realizado, en esta etapa fueron finalizados todos los elementos del método de prueba.

Para realizar la evaluación se escogieron los siguientes criterios:

Característica de Calidad Interna	Nombre de la Métrica	Pregunta	Tipo de Escala Métrica	Entrada para Medición
Funcionalidad	Cumplimiento de la funcionalidad	¿El plugin permite realizar la conexión a la base de datos orientada a grafos Neo4j?	Binario: Si o no	Si es el nivel óptimo
		¿El plugin permite visualizar los elementos geográficos almacenados en la base de datos Neo4j?	Binario: Si o no	Si es el nivel óptimo
	Idoneidad	¿El plugin contiene los elementos necesarios para definir la conexión?	Binario: Si o no	Si es el nivel óptimo
		¿El plugin contiene los elementos necesarios para visualizar los datos en QGIS?	Binario: Si o no	Si es el nivel óptimo
Usabilidad	Facilidad de aprendizaje	¿Qué tan oportuna es la retroalimentación por parte de la herramienta cuando se presentan errores al ingresar los parámetros para definir la conexión o visualización?	Escala ponderada 1 – 5	5 es el nivel óptimo
	Comprensión	¿Qué tan intuitivas son las funciones presentes en el plugin?	Escala ponderada 1 – 5	5 es el nivel óptimo
	Operabilidad	Califique la dificulta para llevar acabo cada una de las operaciones (conexión y visualización), teniendo en cuenta la cantidad de pasos para su ejecución.	Escala ponderada 1 – 5	5 es el nivel óptimo
	Atractividad	Que tan adecuada es la disposición y distribución de los elementos de la interfaz	Escala ponderada 1 – 5	5 es el nivel óptimo
Portabilidad	Facilidad de instalación	Califique el esfuerzo necesario para instalar el plugin.	Escala ponderada 1 – 5	5 es el nivel óptimo
		¿Qué tan pertinente es la documentación para llevar acabo la instalación?	Escala ponderada 1 – 5	5 es el nivel óptimo

Tabla 6-1: Metodología de evaluación.

Así mismo se tuvo en cuenta que la ISO/IEC 9126-1 (ISO/IEC, 2001), define un modelo de calidad como "El conjunto de características y las relaciones entre las mismas, que proveen la base para especificar requerimientos de calidad y evaluar calidad". En relación al modelo de calidad del producto software, el estándar ISO/IEC 9126, está dividido en cuatro partes: la primera ISO/IEC 9126-1: Presenta un modelo de calidad del software, estructurado en características y sub características, la segunda ISO/IEC TR 9126-2 proporciona métricas externas para medir los atributos de seis características de calidad externa definidas en y una explicación de cómo aplicar las métricas de calidad de software. La tercera ISO/IEC TR 9126-3: proporciona métricas internas para medir atributos de seis características de calidad interna definidas en ISO/IEC, (2001). Y finalmente ISO/IEC TR 9126-4: define métricas de calidad en uso para medir los atributos definidos en el estándar ISO9126.

7. Resultados

La ejecución de cada una de las etapas de la metodología, trajo consigo una serie de productos asociados al objetivo del presente trabajo. A continuación, se detallan estos resultados haciendo mención a la etapa en la cual fueron concebidos y su rol en la concepción general de la herramienta. Cabe resaltar que la metodología empleada consta de métodos y procesos, donde los métodos hacen referencia a la definición de los elementos que van a ser relacionados a continuación, mientras que los procesos se refieren a la aplicación de estos elementos en un periodo de tiempo.

7.1 Requisitos

Los requisitos fueron concebidos durante la etapa inicial y elaboración, teniendo como resultado la identificación de los actores, las reglas del negocio, los requerimientos funcionales y no funcionales y los casos de uso. En primera instancia, se identificó un único actor, el cual en adelante se denominará usuario. El usuario es todo aquel que interactúe con software especializado en manejo de sistemas de información geográfica con interés en la teoría de grafos aplicada en el ámbito espacial. El usuario debe contar con la capacidad de definir los parámetros para generar la conexión con la base de datos orientada a grafos.

Posterior a esto, fueron definidas las reglas de negocio, cuyo enfoque se orientó hacia las características fundamentales de las bases de datos NoSQL. Por una parte, estas bases de datos buscan preservar la integridad de los datos, lo cual sucede en las bases de datos transaccionales gracias a la atomicidad, consistencia, aislamiento y persistencia de estos (Garay, 1981), mientras que en las bases de datos NoSQL la ausencia de estas particularidades puede implicar conflictos relativos a la disponibilidad de los datos. Estos conflictos pueden ser explicados con base en las características de las bases de datos NoSQL conocidas como el teorema CAP (Browne, 2009), las cuales son consistencia, disponibilidad y tolerancia al fraccionamiento. La consistencia hace referencia a la capacidad que tienen todos los clientes de ver la misma versión de los datos incluso si estos están siendo modificados. La disponibilidad hace referencia a que todos los clientes pueden ver al menos una copia de los datos incluso si alguno de los servidores no se encuentra habilitado, y la tolerancia al fraccionamiento hace referencia a la capacidad de la

herramienta de conservar sus características incluso si este ha sido desplegado en diferentes servidores.

De esta forma, el teorema postula que solo dos de las tres características descritas anteriormente pueden ser satisfechas correctamente, por lo cual el presente trabajo tomó como base para la definición de las reglas del negocio la clasificación definida por (Tudorica & Bucur, 2011) referente a almacenamiento de gran volumen de datos (sin estructurar, estructurado, estructurado en pequeños paquetes), con lo cual se determinó que la importación de la base de datos debe ser mediante el ingreso del nombre de la misma por parte del usuario, además, el plugin debe permitir la lectura de la base de datos en un solo flujo de proceso, y finalmente, el plugin debe integrar la información de la base de datos con las herramientas de visualización del software manejador de sistemas de información geográfica.

Una vez definidas las reglas del negocio, se realizó un análisis de las necesidades del usuario enfocadas a la consecución del objetivo general, con lo cual se presentó la condición que debe estar presente en la herramienta para la satisfacción de estas necesidades. De esta forma, se estableció el siguiente requerimiento: Realizar un aplicativo que permita establecer una conexión entre un software para el manejo de sistemas de información geográfica y una base de datos orientada a grafos, además de permitir la lectura y visualización de la información espacial contenida en esta.

Con el fin de dar cumplimiento al requerimiento mencionado, fueron establecidas una serie de actividades, las cuales se detallan en la figura 7-1, detallando la interacción entre el usuario y los casos de uso, para posteriormente especificar el flujo de procesos de cada uno de estos.

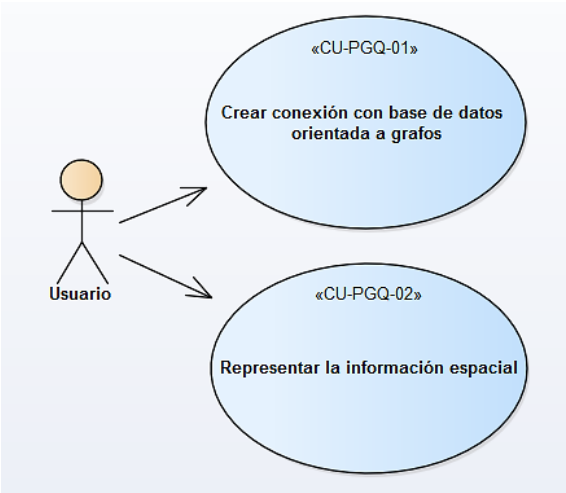


Figura 7-1: Diagrama casos de uso, Fuente propia.

El caso de uso CU-PGQ-01 permite a los usuarios establecer una conexión entre el software para el manejo de información geográfica y una base de datos espacial orientada

a grafos. La información necesaria para ejecutar esta tarea es la dirección IP o dominio, el puerto de escucha, el nombre de la base de datos y los datos de usuario y contraseña.

El evento es iniciado por el usuario, quien ingresa los parámetros requeridos en la herramienta, y este a su vez hace las validaciones necesarias, retroalimentado al usuario en las oportunidades que sea pertinente.

El caso de uso CU-PGQ-02 permite a los usuarios dibujar la entidad espacial que seleccione luego de establecer una conexión entre el software para el manejo de sistemas de información geográfica y una base de datos orientada a grafos. La información requerida corresponde a la lista de entidades geográficas en la base de datos la cual debe ser desplegada una vez el aplicativo ha permitido la conexión.

Luego de establecer el requerimiento funcional, los casos de uso y la interacción del usuario con estos, se especificaron los requerimientos no funcionales, los cuales están orientadas a describir las características de funcionamiento de la herramienta. El aplicativo debe estar hecha en Python, además debe contar con características de persistencia para sus transacciones. También debe ser multiplataforma, usable, amigable y eficiente, y finalmente, la herramienta debe ser software libre o en su defecto open source.

7.2 Arquitectura

La arquitectura de la herramienta fue implementada de forma preliminar en la fase inicial, teniendo la mayor parte de su desarrollo en la etapa de elaboración y refinando algunas características en la fase de construcción. La arquitectura cuenta con la definición del modelo de dominio, de interacción, y de clases, para seguir posteriormente con la vista de alto nivel, vista de despliegue y vista de implementación.

7.2.1 Modelo de dominio

El modelo de dominio presentado en la figura 7-2, permite representar el contexto del problema. Corresponde a un modelo conceptual donde son descritas las entidades con sus respectivos atributos, así como las relaciones entre estas.

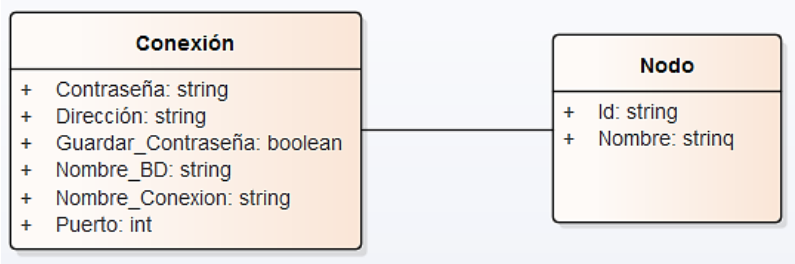
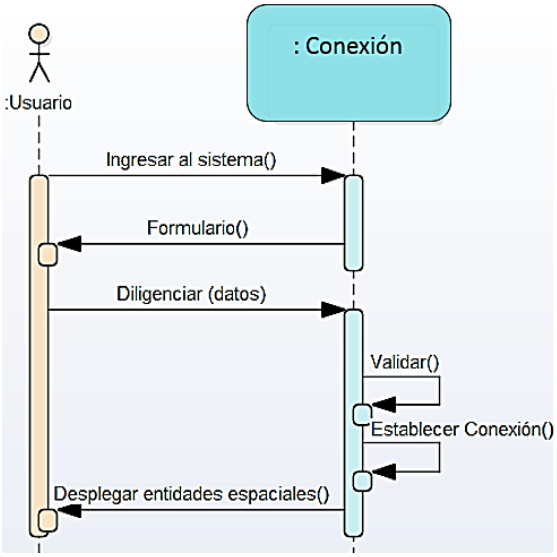


Figura 7-2: Modelo de dominio, Fuente propia.

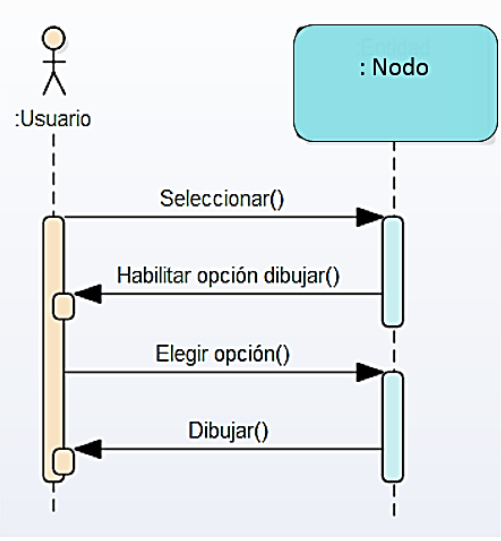
Se observa cómo la conexión está relacionada con la base de datos orientada a grafos que contiene la información espacial. Los atributos de la conexión constan de los parámetros necesarios para que esta pueda establecerse, mientras que los atributos para nodo corresponden básicamente a elementos de identificación y descripción.

7.2.2 Modelo de Interacción

En la figura 7-3 son presentados los diagramas de secuencias para cada uno de los casos de uso, los cuales detallan los objetos que participan en la interacción y la secuencia de mensajes que intercambian.



a) Interacción usuario conexión



b) Interacción usuario entidad

Figura 7-3: Diagramas de secuencias, Fuente propia.

7.2.3 Diagrama de clases

En la figura 7-4 es presentado el diagrama de clases, el cual describe tanto las clases de la herramienta y sus atributos, como las operaciones o métodos, y las relaciones entre los objetos. La clase Neo4QGIS representa la herramienta que permite la conexión entre el cliente SIG y la base de datos orientada a grafos. Cuenta entre sus atributos con los parámetros necesarios para establecer la conexión, y entre sus métodos con las acciones necesarias para llevar esta conexión a cabo, así como la representación gráfica de los elementos espaciales contenidos en la base de datos. De esta forma, son satisfechos los requerimientos funcionales planteados en el diseño de la herramienta. En cuanto a la clase Neo4QGISDialgog, se presenta una herencia con respecto a la clase descrita anteriormente. Se obtienen los elementos necesarios para ejecutar los métodos una vez la herramienta ha realizado las validaciones respectivas.

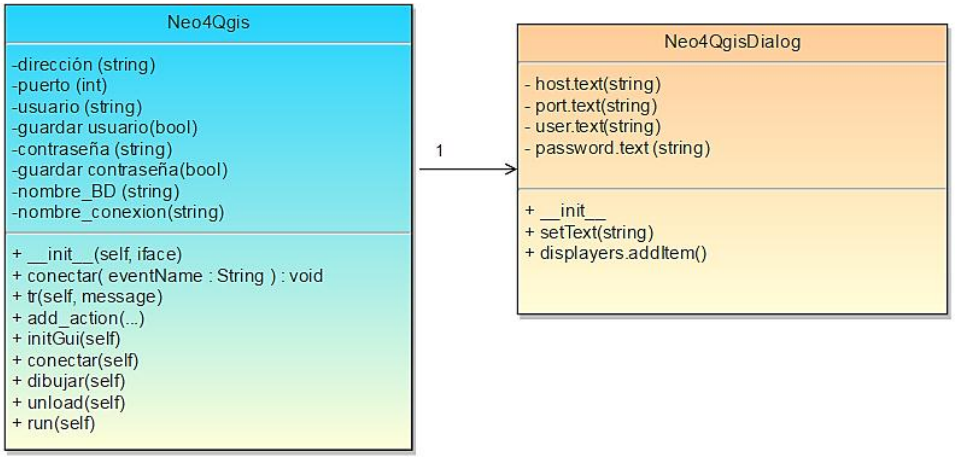


Figura 7-4: Diagrama de clases, Fuente propia.

7.2.4 Vista de alto nivel

En la figura 7-5 se presenta la vista de alto nivel, donde se muestra la organización fundamental de la herramienta que permite la conexión a bases de datos orientadas a grafos a través de un software para el manejo de sistemas de información geográfica. Se elige a QGIS como la plataforma base, puesto que es un programa que cuenta con Licencia Pública General GNU que permite a los usuarios finales la posibilidad de usar, copiar, alterar y analizar el software (Porcel & Rodríguez, 2005). Este software se encuentra programado en C++, Python y QT, lo que facilita el uso de diferentes librerías y bindings de código abierto y libre distribución, permitiendo así la difusión entre la comunidad con interés en las bases de datos orientadas a grafos desde una perspectiva espacial.

Se tuvo un especial interés en este software puesto que satisface las reglas de negocio al contar con herramientas de visualización y despliegue aptas para los objetos geográficos trabajados en la base de datos orientada a grafos, además es multiplataforma, usable, amigable y eficiente, y permite implementar extensiones con la posibilidad de configurar en estas un solo flujo de procesos con los parámetros requeridos para la conexión, valiéndose del framewor QT y el lenguaje Python tal como se explica a continuación.

La capa de presentación se encuentra soportada en la librería que sirve de conector para python al framework multiplataforma QT (PyQT) Este binding de la biblioteca gráfica QT opera en un sistema operativo multiplataforma. La capa de negocio está soportada por la librería que permite utilizar los elementos de QGIS desde python (PyQGIS), la cual cuenta con una sintaxis fácil de entender que soporta programación imperativa y orientación a objetos. Por otra parte, Py2neo es una librería que permite la conexión de Neo4j con aplicaciones creadas en Python desde la línea de comando (Small, 2016). Finalmente, se elige a Neo4j como base de datos orientada a grafos. Este motor es implementado en Java y opera en sistema operativo multiplataforma. La estructura de datos espaciales proviene

de un plugin llamado Neo4j-Spatial, el cual soporta las geometrías de punto, línea, polígono, multipunto, multilínea y multipolígono, tal como es descrito por las especificaciones OpenGIS. Neo4j tiene un modelo único de datos que guarda objetos y relaciones como nodos y vértices en un grafo (Baas, 2012).

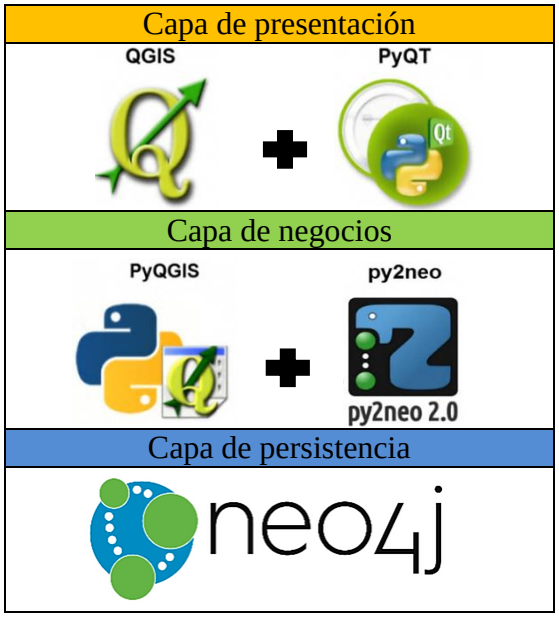


Figura 7-5: Vista de alto nivel, Fuente propia.

7.2.5 Vista de Despliegue

En la figura 7-6 se presenta la vista de despliegue. Esta vista permite apreciar como nodos la topología del hardware sobre la cual opera la herramienta y luego establece la relación con el software, el cual es representado en sus componentes físicos, considerando las asociaciones que existen entre estos.

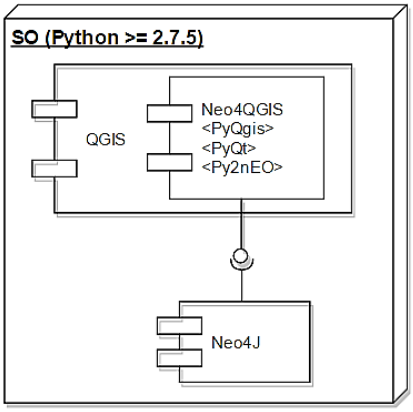


Figura 7-6: Vista Despliegue, Fuente propia.

El plugin se ejecuta sobre un entorno Python superior a la versión 2.7.5. QGIS es el componente que corresponde al software para el manejo de sistemas de información geográfica sobre el cual el plugin se va a ejecutar. El componente relativo al plugin recibe el nombre de Neo4QGIS, el cual utiliza el binding PyQT para implementar la capa de presentación, mientras que la librería PyQGIS permite utilizar los elementos de QGIS desde Python. La librería Py2Neo es la responsable de la conexión entre Neo4j con elementos definidos en Python, en este caso el plugin. Finalmente, el componente Neo4j corresponde a la base de datos NoSQL orientada a grafos, la cual cuenta con instanciación de objetos espaciales con geometría tipo punto.

7.2.6 Vista de implementación

La figura 7-7 contiene la vista de implementación, la cual muestra los componentes en términos de clases, permitiendo la visualización del modelo físico en términos del modelo conceptual.

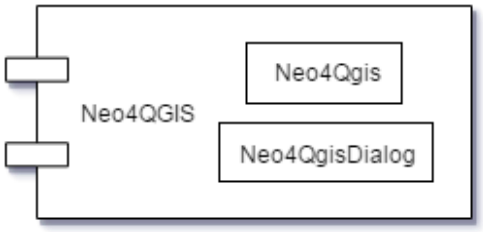


Figura 7-7: Vista de Implementación, Fuente propia.

Las clases se encuentran dentro del componente del aplicativo, donde se tienen todas las librerías y bindings necesarios para la ejecución de este. Las clases contienen los métodos necesarios para realizar la conexión y la representación gráfica de los objetos espaciales de la base de datos, razón por la cual el componente Neo4j no está en términos de ninguna clase de forma explícita. La relación se produce a través de la conexión con el aplicativo.

7.3 Implementación

La implementación da inicio de forma preliminar en la fase de elaboración, pero tiene su pleno desarrollo en la etapa de construcción. El desarrollo consistió en primera instancia en la definición de la conexión de Neo4j con Python, fase en la que además tuvo lugar la instanciación de objetos geográficos en la base de datos. Posterior a esto, se definió el despliegue de los objetos mediante el uso de PyQGIS, a la vez que se implementan las respectivas herramientas en la interfaz de usuario.

7.3.1 Creación de la conexión de Neo4J con Python

Existe una librería que sirve como cliente y proporciona además un set de herramientas para trabajar con Neo4j desde script y aplicaciones desarrolladas en lenguaje Python, así

como desde la misma línea de comandos. Para el desarrollo de este proyecto esta librería se encontró fácil de instalar y los comandos no fueron muy complejos de entender.

La apertura de la conexión entre python y Neo4j se realizó haciendo uso de las siguientes sentencias de código:

```
def conectar(self):
        host = self.dlg.host.text()
        port = self.dlg.port.text()
        user = self.dlg.user.text()
        clav = self.dlg.password.text()
        self.dlg.dbase.setText("http://" + host + ":" + port + "/db/data/")
        try:
                authenticate(host + ":" + port, user, clav)
                DEFAULT_DB = self.dlg.dbase.text() # "http://localhost:7474/db/data/"
                self.dlg.dbase.setText(DEFAULT_DB)
                test = Graph(DEFAULT_DB)
                test.cypher.execute("MATCH (n) RETURN n")
                self.dlg.dbase.setStyleSheet('color: Green')
                self.dlg.dbase.setText("Conexion OK!! to DB \n" + DEFAULT_DB)
                except Unauthorized:
                self.dlg.dbase.setStyleSheet('color: red')
                self.dlg.dbase.setText("Datos Erroneos: \n Usuario o contrasena")
                except NetworkAddressError:
                self.dlg.dbase.setStyleSheet('color: red')
                self.dlg.dbase.setText("Datos Erroneos: \n Host")
        except SocketError:
                self.dlg.dbase.setStyleSheet('color: red')
                self.dlg.dbase.setText("Datos Erroneos: \n Puerto")
        graph = Graph(DEFAULT_DB)
        layers = graph.cypher.execute("MATCH (n) WHERE has(n.layer) RETURN DISTINCT
        \"node\" as element, n.layer AS layer UNION ALL MATCH ()-[r]-() WHERE has(r.layer)
        RETURN DISTINCT \"relationship\" AS element, r.layer AS layer")
        for i in range(len(layers)):
        self.dlg.displayers.addItem(layers[i][1])
```

Figura 7-8: Líneas de código usadas en la conexión.

Luego de esto se definió un objeto espacial en la base de datos orientada a nodos, donde nos permite realizar la visualización del grafo de datos de prueba cargados en Neo4j para el plugin, como se observa en la siguiente figura:

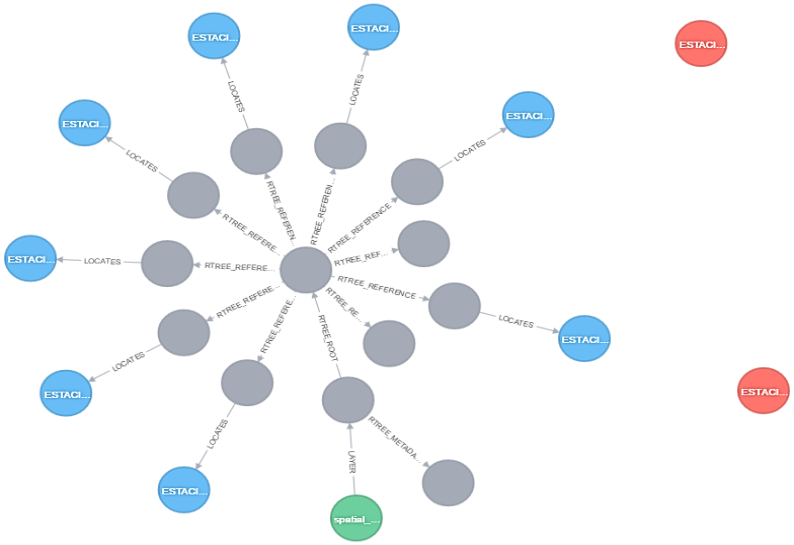


Figura 7-9: Presentación de los puntos cargados en Neo4j, Fuente propia.

7.3.2 Creación de la clase dibujar

Una vez establecida la conexión, se realizó la visualización de las capas por medio de las siguientes sentencias de código.

```
def dibujar(self):
        if (self.dlg.displayers.currentText() == "SInt"):
                MLayer = QgsVectorLayer("Point", "SInt", "memory")
                nestaci = graph.cypher.execute("MATCH (n:SInt) RETURN
                n._py2neo_geometry_name")
                geom = graph.cypher.execute("MATCH (n) WHERE has(n.wkt) \
                AND n.wkt CONTAINS \"POINT\" RETURN \
                DISTINCT \"node\" as element, n.wkt AS wkt LIMIT 25 UNION ALL \
                MATCH ()-[r]-() WHERE has(r.wkt) RETURN DISTINCT \"relationship\" \
                AS element, r.wkt AS wkt ")
        elif (self.dlg.displayers.currentText() == "Loca"):
                MLayer = QgsVectorLayer("Polygon", "Loca", "memory")
                nestaci = graph.cypher.execute("MATCH (n:Loca) RETURN
                n._py2neo_geometry_name")
                geom = graph.cypher.execute("MATCH (n) WHERE has(n.wkt) AND n.wkt CONTAINS
                \"POLYGON\" RETURN DISTINCT \"node\" as element, n.wkt AS wkt UNION ALL
                MATCH ()-[r]-() WHERE has(r.wkt) RETURN DISTINCT \"relationship\" AS
                element, r.wkt AS wkt")
        else:
                self.dlg.dbase.setStyleSheet('color: red')
                self.dlg.dbase.setText("Datos Erroneos: \n No se puede Dibujar Capa
                Seleccionada")
        for i in range(len(geom)):
                print nestaci[i][0]+ " <--> " + geom[i][1]
                the_geometry = QgsGeometry.fromWkt(geom[i][1])
                fet = QgsFeature()
                fet.setGeometry(the_geometry)
                MLayer.dataProvider().addFeatures( [fet] )
        MLayer.updateExtents()
        QgsMapLayerRegistry.instance().addMapLayer(MLayer)
```

Figura 7-10: Líneas de código usadas en la conexión.

7.3.3 Visualización de la interfaz

La interfaz de la herramienta se compone de cuatro parámetros que debe introducir el usuario, los cuales corresponden a host, puerto, usuario y password, una vez introducidos estos valores se debe realizar conexión por medio del botón *Conectar*. Al establecer la conexión aparecerá en la lista desplegable de *Capas Disponibles* las capas existentes en la base de datos Neo4J, donde se debe seleccionar la capa que se quiera visualizar por medio del botón *Dibujar*. Adicionalmente existen dos botones que facilitan la salida de la interfaz *Aceptar* que se utiliza cuando el usuario ha completado los ítems o *Cancelar* para salir de la interfaz.

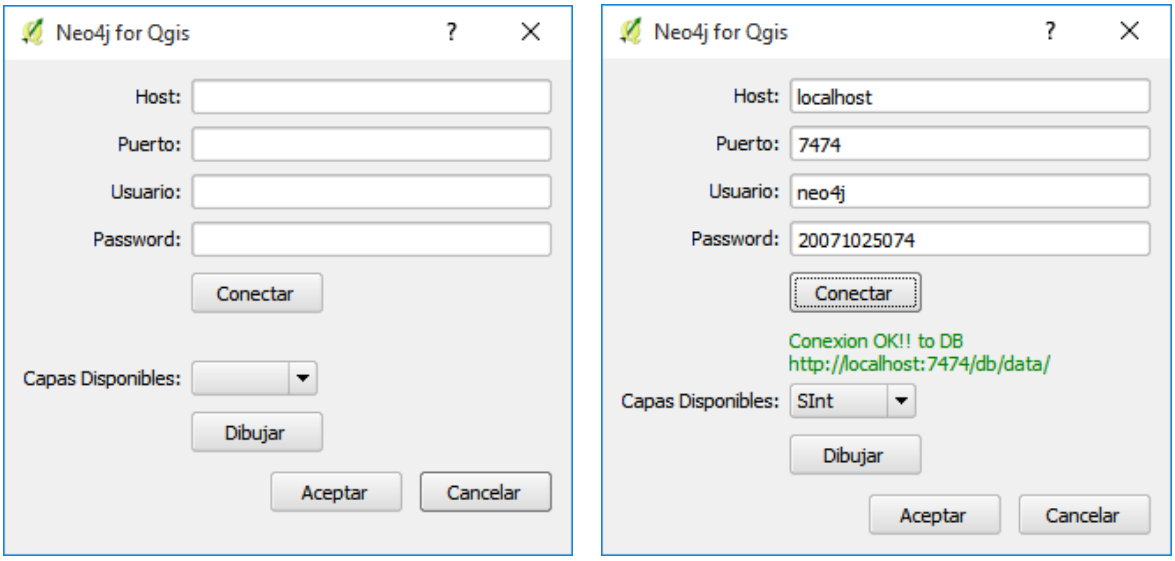


Figura 7-11: Visualización de la interfaz gráfica, Fuente propia

7.4 Pruebas

La evaluación de calidad se realizó de acuerdo al estándar ISO 9126 (ISO/IEC, 2001), que presenta un marco conceptual para el modelo de calidad y define un conjunto de características que son refinadas en subcaracterísticas y las cuales deben cumplir todo producto software para ser considerado de calidad (Alfonzo, 2012).

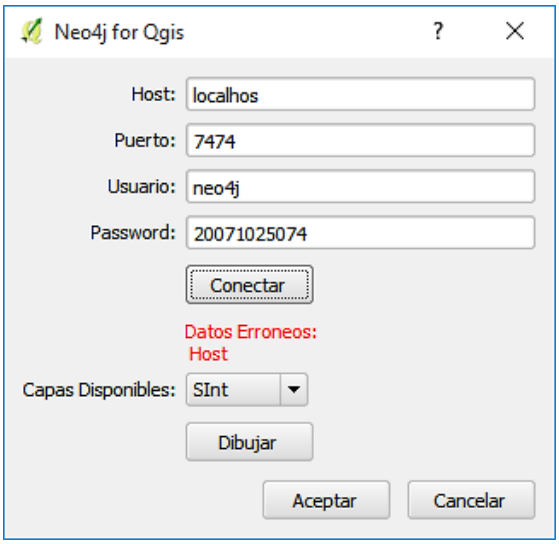
De acuerdo a la metodología de evaluación Tabla 6-1, se realiza la evaluación cuantitativa de cada una de los criterios dando como resultado el 90% de satisfacción.

Característica de Calidad Interna	Nombre de la Métrica	Pregunta	Resultados
Funcionalidad	Cumplimiento de la funcionalidad	¿El plugin permite realizar la conexión a la base de datos orientada a grafos Neo4j?	5
		¿El plugin permite visualizar los elementos geográficos almacenados en la base de datos Neo4j?	5
	Idoneidad	¿El plugin contiene los elementos necesarios para definir la conexión?	5
		¿El plugin contiene los elementos necesarios para visualizar los datos en QGIS?	5
Usabilidad	Facilidad de aprendizaje	¿Qué tan oportuna es la retroalimentación por parte de la herramienta cuando se presentan errores al ingresar los parámetros para definir la conexión o	4

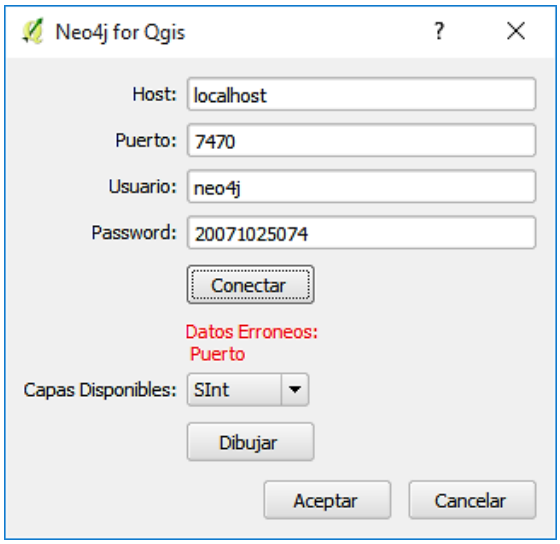
		visualización?	
	Comprensión	¿Qué tan intuitivas son las funciones presentes en el plugin?	4
	Operabilidad	Califique la dificulta para llevar acabo cada una de las operaciones (conexión y visualización), teniendo en cuenta la cantidad de pasos para su ejecución.	4
	Atractividad	Que tan adecuada es la disposición y distribución de los elementos de la interfaz	5
Portabilidad	Facilidad de instalación	Califique el esfuerzo necesario para instalar el plugin.	4
		¿Qué tan pertinente es la documentación para llevar acabo la instalación?	4
Resultado general			4.5

Tabla 7-2: Resultado de la evaluación por los usuarios

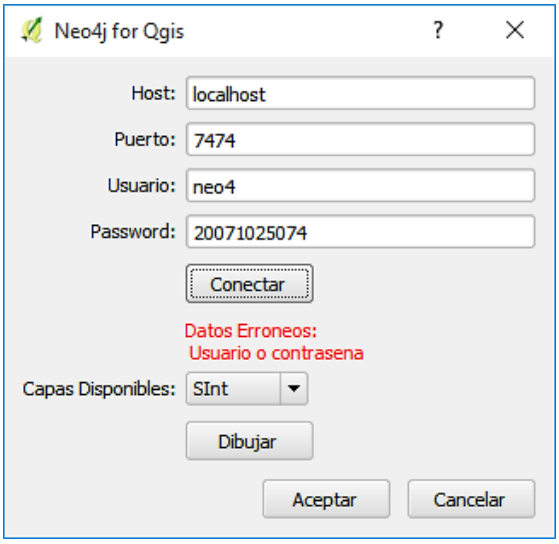
Adicionalmente se realizó la prueba de posibles errores que los usuarios pueden presentar al indica algún campo erróneo, validando así que los mensajes de retroalimentación que presenta la herramienta fueran coherentes con los errores cometidos y ayudaran a la solución del mismo, como se observa en la siguiente figura.



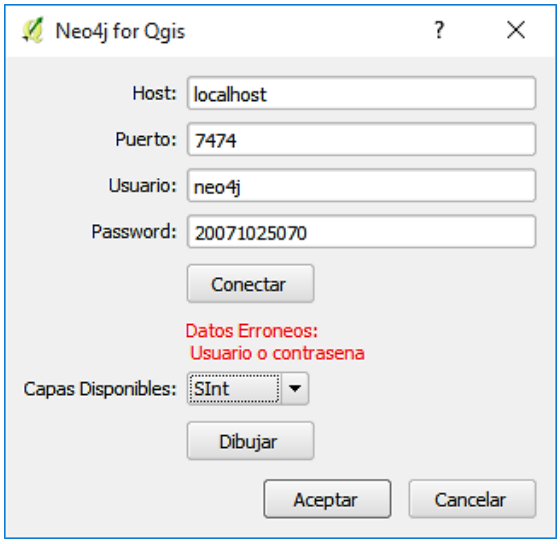
a) Introducir mal el host



b) Introducir mal el puerto



c) Introducir a mal el usuario



d) Introducir mal la contraseña

Figura 7-12: Prueba de posibles errores, Fuente propia

Conclusiones y recomendaciones

El presente capítulo aborda las conclusiones del trabajo desarrollado, así como las recomendaciones enfocadas a una serie de aspectos importantes a tener en cuenta para las próximas investigaciones orientadas análisis, diseño e implementación de herramientas para la gestión de información geográfica en bases de datos orientadas a grafos.

Conclusiones

Se realizó el análisis, diseño e implementación de la herramienta para la gestión de información geográfica en bases de datos orientadas a grafos desde un cliente SIG, teniendo como base la metodología de desarrollo de software basada en cortos ciclos de programación, que cuenta con la posibilidad de trabajar en paralelo la arquitectura e implementación de la herramienta. Lo cual representa ventajas en la calidad y en los tiempos de desarrollo para aplicativos sencillos como por ejemplo Neo4QGIS, pues permite la sinergia del equipo de trabajo y la rápida adaptación ante las eventualidades presentadas durante la implementación.

Se logró la integración de programas para el manejo de sistemas de información geográfica (QGIS) con motores de bases de datos orientados a grafos (Neo4j) a través de librerías y bindings para python, así mismo se pudo realizar la visualización de los datos, cumpliendo con el los requisitos de usuario al generar la herramienta que permitió establecer una conexión entre un software para el manejo de sistemas de información geográfica y una base de datos orientada a grafos, además de permitir la lectura y visualización de la información espacial contenida en esta.

En el contexto de información geográfica, el tratamiento de los datos implica múltiples desafíos, uno de los cuales es el almacenamiento. Este factor puede ser determinante en el éxito o fracaso de un determinado proyecto, por lo cual se considera que el gran reto que se presenta al nuevo paradigma de bases de datos espaciales orientadas a grafos es proporcionar ventajas comparativas con respecto a las bases de datos relacionales para que no ocurra con estas lo que ocurrió con las bases de datos orientadas a objetos que prometían mucho pero no lograron su objetivo.

Finalmente se pudo evidenciar que las capacidades de las bases de datos geográficas orientadas a grafos son muchas, pero también el hecho que proporciona una facilidad de uso no tan alta y una curva de aprendizaje algo elevada pone en manifiesto que aún queda un buen camino por recorrer en su intento de posicionarse en el mercado de las bases de datos espaciales.

Recomendaciones

El presente trabajo sirve como base para continuar con el manejo de bases de datos orientadas a grafos dese el ámbito espacial. Por una parte, se puede ampliar a todas a las primitivas espaciales tales como líneas y polígonos. Por otra parte se pueden incluir funciones de creación, edición y eliminación de los objetos geográficos, además de las actuales funciones de visualización.

Es importante así mismo difundir entre los usuarios de sistemas de información geográfica los beneficios de trabajar con estos motores de bases de datos, pues de esta forma la diversificación de nuevas aplicaciones implicaría un aumento de las necesidades informáticas y con ello una ampliación y consolidación de herramientas que permitan satisfacer dichas necesidades desde los sistemas de información geográfica.

A partir de la experiencia en el desarrollo es importante recalcar que el lenguaje de consulta Cypher que se debe aprender es confuso en principio y cuesta adaptarse si uno tiene experiencia con las bases de datos relacionales, pero si es por primera vez es posible que no sea traumático el aprendizaje como si en cambio de paradigma.

El trabajo realizado es innovador en un contexto sobre el cual las aplicaciones de herramientas geográficas a bases de datos espaciales orientadas a grafos no se encuentran muy desarrolladas. Es por esto que se presentan retos de estandarización, documentación y fiabilidad, pero de igual forma, su incursión en campos donde se requiere por ejemplo el manejo de datos no estructurados, como los sistemas espaciales, va abriendo camino para atenuar dichas desventajas, mejorar sus capacidades actuales e implementar nuevas funcionalidades que le permitan alcanzar su madurez.

Bibliografía

Alfonzo, P. (2012). Revisión de modelos para evaluar la calidad deproductos Web. Experimentación en portales bancarios del NEA. *Universidad Nacional de La Plata*.

Araque, A. (2012). *Sistema de Información Geográfica para la mejora de la gestion y a toma de decisiones difusa en entornos oleícolas.* Jaén, España: Escuela Politécnica Superior de Jaén, Departamento de Informática.

Baas, B. (2012). NoSQL spatial: Neo4j versus PostGIS. *TU Delft, Delft University of Technology*.

Baduino, R. (2007). Introduction to OpenUP (Open Unified Process). *Eclipse Site*.

Browne, J. (2009). Brewer's CAP theorem. *J. Browne blog*.

Euler, L. (1766). Solutio problematis ad geometriam situs pertinentis. *Reimpreso en Opera Omnia Series Prima, Vol. 7.*, 1-10.

Garay, J. (1981). The Transaction Concept: Virtues and Limitations. *VLDB, 81*.

González, J., & Cáceres, G. (2013). Comparison of GIS Desktop Tools for Development of SIGPOT . *IEEE latin america transactions*, Vol. 11, No. 1.

Group, B.R. (2001). Business Rules, What are they Really. *http://www.businessrulesgroup.org*.

Hunger, M., Boyd, R., & Lyon, W. (2016). The Definitive Guide to Graph Databases. *Neo4j*.

ISO/IEC, 9.-1. (2001). Software engineering - Product quality - Part 1. *International Organization for Standardization*.

Martinez, C., & Michel, R. (2007). Aplicación de bases de datos orientadas a grafos en la astrofísica. *Revista Iberoamericana para la Investigación y el Desarrollo Educativo* .

Neo4j. (2016, Abril 28). *Base de datos de grafos - Neo4j*. Retrieved from http://neo4j.com/developer/graph-database/

OpenUP. (2016, 04 29). *Eclipse Public* . Retrieved from http://epf.eclipse.org/wikis/openup/

Porcel Iturralde, M. L., & Rodríguez Maderos, M. (2005). Software libre: una alternativa para las bibliotecas. *Acimed*.

Ramírez, H., & Herrera, J. (2013). A JOURNEY THROUGH SPATIAL DATABASE NoSQL. *Redes de ingeniería, Universidad Distrital*, Vol 4, Núm 2.

Robinson, I., Jim, W., & Eifrem, E. (2015). *Graph Databases.* United States of America: O'Reilly Media.

Small, N. (2016). *py2neo*. Retrieved from The Py2neo v3 beta 2 Handbook: http://py2neo.org/v3/#the-py2neo-v3-beta-2-handbook

Taverner, C. (2011, Marzo 8). *Neo4j*. Retrieved 04 27, 2016, from http://neo4j.com/blog/neo4j-spatial-part1-finding-things-close-to-other-things/

Tudorica, B. G., & Bucur, C. (2011). A comparison between several NoSQL databases with comments and notes. *Roedunet International Conference (RoEduNet), 2011 10th*.

Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., & Wilkins, D. (2010). A Comparison of a Graph Database and a Relational Database. *A Data Provenance Perspectiv*.

Washington , V. (2014). Bases de datos orientadas a grafos y su enfoque en el mundo real. *Escuela Técnica Superior de Ingenieros en Telecomunicaciones* .