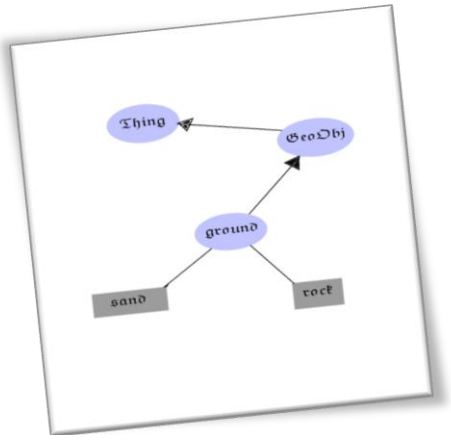
Título: Generación de consultas para la manipulación de Geo-ontologías desde la plataforma GeneSIG.

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

Autor: Adrian Fuentenegra Pérez.
Tutores: Ing. Adrián Gracia Águila.
MSc. Manuel Enrique Puebla Martínez.
Co-tutor: Ing. Alain León Companioni.



DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco 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 _____.

Tutores:

Ing. Adrián Gracia Aguila

_____

Autor:

Adrian Fuentenegra Pérez

MSc. Manuel E. Puebla Martínez

_____

Firma del Autor

_____

Firma del/de los Tutor(res)

Agradezco:

A mi mamá Amarilis por traerme al mundo y haber sido paciente conmigo hasta estos días y mejor madre imposible.

A mi papá Ernesto, quién es mi ídolo, mi viejo, mi confidente.

A mi hermana preferida (la única y menor que yo) Ariadna, de la cuál heredé su inteligencia, jeje.

A mi novia Marlen, quién ha sido mi apoyo directo durante estos últimos años.

A mis abuelos Argelio y Delia a quienes quiero mucho y deseo mucha salud.

A mis tíos Norge, Anelis y Aracelis, Gudelia, Giraldo y Yolanda.

A mis primos Mandy, Lily, Karlen, Anabel, Yasmany, Gabriel, Erick, Ariel, Iroel y Eldy.

A mis amigos Yisel y Rafael Andrés por darme su apoyo y consejos durante toda la carrera.

A mis amigos Luis Gabriel, Reynel, Félix, Luis Ángel, Adrian Bello, Guillermo y Yanier García, a quienes conocí aquí y siempre recordaré con mucho aprecio.

A mi colega de trabajo Aurelio por su dedicación y colaboración en esta investigación.

A Adrián Gracia y a Puebla, mis tutores por confiarme esta investigación y saber guiarme en su desarrollo.

A Alain Companioni, por sus consejos y recomendaciones como co-tutor de esta investigación.

A todos mis compañeros de aula y de proyectos durante estos 5 años de carrera.

A mis colegas de la comunidad del Software Libre, en especial al equipo de Nova, por ser mi inspiración cómo informático.

A los profes que contribuyeron en mi formación como profesional.

Y a todo el que lea estas líneas.

Dedico esta investigación:

Especialmente a mi papá, mi mamá y mi hermana, las tres personas que mayor papel han jugado en lograr ser la persona que soy hoy día, quienes me han apoyado en todo momento y he aquí el fruto de su labor como buena familia que son.

Al Comandante en Jefe Fidel, por regalarnos esta Revolución y esta Universidad de Excelencia, en la cuál pude formarme como profesional y mejor revolucionario.

Frase célebre:

"Usar la ciencia y la computación sin caer en lenguaje tecnicista e ininteligible de élites especializadas. Sed de saber, constancia, ejercicios físicos".

Fidel Castro Ruz

Resumen:

El gran volumen de información evidente en las Bases de Datos Espaciales provoca dificultades en la recuperación eficaz del conocimiento embebido en las estructuras geográficas almacenadas. La información consultada por el usuario no satisface del todo sus necesidades, aún sabiendo que se puede obtener más información a través de esas consultas.

La incorporación de metadatos semánticos surge como mecanismo de transformación de los espacios de información a espacios de conocimientos. La idea central de esta investigación consiste en lograr sentar las bases para que un Sistema de Información Geográfica sea capaz de reutilizar, manipular y compartir el conocimiento que tiene implícito en sus estructuras de datos y bases de datos espaciales para convertirse en un Sistema de Información Geográfica Gobernado por Ontologías.

Palabras Claves: Geo-ontología, Ontología, OWL, Sistemas de Información Geográfica Gobernados por Ontologías, Sparql.

Tabla de contenido

Introducción.

Las Ciencias Informáticas en el mundo actual han crecido vertiginosamente. El tajante auge de las telecomunicaciones a través de las nuevas tecnologías de la información y las comunicaciones (TIC) ha hecho prácticamente imposible que cualquier esfera, tanto económica como social, pueda desarrollarse si no están presentes dichas tecnologías.

Asimismo, la Web, como espacio estructurado y preparado para el intercambio de información, está diseñada para el consumo humano, donde las páginas web son creadas y extendidas por las personas. No existe un formato común para la visualización de dicha información, por lo que los desarrolladores de páginas web crean sus páginas en dependencia de los potenciales usuarios que las consumen, por lo que en los últimos años el formato más usado es la codificación HTML.

La web semántica (WS), creada por Tim Berners-Lee en la década de los ’90, es una web extendida, dotada de mayor significado. Es una potente área en la concurrencia de la Inteligencia Artificial y las tecnologías web, que permite que las máquinas tengan su propio nivel de comprensión de la web, suficiente como para asumir la parte más costosa y habitual del trabajo que usualmente realizan los usuarios que navegan e interactúan en la web. Se basa en el principio de añadir metadatos semánticos y ontológicos a la World Wide Web (WWW). Se desarrolla con lenguajes universales que permiten a los usuarios encontrar respuestas a sus interrogantes de una manera más rápida y eficiente, gracias a la mejor organización y estructuración de la información, por lo que es más viable compartir, integrar y recuperar.

Con el perfeccionamiento de la Web urge la necesidad de compartición y reutilización de la información, así como los estudios realizados en el campo de las ontologías como potenciales estructuras preparadas para la representación de la expresividad semántica embebida en los sitios web, donde su importancia ha crecido exponencialmente cuando el estado de las búsquedas y recuperación de la información comienzan a convertirse en un problema.

El término ontología, en el marco de las TIC, cobra una gran importancia y a su vez, es parte del cimiento que sustenta a la WS, y se fundamenta como representación abstracta del entorno real hacia un dominio específico, definiendo de este último su significado.

El apogeo de las ontologías hoy día es permisible dadas las capacidades que tienen estas estructuras para la supresión de conflictos semánticos y terminológicos. Así mismo, diversos

investigadores han dirigido sus estudios en esa dirección, por la gran importancia que representan hoy día las ontologías. Dentro de estos autores se encuentran: Federico Peinado Gil y su tesis doctoral “Un Armazón para el Desarrollo de Aplicaciones de Narración Automática Basado en Componentes Ontológicos Reutilizables”; Nuria Torres Rodríguez y su tesis doctoral “Sistemas de Análisis Automático de Fotografías. Modelo Conceptual según los Estándares de la Web Semántica”; Francisco García-Sánchez y su tesis doctoral “Sistema Basado en Tecnologías del Conocimiento para Entornos de Servicios Web Semánticos” y José Antonio Macías Iglesias con su tesis doctoral “Autoría de Documentos Web Dinámicos Mediante Ontologías y Técnicas de Programación por Demostración”.

Los Sistemas de Información Geográfica (SIG) no están exentos a esta nueva concepción de manejo de información y actualmente, debido al monumental volumen de datos almacenados referentes a la Tierra, ha surgido un contexto propicio donde la comunidad científica internacional ha manifestado su preocupación por el desarrollo de tecnologías para su unificación y de las herramientas para el manejo y análisis de dichos datos. Según (Martínez-Terrero, y otros, octubre 2010) el desarrollo de ontologías, se presenta como el instrumento adecuado para alcanzar la integración semántica en el entorno de las infraestructuras de los datos espaciales de una manera mucho más abstracta, en la que el conocimiento juegue un papel fundamental. Es así como se han concebido los SIG de nueva generación, en los que las ontologías son un componente activo dentro de su arquitectura. Estos Sistemas de Información Geográfica Gobernados por Ontologías (SIGGO) son construidos utilizando clases derivadas de las mismas, y como resultado es posible utilizar todo el conocimiento embebido; estas clases contienen atributos y funciones que darán soporte a todas las funcionalidades del sistema. La nueva generación de SIGGO deberá ser capaz de resolver la interoperabilidad semántica, en la cual un hecho puede tener un valor mayor que solamente su descripción (Garea, y otros, 2009).

Históricamente, la Cartografía y la elaboración de mapas se han convertido en las herramientas precisas donde las Ciencias Exactas como la matemática y las proyecciones de un geoide o esfera sobre el plano, han sido la guía exitosa hacia la confección de los mapas temáticos y básicos, además de las cartas náuticas, por lo que es posible afirmar que “la confección de mapas y cartas por medios matemáticos exactos es una ciencia madura” (Garea-Llano, y otros). Al existir la ausencia de la unión semántica entre los mapas es cuando entran en escena las geo-ontologías como derivación de las ontologías y a su vez los SIGGO.

El campo de investigación en torno a las ontologías en estos momentos es uno de los más aplicables en las tecnologías de varios países. En Cuba por ejemplo, se destaca la labor del Centro de Aplicaciones de Tecnologías de Avanzada (CENATAV), surgido en el 2004 y está orientado a investigaciones teóricas y aplicadas al área de Reconocimiento de Patrones (RP) y Minería de Datos (MD).

Numerosos profesionales cubanos en estos momentos son de los más destacados con sus pesquisas en el área de las ontologías, geo-ontologías y semántica espacial, entre ellos se encuentran: Dr. Eduardo Garea Llano, jefe del Departamento de RP del CENATAV, Ing. Rainer Larín Fonseca y Lic. Francisco Vera Voronisky, ambos investigadores del Departamento de RP del CENATAV y MSc. Rafael Oliva Santos, Profesor Principal Asistente (PPA) del Departamento de Computación, Facultad de Matemática y Computación (MATCOM), Universidad de La Habana (UH).

Además, Cuba como país en vías de desarrollo no se encuentra ajena a la aplicación de las TIC. Para ello, desde hace algunos años se lleva a cabo el Programa Nacional para la Informatización de la Sociedad Cubana (PNISC), cuyo objetivo principal es acercar a nuestra sociedad, de una forma u otra, al manejo adecuado delas TIC, ya que el bloqueo económico impuesto a nuestro país impide que no se tenga mayor desarrollo en este campo, los cuál dificulta grandemente la economía cubana. A pesar de esto, la Universidad de las Ciencias Informáticas (UCI), nacida al calor de la Batalla de Ideas, visionada por el Comandante en Jefe Fidel Castro como “universidad de excelencia”, forma parte activa del PNISC y tiene, entre otras, la misión de producir software de calidad, tanto para clientes nacionales como internacionales. En la misma se encuentra además el Centro de Desarrollo Geo-Informática y Señales Digitales (GEYSED), el cual cuenta en su estructura con varias líneas de investigación y desarrollo, entre las mismas se encuentra la línea de Manejo de Información Geográfica, la cual se especializa en la implementación de la plataforma soberana GeneSIG, que es un producto utilizado para el desarrollo de SIG.

Un problema real existente en la actualidad es que las diferentes organizaciones que centran sus responsabilidades en la confección de los mapas y bases de datos espaciales, no llegan a un consenso entre ellas respecto a la expresividad semántica o significado de las estructuras que ellos crean. La situación no recae en cómo representar la información geográfica, sino que

varios mapas o bases de datos espaciales representen la misma información dependiendo del conocimiento almacenado en las estructuras geográficas.

La necesidad de compartición e intercambio de la información se han logrado en buena parte gracias a los SIG, que son capaces de manejar el gran volumen de datos geográficos almacenados. Pero sucede que la interoperabilidad semántica de la información en los SIG aún no ha sido alcanzada (Vckovski, 1999). Por la gran complejidad y abundancia de información contenida en los datos geográficos brota una serie de problemas que específicamente se enfocan en la interoperabilidad dentro de ellos. Las estructuras que permiten manejar esos eventos son las geo-ontologías, que como subconjunto de las ontologías están preparadas en el contexto de los datos geográficos.

La concepción de los SIGGO como fase superior a los SIG, interviene en este proceso como un medio integrador que obvia la clasificación de los datos a través de su forma de representación. Estos sistemas utilizan el punto de vista de la expresividad semántica para de una forma natural integrar los distintos tipos de información a través de entes flexibles, que comúnmente, se encuentran basados en estándares y por consecuente son cimentados en ontologías.

A pesar de las ventajas de GeneSIG respecto a otros SIG urge una creciente necesidad de encontrar una solución que permita la integración de datos geográficos de una manera mucho más abstracta en la que el conocimiento juegue un rol esencial, en la que se explote con mayor efectividad la información semántica existente que se encuentra embebida en los datos almacenados. Dada esta problemática surge como problema a resolver: **¿Cómo manipular, desde la plataforma GENESIG, la expresividad semántica que brindan las Geo-ontologías para obtener y generar nuevos conocimientos?**

Luego de un análisis del problema existente, urge la necesidad de realizar un estudio exhaustivo a **las geo-ontologías como medios de almacenamiento de información semántica**, centrando la atención en **el proceso de realización de consultas semánticas a las geo-ontologías desde los SIG.**

Precisando como objetivo general de la investigación: **Desarrollar un editor de consultas para la manipulación de geo-ontologías desde la plataforma GeneSIG, que permita aprovechar la expresividad semántica que brindan las mismas.**

Para dar cumplimiento al objetivo general se plantean las siguientes tareas de la investigación:

- **Caracterizar el estado del arte relacionado con las geo-ontologías como un sub-conjunto de las Ontologías y los lenguajes de consultas a las mismas.**
- **Analizar el estado actual de la realización de los editores de consultas para la web semántica.**
- **Identificar el lenguaje de consultas a utilizar para acceder a las ontologías.**
- **Identificar las herramientas a utilizar en la construcción de la propuesta de solución.**
- **Fundamentar la metodología de desarrollo de software a usar en el proceso.**
- **Modelar los requisitos funcionales y no funcionales de la propuesta de solución.**
- **Diseñar la propuesta de solución.**
- **Implementar la propuesta de solución.**
- **Validar la propuesta de solución.**

Por lo que urge como idea a defender: que **el desarrollo de un editor de consultas semánticas a geo-ontologías permitirá a la plataforma GeneSIG manipular la expresividad semántica implícita en sus estructuras para obtener y generar nuevos conocimientos.**

En cuya solución se utilizaron como métodos científicos:

Métodos Teóricos:

- **Analítico-Sintético:** en el resumen, enunciación y descripción de los requerimientos enunciados por los desarrolladores de la Plataforma GeneSIG.
- **Análisis Histórico-Lógico:** en el estudio del estado del arte de los editores de consultas a geo-ontologías desarrollados en el país y el resto del mundo, su uso en el ámbito nacional, así como las ventajas y desventajas que poseen.

- **Modelación:** en la realización de una reproducción simplificada de la realidad existente en el manejo de la información desde la plataforma GeneSIG.

Y como posible resultado de la investigación se obtendría: **Prototipo funcional de la aplicación "Editor de consultas para la manipulación de Geo-ontologías desde la Plataforma GeneSIG".**

La presente investigación científica cuenta con una estructura de 5 capítulos que serán pormenorizados con las siguientes cualidades:

Capítulo #1: Se realiza la Fundamentación Teórica y una caracterización general del objeto de estudio de la investigación y de las soluciones existentes.

Capítulo #2: Se especifican las herramientas necesarias para el desarrollo de la solución tomando, así como, argumentar la metodología de desarrollo que mejor se ajusta a la propuesta de solución a la cual se desea arribar.

Capítulo #3: Se realiza un levantamiento de los requisitos funcionales y no funcionales del sistema con el fin de lograr los objetivos planteados, y a partir de la recopilación de los requisitos funcionales construir el Modelo de Casos de Uso del Sistema.

Capítulo #4: Se realizan los diagramas de clases y de secuencia del diseño correspondiente a los casos de uso identificados, el Diagrama de Despliegue y Modelo de implementación.

Capítulo #5: A partir del posible resultado, se realiza la validación de la propuesta de solución.

Capítulo #1 Estudio del arte y Fundamentación Teórica.

1.1 Introducción

En el presente capítulo se describe una panorámica general acerca de los principales conceptos asociados al problema planteado, así como el estado actual del desarrollo de editores de consultas a ontologías y más específicamente las geo-ontologías. Además se valoran las características de los sistemas existentes haciendo uso de las mismas.

1.2 Conceptos asociados al dominio del problema

1.2.1 Ontología:

(Gruber, 1993) Afirma que una ontología no es más que "una especificación explícita y formal sobre una conceptualización compartida", y más tarde, (Gruber, 2001) concluye que las ontologías adquieren nuevos roles dentro de la Inteligencia Artificial (IA), bajo la concepción de que lo que existe es aquello que puede ser representado.

También (Gruber, 2001) plantea que es una especificación formal, explícita de una común conceptualización, es decir provee de un modelo explícito obtenido por consenso descrito en un lenguaje que contiene a los conceptos, propiedades y relaciones más relevantes en un dominio y que es comprensible para una máquina.

Según (Guarino, 1998), una ontología es un artefacto de ingeniería, construido por un vocabulario específico utilizado para describir una cierta realidad, más un conjunto de hipótesis explícitas en relación con el significado de las palabras del vocabulario.

Otro autor, (Swartout, 1997) llega la conclusión que una ontología es un conjunto estructurado de términos que describen algún dominio o tema. La idea es que una ontología proporciona el esqueleto de una base de conocimientos.

El autor de esta investigación considera que las ontologías son la elaboración y formulación de una absoluta e inflexible representación conceptual de los entes que se encuentran bajo las restricciones de uno o varios dominios específicos, que tienen como meta común proporcionar la comunicación y compartición de la información entre dichos entes y los distintos sistemas.

Composición y tipos de ontologías (Gruber, 1993)**.**

En esencia, una ontología está compuesta por los siguientes puntos:

- Clases: Son las ideas a formalizar y representan los conceptos en el sentido más amplio. Las clases en una ontología se suelen organizar en taxonomías a las que se les pueden aplicar los mecanismos de herencia.
- Contexto: Circunstancia bajo la cual un concepto está siendo ocupado.
- Conceptos: Son las ideas básicas que se intentan formalizar. Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.
- Relaciones: Representan la interacción y enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio. Por ejemplo: subclase-de, parte-de, parte-exhaustiva-de, conectado-a, etc.
- Instancias: Concepto particular que ya no puede ser partido (concepto atómico o concepto terminal).
- Axiomas: Los axiomas formales sirven para modelar sentencias que son siempre ciertas. Además también se usan para verificar la consistencia de la propia ontología.
- Restricciones: Manejadas para omitir algunos casos.
- Atributos: Ayudan a entender mejor a una relación.

Las ontologías se clasifican de acuerdo a su dependencia y relación con una tarea específica desde un punto de vista (Guarino, 1995), (ver Fig. #1)**:**

- Ontologías de Alto Nivel o Genéricas: Describen conceptos más generales. En relación con los Sistemas de Información, estas ontologías describirían conceptos básicos. Por ejemplo, una teoría describiría partes y todos, y sus relaciones con la topología.
- Ontologías de Dominio: Describen un vocabulario relacionado con un dominio genérico. Por ejemplo, podría ser una descripción de datos y entidades relacionados con la sensorización remota con un ambiente urbano.
- Ontologías de Tareas o de Técnicas básicas: Describen una tarea, actividad o artefacto. Por ejemplo, la evaluación de la contaminación sonora en ambientes urbanos o la descripción de características generales de componentes, procesos o funciones.
- Ontologías de Aplicación: Describen conceptos que dependen tanto de un dominio específico como de una tarea específica y, generalmente son una especialización de

ambas. Ellas representan las necesidades de los usuarios relacionados con una aplicación específica.

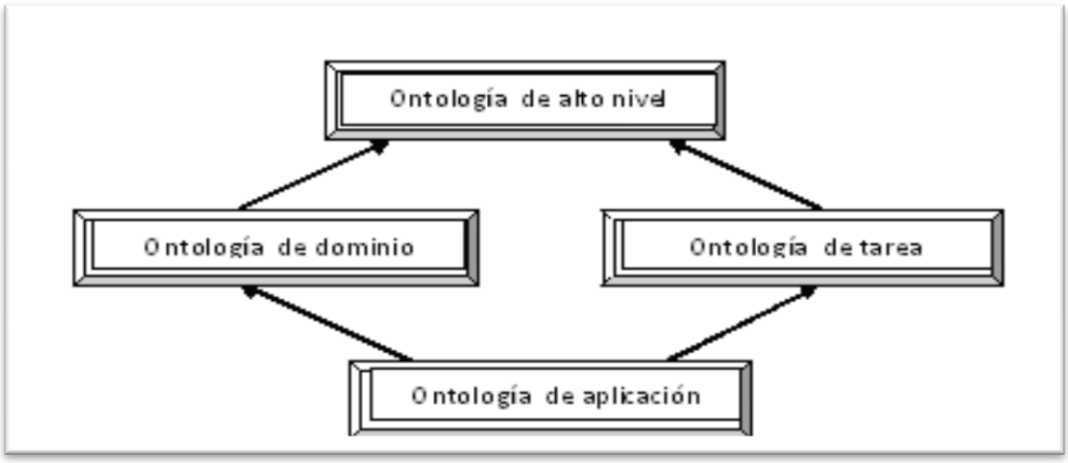


Fig. #1 Tipos de ontologías (Guarino, 1995).

1.2.2 Geo-ontología:

Los autores (Lutz, y otros) determinan que una geo-ontología es una ontología que ofrece una descripción de entidades geográficas y difiere de otras ontologías por la presencia predominante de relaciones topológicas que caracteriza a la información espacial. Una geo-ontología tiene dos tipos básicos de conceptos, los conceptos físicos y los conceptos sociales.

Según (Oliva-Santos, 2009), las geo-ontologías son estructuras en las que es posible la integración o asociación dato-conocimiento. Se explica también que las anotaciones de los objetos geográficos se modelarán como instancias de conceptos pertenecientes a geo-ontologías, de modo que hay que distinguir de cual ontología se está haciendo referencia.

El autor de esta investigación concluye que las geo-ontologías son la rama de las ontologías que se encargan de almacenar, compartir y representar la expresividad semántica existente en los objetos geográficos a través de sus relaciones topológicas. Son fuertes estructuras de metadatos semánticos con contenido de información geográfica muy útil, y su utilización permite aprovechar al máximo todas las bondades que brindan las ontologías como conjuntos de representación del conocimiento.

Características de las geo-ontologías.

Dentro del dominio geo-espacial las ontologías se utilizan en la modelación de un fenómeno espacial, en la representación de conceptos geográficos y las relaciones entre los mismos. Una geo-ontología, es una ontología que representa la información de un dominio de expresividad geo-espacial y brinda una descripción formal de la semántica de las entidades y sus relaciones. Una geo-ontología es capaz de describir los objetos que se le pueden asignar localizaciones en la superficie terrestre y a las relaciones existentes entre estos (ej: Ver Fig #2).

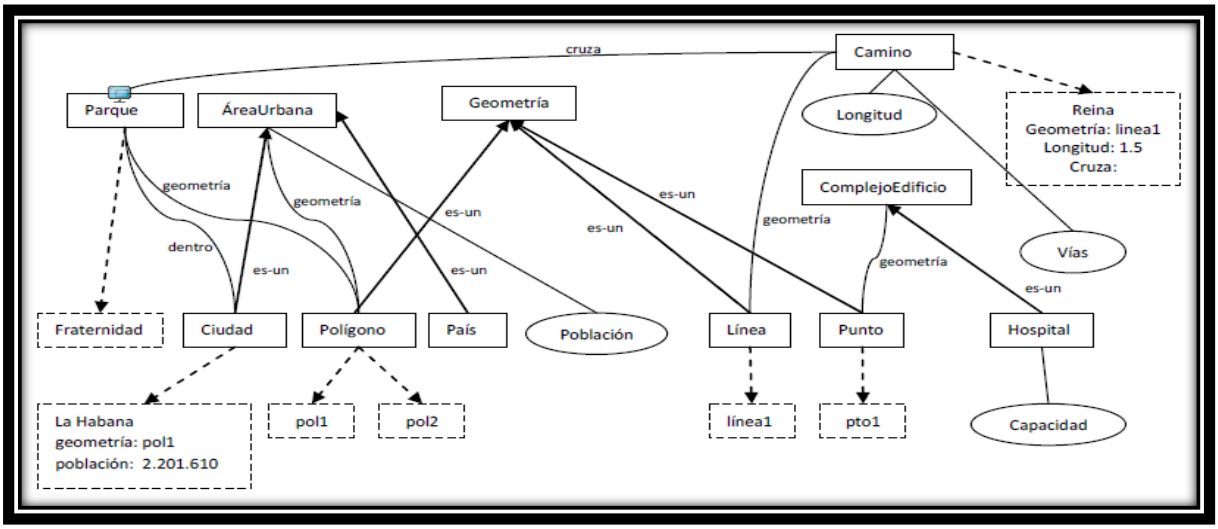


Fig. #2 Representación gráfica de una geo-ontología (Garea, y otros, 2009).

1.2.3 Geo-semántica.

El área de estudio de la geo-semántica se encuentra en una fase de crecimiento vertiginoso, donde la utilización de las ontologías juega el papel más importante. Esta área cuenta con el soporte de más de tres décadas de desarrollo de SIG, también la experiencia en la Ingeniería Ontológica como rama de la Inteligencia Artificial, fundamentando teóricamente la próxima generación de los SIG, los SIGGO.

1.2.4 Sistemas de Información Geográfica.

Un SIG es un intento más o menos logrado según los casos de constituir una visión esquemática de una realidad compleja (Bosque-Sendra, 1992).

Un SIG puede ser concebido como una especialización de un sistema de bases de datos, caracterizado por su capacidad de manejar datos geográficos, que están geo-referenciados y los cuales pueden ser visualizados como mapas (Braken, 1992).

Se puede definir SIG como una base de datos especializada que contiene objetos geométricos (Cebrian, 1994).

El autor de la presente investigación afirma que un SIG no es más que la integración de programas informáticos, periféricos y medios implementados, capaces de capturar, analizar y representar información de los objetos geo-referenciados a través de mapas temáticos en un ordenador. Dicha información puede ser consultada, compartida o modificada según las necesidades del usuario del SIG. Son sistemas que facilitan la toma de decisiones respecto a la información geo-referenciada que estos brindan.

Componentes y características de un SIG. (SARABIA-LÓPEZ, Diciembre 2008)

En breve se enuncian los principales componentes de un SIG, así como sus principales características.

Componentes básicos de un SIG.

- **Equipos** (Hardware): permite la entrada y salida de la información geográfica en diferentes medios y formas.
- **Programas** (Software): nos proporciona las herramientas necesarias para almacenar, analizar y desplegar la información geográfica.
- **Datos**: se hace referencia a la información que nos garantice el funcionamiento analítico de nuestro SIG.
    - ✓ Los datos geográficos pueden ser obtenidos directamente por la persona que implementa el sistema o tomar la información ya disponible, por ejemplo diccionarios de datos geográficos del INEGI.

- ✓ El sistema de información geográfica integra los datos espaciales con otros recursos de datos y pueden incluso utilizar los manejadores de bases de datos más comunes para manejar la información geográfica.

- **Recursos Humanos**: este bloque hace referencia al personal que opera, desarrolla y administra el sistema; y establece planes para aplicarlo en problemas del mundo real.

- **Metodologías**: el SIG debe operar bajo un plan bien diseñado para trabajar de acuerdo con aplicación.

Características principales de un SIG.

1) La capacidad de visualización de información geográfica compleja a través de mapas.

2) La funcionalidad de los SIG como una base de datos sofisticada, en la que se mantiene y relaciona información espacial y temática.

3) La diferencia con las bases de datos convencionales radica en que toda la información contenida en un GIS está unida a entidades geográficamente localizadas. Por ello en un GIS la posición de las entidades constituye el eje del almacenamiento, recuperación y análisis de los datos.

4) Son una tecnología de integración de información.

5) Se han desarrollado a partir de innovaciones tecnológicas habidas en campos especializados, de la geografía y otras ciencias (tratamiento de imágenes, análisis fotogramétricos, cartografía automática, etc.), para constituir un sistema único, más potente que la suma de las partes.

6) Permiten unificar la información en estructuras coherentes.

7) Este carácter integrador y abierto, hace de los SIG un área de contacto entre variados tipos de aplicaciones informáticas, destinadas al manejo de información con propósitos y formas diversas; por ejemplo: programas estadísticos, gestores de bases de datos, programas gráficos, hojas de cálculo, procesadores de texto, etc.

8) Los límites y diferencias entre los SIG, los programas de diseño asistido por computadora (CAD), los de cartografía temática y los de tratamiento de imágenes son especialmente difusos. Aunque sus diferencias estriban sobre todo en el modelo de datos y en las capacidades de análisis de información espacial.

1.2.5 Sistemas de Información Geográfica Gobernados por Ontologías

Un SIG que utiliza ontologías para generar nueva información a partir de un conjunto previo de datos es lo que se denomina SIGGO. En los SIGGO las ontologías son una componente más, como lo es la base de datos temáticos o espaciales que interviene y coopera de la misma manera para alcanzar los objetivos para los cuales fue creado (Larin, y otros).

Los SIGGO no son más que SIG en los que se incorporan el uso de ontologías como un componente activo. Los SIGGO son construidos utilizando clases derivadas de las ontologías y como consecuencias de esto se extrae el conocimiento embebido en estas para aportar mayor eficiencia y robustez al sistema (Larin, y otros).

Los SIGGO surgen a partir de que los SIG convencionales no soportan sus negocios a través de la expresividad semántica de los objetos espaciales, pero una vez logrando que un SIG se integre con un sistema proveedor de geo-ontologías pueda hacer uso del conocimiento embebido en estas y a su vez logre generar nuevo conocimiento que consiga almacenarse y ser utilizado posteriormente, se convierte entonces en un SIGGO.

Estructura de un SIGGO.

Los SIGGO tienen dos fases que son fundamentales en su estructura (Garea-Llano, y otros, Febrero 2007), Ver Fig. #3:

1. **Fase de generación del conocimiento:** comprende la especificación de las ontologías utilizando un editor de ontologías, la generación de nuevas ontologías a partir de las ya existentes y la traducción de ontologías a componentes de software.

2. **Fase de Uso del conocimiento:** se apoya en los productos obtenidos en la fase anterior: una serie de ontologías especificadas en un lenguaje formal y en una serie de clases. Las ontologías están disponibles para ser navegadas por el usuario final y para

su utilización en la generación de aplicaciones, análisis y finalmente las posibles alternativas de decisión.

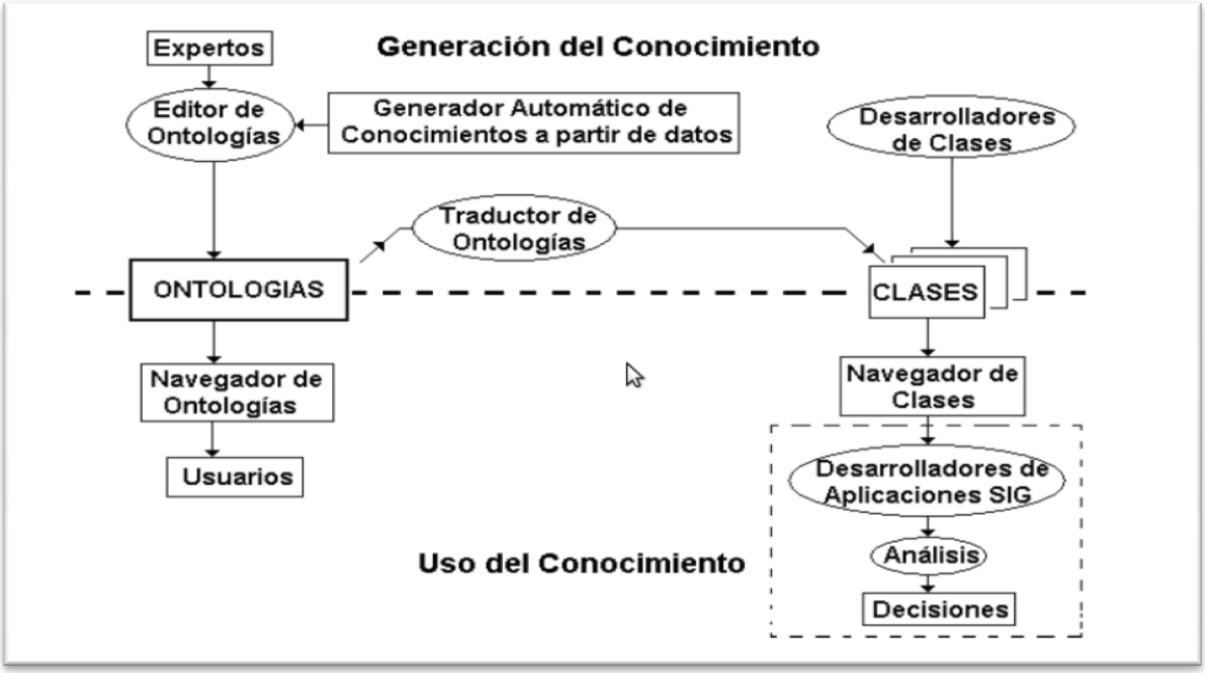


Fig. #3: Estructura de un SIGGO. (Garea-Llano, y otros, Febrero 2007)

1.2.6 Editor de consultas

Un editor de consultas en un programa que permite crear, compilar, modificar y generar consultas a través de un editor de textos siguiendo las especificidades de un determinado lenguaje de programación y es capaz de obtener resultados de dichas consultas.

Editores de consultas para la web semántica.

En esta sección de la investigación se puede observar una panorámica general del estado en que se encuentra el desarrollo de editores de consultas para la web semántica, más específicamente a las ontologías como componentes principales de esta web y su vinculación y/o integración con diferentes tipos de sistemas informáticos dentro de los cuales se encuentran los SIGGO.

Twinkle (Espina)(ver Fig. #4)

Twinkle es una sencilla interfaz gráfica de usuario que contiene el motor de consulta SPARQL ARQ. La herramienta puede ser útil tanto para las personas que deseen aprender el lenguaje de consulta SPARQL, así como los que realizan el desarrollo de la Web Semántica.

Principales características:

- Cargar, editar y guardar las consultas SPARQL.
- Insertar PREFIX en las declaraciones de las consultas.
- Configurar espacios de nombres personalizados para que se puedan insertar rápidamente en las consultas.
- Cancelar consultas de larga ejecución.
- Guardar los resultados en archivo.
- Búsqueda de archivos locales y remotos documentos RDF.
- Consulta de datos RDF en bases de datos relacionales.
- Consulta SPARQL puntos finales en línea, tales como DBpedia, reyvu.com y GovTrack.
- Consulta con SPARQL estándar, o la sintaxis ARQ extendida que soporta COUNT, etc.
- Utilizar ARQ en funciones de extensión y las funciones de propiedad.
- Aplicar inferencia (por ejemplo, normas de Jena, RDF Schema, ontología OWL) al ejecutar las consultas.
- Configurar las fuentes de datos de uso general para el acceso rápido.

Ocurre que Twinkle, a pesar de las características que tiene, al ser de interfaz para escritorio, no se ajusta a la solución de esta investigación, por lo que no fue utilizado, porque para manejar información semántica desde la plataforma GeneSIG se necesita una interfaz Web, con el mínimo de requerimientos, accesible y sin necesariamente tener que instalar la aplicación en la estación de trabajo.

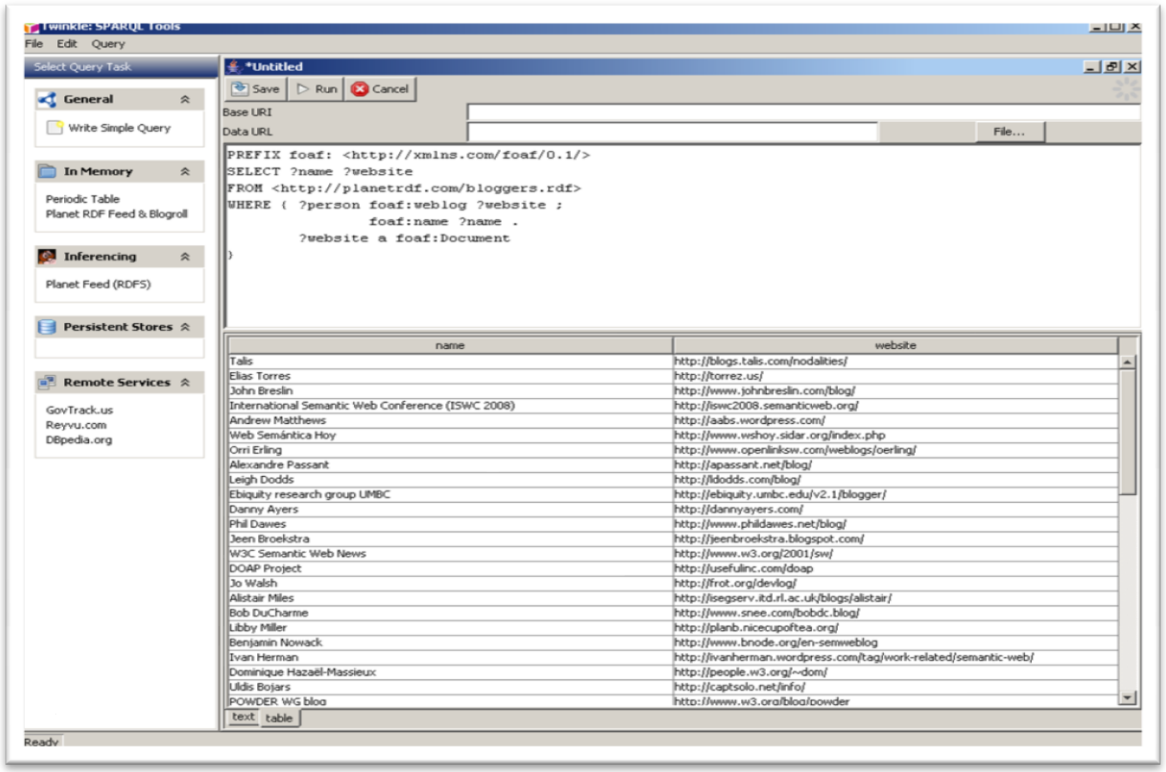


Fig. #4Twinkle (Espina).

SPARQLer (ver Fig. #5)

Es un editor de consultas publicado en internet que utiliza Joseki (Espina) como motor HTTP que soporta el protocolo y el RDF Query Language SPARQL y este último proporciona:

- Manejo de Datos RDF en ficheros y Bases de Datos (BD).
- Implementación GET/POST del protocolo.
- Funcionamiento como servidor.
- Funcionamiento como Servlet[1] dentro de cualquier servidor java (Jetty, Tomcat).
- Está programado en Java y cuenta con licencia de código abierto.

[1]Servlet es un programa que se ejecuta en un servidor.

Fig. #5SPARQLer.

OntoEditor (Vasconcelos, y otros)(ver Fig. #6)

OntoEditor se implementa como un Applet[2] de Java con el modelo cliente-servidor. Cuenta con una interfaz gráfica web usando componentes Swing[3]. Esto permite a los gráficos manipulación de las ontologías, de manera que, el usuario puede crear y visualizar una ontología dada en forma de gráfico. Después de la creación de una ontología, el usuario puede guardarlo en un sistema de base de datos relacional o exportarlo como un archivo de texto.

Para la realización de consultas, OntoEditor utiliza el lenguaje XMLQL a través del procesador del mismo nombre y básicamente el proceso de de consultas consiste en:

- Primero: El sistema monta una consulta XMLQL usando los datos del usuario.
- Segundo: El procesador XMLQL es invocado.
- Tercero: Para alguna ontología que contenga el elemento buscado, un valor es retornado para dicha ontología.

[2]Applet es un componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web.

[3] Swing es una biblioteca gráfica para Java.

- Y por último: Tiene acceso a la base de datos en busca de más información sobre el término recuperados.

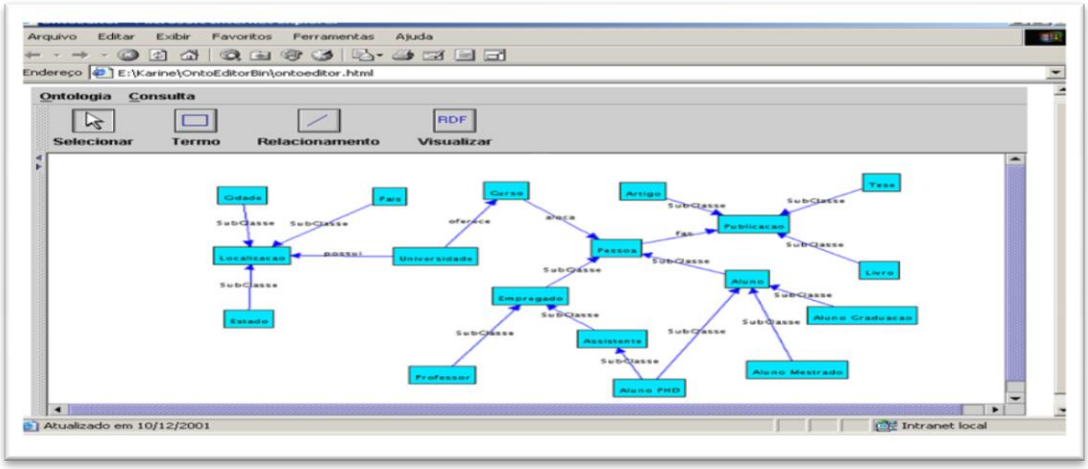


Fig. #6: OntoEditor (Vasconcelos, y otros).

De dicho editor se pudieron utilizar algunas ideas para la posible solución de esta investigación, por ejemplo la forma en que se maneja la información de las ontologías que intervienen durante el proceso de consultas y la forma de acceso a las BD.

1.2.7 Base de datos espacial (BDE)

Un concepto de (NCGIA, 1990) sugiere que una BDE "es una colección de datos referenciados en el espacio que actúa como un modelo de la realidad".

Investigaciones de (SARABIA-LÓPEZ, Diciembre 2008) concluyen que el espacio establece un marco de referencia para definir la localización y relación entre objetos. El que normalmente se utiliza es el espacio físico, el cual es un dominio manipulable, perceptible y que además sirve de referencia. La construcción de una base de datos geográfica implica un proceso de abstracción para pasar de la complejidad del mundo real a una representación simplificada que pueda ser procesada por el lenguaje utilizado por las computadoras.

Se puede entender como BDE a la colección de los datos de los objetos geográficamente referenciados que representan la realidad de una zona geográfica.

1.3 Objeto de estudio.

El objeto de estudio de esta investigación son las geo-ontologías como medios de almacenamiento de información semántica, centrando la atención en el proceso de realización consultas semánticas a las geo-ontologías y para esto es necesario centrarse en el funcionamiento del manejo de la información de la Plataforma GeneSIG, estructura y composición de un SIGGO, así como el estado en que se encuentra el desarrollo de los lenguajes de consultas a ontologías más específicamente las geo-ontologías.

1.3.1 Descripción del Dominio del Problema.

En el Centro GEYSED se cuenta con la línea de desarrollo de Manipulación de Información Geográfica, cuenta a su vez con el producto GeneSIG, que se especializa en la implementación de una plataforma soberana para el desarrollo de SIG. Su principal misión es que a partir de la base de la plataforma, que cuenta con más de 50 funcionalidades SIG entre las que son estándares en un SIG y las de innovación propia de su equipo de desarrollo, personalizar la aplicación hacia un negocio específico de acuerdo a los requerimientos del cliente en cuestión(Fig. #7).

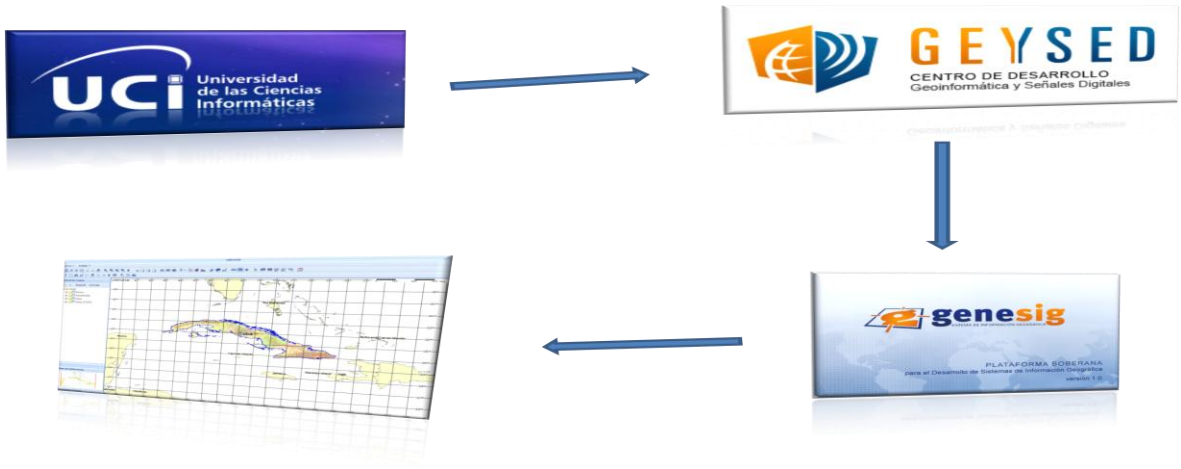


Fig. #7 Descripción del dominio del problema.

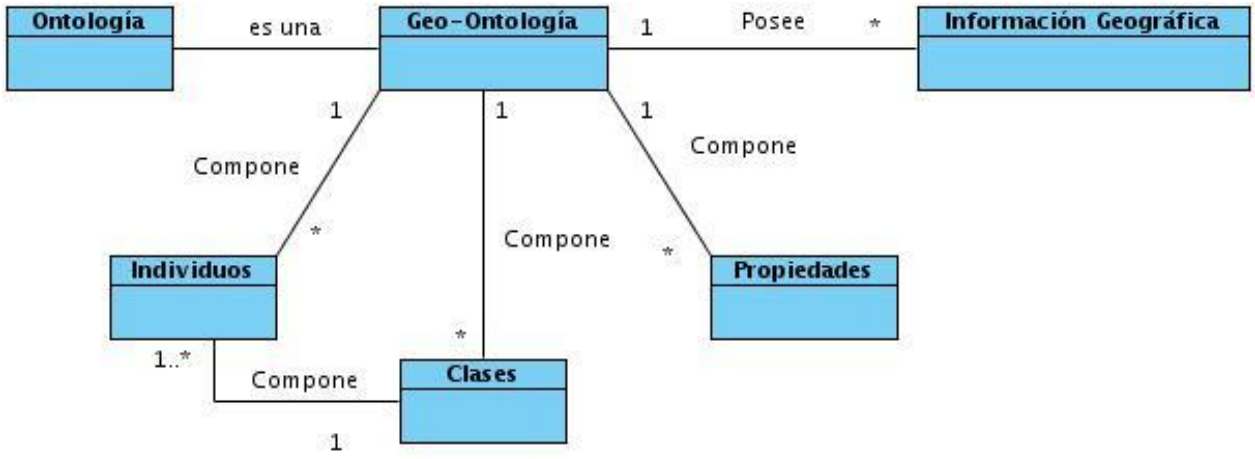


Fig. #8 Modelo del dominio.

1.3.2 Lenguajes de consultas a ontologías.

Los lenguajes de consultas a ontologías son muy diversos y su uso está dado según el tipo de implementación de las ontologías. Estos se dividen principalmente en dos grupos de lenguajes:

Grupo DL

Las lógicas descriptivas están muy relacionadas con el desarrollo de las ontologías tal como se usan en la actualidad en la web semántica. La lógica descriptiva se basa en representar el conocimiento utilizando por una terminología o vocabulario del dominio (TBOX) y por otra un conjunto de afirmaciones (ABOX).

El vocabulario consiste en "conceptos" y "roles". Los conceptos corresponden a conjuntos de elementos y los roles a relaciones binarias entre elementos. Existen conceptos y roles "atómico" y “complejo".

Por ejemplo, si tenemos como conceptos atómicos “Persona” y “Hembra” y como role atómico “tieneHijo”, podríamos tener como TBOX representando las relaciones familiares algo como lo siguiente:

Mujer ≡ Persona ∩ Hembra

Hombre ≡ Persona ∩ ¬ Mujer

Madre ≡ Mujer ∩ ∃ tieneHijo.Persona

Padre ≡ Hombre ∩ ∃ tieneHijo.Persona

Progenitor ≡ Madre ∪ Padre

Esposa ≡ Mujer ∩ tieneMarido.Hombre

Marido ≡ Hombre ∩ tieneEsposa.Mujer

La información recogida en la ABOX podría ser algo como:

tieneHijo(Juan, Miguel)

Padre(Juan)

Esposa(Juan, María)

Se pueden construir y existen razonadores que permiten razonar sobre las TBOX y ABOX, pudiendo determinar por ejemplo si el contenido de la TBOX es factible, o qué relaciones están incluidas en otras. Al trabajar sobre el ABOX, un razonador puede indicar que a partir de las afirmaciones existentes un determinado elemento es una instancia de un concepto y si las afirmaciones son consistentes con el modelo.

En resumen, el grupo de lenguajes DL donde es más utilizado es en la construcción de las ontologías, representación de expresividad terminológica de un dominio, de manera robusta, organizada y bien estructurada a través de formalismos lógicos. A partir de esto se puede obviar como posible variante de grupo de lenguajes para la posible solución de la investigación, que pretende utilizar un lenguaje entendible, con sintaxis sencilla y robusta, además que la curva de aprendizaje sea fuerte en este sentido.

Grupo QL.

El grupo QL se caracteriza por ser el más usado en los procesos de consultas a ontologías y sus implementaciones son más factibles para la recuperación de la información, sus sintaxis son similares al estándar SQL:

- **RQL** (Rodil-Garrido, 2006)**:** lenguaje tipado, que siguiendo un enfoque funcional, define un conjunto de consultas básicas e iteradores. Define algunas funciones que le sirven para obtener resultados básicos de la ontología, al mismo tiempo que auxilian en la construcción de consultas más complejas. Dichas funciones pueden ejecutarse independientemente.
  - ✓ Ejemplo de consulta:

SELECT $C1, $C2
FROM {$C1} creates {$C2}

- **RDQL** (Rodil-Garrido, 2006)**:** es una implementación de un lenguaje de consulta similar a SQL pero para RDF. Ha sido desarrollado por HP y es soportado por Jena2. De hecho, también podemos decir que RDQL es el lenguaje de consulta para grafos RDF de Jena2. El hecho de que sea similar a SQL, resulta familiar a muchos usuarios.

  RDQL trata RDF como datos y provee consultas con patrones de sentencias y restricciones sobre un modelo RDF. Se diseñó para ser usado en scripts y para experimentación en lenguajes de modelación de información. El lenguaje está derivado de SquishQL, el cual a su vez está basado en RdfQL, un lenguaje de bases de datos escalables creadas para trabajar con Servicios Web semánticos.

  La cláusula SELECT define las variables que se requieren mostrar en el conjunto resultante. La cláusula WHERE define un patrón de sub-grafo en términos de variables y constantes. La cláusula AND introduce un filtro en las variables; solamente los resultados que pasan el filtro son incluidos en el conjunto resultante de la consulta. La cláusula USING define abreviaciones para espacios de nombre.

  - ✓ Ejemplo de consulta:
    SELECT ?x, ?y
    FROM <http://example.com/sample.rdf>
    WHERE (?x,<dc:name>,?y)USING dc for <http://www.dc.com#>

- **SeRQL** (Rodil-Garrido, 2006): es un lenguaje declarativo de consulta de instancias y esquemas RDF, que aprovecha todas las características del modelamiento RDF. Fue desarrollado por la empresa holandesa Aidministrator. SeRQL es soportado por el sistema Sesame.

  Una consulta SeRQL se construye típicamente con un conjunto de seis cláusulas. Estas son: SELECT, FROM, WHERE, LIMIT, OFFSET y USING NAMESPACE. Es fácil reconocer las primeras cinco cláusulas del SQL, pero su uso es diferente. Para las llamadas “construct queries”, las cláusulas son las mismas, con la excepción de la

primera, que sustituiremos por la cláusula CONSTRUCT. Por tanto, la primera cláusula será SELECT o CONSTRUCT.

Con SELECT, podemos indicar que valores obtendremos y en qué orden, en cambio, con CONSTRUCT, indicaremos que triples deseamos obtener. La cláusula FROM es opcional y determina la ruta de la expresión, es decir, define el camino o ruta de un grafo RDF, el cual es relevante para la consulta.

Hay que hacer notar que cuando la cláusula FROM no se incluye, se devuelven constantes especificadas en las cláusulas SELECT o CONSTRUCT. La cláusula WHERE es opcional y puede tener operadores lógicos. Las cláusulas LIMIT y OFFSET también son optativas. Pueden ser usadas separadas o conjuntamente en orden, para obtener un subconjunto de la consulta. Su uso es muy similar a las cláusulas LIMIT y OFFSET de SQL.

Finalmente, la cláusula USING NAMESPACE es también opcional y puede contener declaraciones de espacios de nombres.

- ✓ Ejemplo de consulta:

```
SELECT Country
FROM {Country} ex:population {Population}
WHERE Population < "1000000"^^xsd:positive Integer
USING NAMESPACE
ex = <http://example.org/things#>
```

- **Lenguaje Triple** (Rodil-Garrido, 2006)**:** se basa en consultas y reglas del lenguaje. El lenguaje es derivado de F-Logic. Los Triples RDF (S, P, O) son representados en F-Logic como expresiones S[P->0] y puede ser jerarquizado. Por ejemplo, la expresión S[P1->01, P2->02 [P3->03]] corresponde a tres triples RDF, (S, P1, 01), (S, P2, 02) y (02, P3, 03).

  Triple no distingue entre consultas y reglas. Triple no codifica una semántica fija de RDF.

- **N3** (Rodil-Garrido, 2006)**:** provee texto basado en sintaxis RDF. Por lo tanto, el modelo de datos de N3 se forma con el modelo de datos RDF. Además, N3 define reglas, las cuales usan una sintaxis especial, por ejemplo:

  ?y rdfs:label "foo" => ?y a :Query Result.

- **Versa** (Rodil-Garrido, 2006)**:** está diseñado para ser integrado en otros lenguajes de programación. Versa permite tratar y consultar nodos y arcos en RDF. Utiliza una sintaxis simple y expresiva. Utiliza constructores de los espacios de nombres de XML.

- **SPARQL:** Es un lenguaje de consulta capaz de obtener información desde grafos RDF. Es la propuesta de estándar del W3C. Al igual que SeRQL, es soportado por Sesame y Jena. Proporciona facilidades para (Rodil-Garrido, 2006):

  - ✓ Extraer información en forma de URI's, nodos blancos y literales.
  - ✓ Extraer sub-grafos RDF.
  - ✓ Construir nuevos grafos RDF basados en los grafos incluidos en la consulta.

  Actualmente RDF se utiliza en la representación de información personal, redes sociales, metadatos de recursos web como video, sonido e imágenes, así como para proporcionar un medio de integración entre fuentes de información heterogéneas, teniendo como objetivo principal, facilitar la identificación entre recursos en la web. Entonces con un lenguaje de consulta RDF, como SPARQL, los desarrolladores y usuarios finales se encontrarán con una gran cantidad de información (recursos reconocibles) teniendo la facilidad para representar y utilizar los resultados obtenidos de manera más eficiente, sustentándose así su utilidad en la necesidad de recuperar y organizar la información de diferentes fuentes. Las tripletas RDF proveen un mecanismo de persistente almacenamiento y de acceso a grafos RDF, porque los modelos en forma de tripletas garantizan flexibilidad, son más simples y están basados en estándares. Ofrece a los desarrolladores y usuarios finales un camino para presentar y utilizar los resultados de búsquedas a través de una gran variedad de información como puede ser datos personales, redes sociales y metadatos sobre recursos digitales, como música e imágenes. SPARQL también proporciona un camino de integración sobre recursos diferentes. (Valencia-Castillo)

Similarmente a los lenguajes de acceso a datos en los Sistemas de Bases de Datos (SBD), SPARQL es conveniente para la utilización de forma local y remota. SPARQL está basado en patrones de grafos. Los patrones más escuetos son las tripletas, los cuales son similares a las tripletas RDF pero con la ventaja de contener una variable en el sujeto, predicado u objeto.

SPARQL tiene especificaciones que explican diferentes partes de su funcionalidad; en general, consiste en un lenguaje de consulta, un formato para las respuestas, y un medio para el transporte de consultas y respuestas (Valencia-Castillo):

- ✓ SPARQL Query Language: Componente principal de SPARQL. Explica la sintaxis para la composición de sentencias y su concordancia.
- ✓ SPARQL Protocol for RDF: Formato utilizado para la devolución de los resultados de las búsquedas (queries SELECT o ASK), a partir de un esquema XML.
- ✓ SPARQL Query Results XML Format: Describe el acceso remoto de datos y la transmisión de consultas de los clientes a los procesadores. Utiliza WSDL para definir protocolos remotos para la consulta de bases de datos basadas en RDF.

El lenguaje de consultas SPARQL está basado en comparación de patrones gráficos. Los patrones gráficos contienen patrones triples. Los patrones triples son como las tripletas RDF, pero con la opción de una variable consulta en lugar de un término RDF en las posiciones del sujeto, predicado u objeto. Combinando los patrones triples obtenemos un patrón grafico básico, donde es necesaria una comparación exacta entre gráficos (Valencia-Castillo).

La función de la palabra clave PREFIX es equivalente a la declaración de espacios de nombre (namespaces) en XML, es decir asocia una URI a una etiqueta, que se usará más adelante para describir el namespace. Pueden incluirse varias de estas etiquetas en una misma consulta. Todo comienzo de una consulta SPARQL queda marcada por la palabra clave SELECT, similar a su uso en SQL, sirve para definir los datos que deben ser devueltos en la respuesta. La palabra clave FROM identifica los datos sobre los que se ejecutará la consulta, es necesario indicar que una consulta puede incluir

varios FROM. La palabra clave WHERE indica el patrón sobre el que se filtrarán los tripletes del RDF (Valencia-Castillo).

La curva de aprendizaje de SPARQL es mayor respecto a sus similares, dadas las potencialidades de este en la integración con SBD y servicios web (WSDL), devolución de resultados de las búsquedas, así como la construcción de nuevos grafos RDF. Existe mayor cantidad de documentación publicada acerca de SPARQL, bien fundamentada y actualizada, por lo que se ha decidido que será el utilizado como parte de la solución a esta investigación.

Sintaxis básica de una consulta SPARQL	
Prologue (optional)	BASE <iri> PREFIX prefix: <iri> (repeatable)
Query Result forms (required, pick 1)	SELECT (DISTINCT)sequence of ?variable SELECT (DISTINCT)* DESCRIBE sequence of ?variable or <iri> DESCRIBE * CONSTRUCT { graph pattern } ASK
QueryDatasetSources (optional)	Add triples to the background graph (repeatable): FROM <iri> Add a named graph (repeatable): FROM NAMED <iri>
Graph Pattern (optional, required for ASK)	WHERE { graphpattern z}
QueryResultsOrdering (optional)	ORDER BY ...
QueryResultsSelection (optional)	LIMIT n, OFFSET m

Tabla #2: Sintaxis básica de SPARQL (Valencia-Castillo)**.**

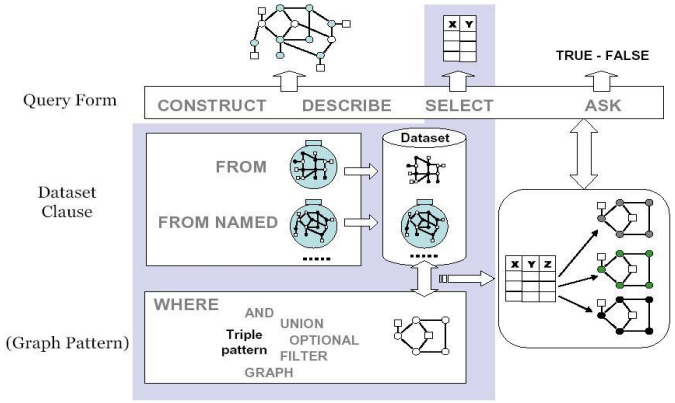


Fig. #9 Estructura General de una consulta Spar2L.

1.4 Conclusiones parciales.

- Durante el desarrollo de este capítulo, se trataron los principales conceptos asociados al dominio del problema, tales como ontología, estado del arte de las geo-ontologías como subconjunto de las ontologías, estructura y funcionamiento de un SIG, transformación de un SIG en SIGGO y funcionamiento de este. Este esbozo permitió entender y aprender cómo a través de geo-ontologías es posible representar de forma explícita la información semántica embebida en los metadatos geográficos.

- Se analizaron los principales editores de consultas a ontologías, tales como Twinkle, Sparqler y OntoEditor, y se realizó un análisis de los lenguajes de consultas existentes a ontologías, donde se seleccionó el lenguaje SparQL para el ejercicio práctico de la presente investigación debido a su potencia en la construcción de consultas y en la devolución de resultados de estas.

- Los vínculos teóricos mostraron cómo la utilización de las ontologías y sus derivaciones en el tratamiento de la información son aplicables tanto en Cuba como en el Mundo.

- El tratado acerca del desarrollo de las ontologías permitió comprobar la riqueza que encierran las mismas, así como las potencialidades con que cuentan, que son aplicables prácticamente a cualquier área de la Informática, sobre todo en la construcción de sistemas de información.

Capítulo #2 Herramientas a utilizar en la construcción de la solución. Selección y argumentación de la metodología de desarrollo de software a utilizar en el proceso.

2.1 Introducción.

Durante el proceso de desarrollo de este sistema informático realizó un estudio de las herramientas y metodologías de desarrollo que fueron utilizadas para el buen ejercicio del mismo, donde se hace necesario conocer las características de ambos, así como la especificación de las versiones con que estos cuentan. GeneSIG es un producto cuya principal característica es la representación y manipulación de información geo-referenciada, y el resultado de esta investigación consiste en un sistema capaz de prever la integración con GeneSIG, a través del manejo de la información semántica de las estructuras ontológicas correspondientes a las almacenadas en las BDE de GeneSIG. A continuación se desarrolla el anterior planteamiento.

2.2 Valoración de Herramientas

2.2.2 Netbeans

El IDE[4] y Plataforma NetBeans (Fig. #10) es una base modular y extensible usada como una estructura de integración para crear aplicaciones de gran envergadura, tanto web como de escritorio. Distintas empresas especializadas en desarrollo de software, suministran extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. Framework basado en asistentes (diálogos paso a paso).

Fig. #10 NetBeans IDE.

[4]Integrated Development Enviroment: Siglas en ingles de Entorno Integrado de Desarrollo.

NetBeans admítela creación de aplicaciones Web en PHP 5.x lo cual representa una ventaja para el desarrollo de GeneSIG, cuya estructura está programada del lado del servidor en PHP 5.x, y NetBeans además es capaz de reconocer las librerías Ext JS de JavaScript, con las que se han implementado las interfaces de usuario y las validaciones del lado del cliente. Además tiene soporte para AJAX donde agilizan la transmisión de datos en la Web.

Este IDE es muy potente y además para el desarrollo de GeneSIG no necesita hacer uso de librerías específicas del mismo, lo cual representa una ventaja en la portabilidad del producto GeneSIG hacia otros IDEs. Además cuenta con la potencialidad de ser software de código abierto, siguiendo la política de desarrollo de GeneSIG.

2.2.3 Jena Framework

Jena es un marco de trabajo desarrollado en java (y para java), para construir aplicaciones para la Web semántica. Provee un ambiente de programación para RDF, RDFS y OWL, incluyendo un motor de inferencia basado en reglas. Es un proyecto Open-Source y crece con la ayuda del programa de WS de “HP Labs”.

El framework de Jena incluye:

- ✓ Un API para RDF.
- ✓ Lectura y escritura de documentos en formato RDF/XML, N3, y N-Triples.
- ✓ Un API para OWL.
- ✓ Almacenamiento persistente y en memoria.
- ✓ RDQL, un lenguaje de consulta para RDF.

Fig. #11 Jena Framework

El corazón de Jena la API RDF, la cual soporta la creación, manipulación y consulta de grafos RDF. La API también soporta diferentes tecnologías de almacenamiento y plug-ins que permiten la lectura y escritura automática para diferentes lenguajes que los desarrolladores pueden usar para representar grafos RDF.

RDF API

Desde una perspectiva de API, hay dos formas distintas de ver un grafo RDF. En una vista centrada en afirmaciones, un grafo RDF es un conjunto de triples; donde cada triple identifica el nodo al comienzo del arco, el arco en sí, y el nodo al final del arco. Esta vista es conveniente para manipular grafos como un todo, tal como cuando se lee, escribe, o mezclan. En una vista centrada en frames (marcos), un grafo RDF es una colección de recursos, cada uno de los cuales tiene propiedades. Esta vista es análoga al paradigma de programación de orientación a objetos que tiene objetos con atributos. Algunos recursos RDF tienen un comportamiento implícito. Por ejemplo, los contenedores usualmente tienen métodos tales como insert, delete, tamaño, etc. Jena permite a los desarrolladores añadir el comportamiento apropiado a recursos de un tipo específico. La API actual de RDF API provee una primitiva de consulta, un método para extraer subconjuntos de todos los triples que un “objeto selector" selecciona desde un grafo. Una clase selector simple retorna todos los triples que calzan con un patrón dado.

Parser[5]

Los desarrolladores de Jena también mantienen ARP, un parser que lee RDF/XML, el lenguaje estándar para la representación de grafos RDF. Mientras que muchos parsers RDF/XML son codificados a mano, un compilador de compiladores (Javacc) genera el parser ARP desde la gramática RDF/XML. ARP usa un XML parser estándar como preprocesador léxico. El parser generado entonces procesa los símbolos que, en efecto, son eventos en SAX, la API estándar para XML.

Writers

[5] Analizador Sintáctico.

La sintaxis de RDF/XML es flexible, permite lo mismo que los grafos RDF a ser escritos en muchos formatos distintos. Aquí se introduce la noción de estilo. El writer RDF/XML básico de Jena toma ventaja de las características de RDF/XML que permiten expresiones más compactas y elegantes, pero puede escribir gráficos de tamaño arbitrario. Un Pretty-Writer, de forma distinta, toma una mayor ventaja de la sintaxis disponible para producir salida compacta.

Almacenamiento

Jena contiene tres implementaciones de la API: uno almacena los datos en memoria, otro almacena los datos en bases de datos relacionales, y un tercero usa la base de datos incrustada Berkeley DB, un Software open-source de Sleepycat. Un SPI permite introducir nuevos sistemas de almacenamiento fácilmente. La implementación de bases de datos relacionales usa cualquier base de datos que soporte JDBC. La configuración de tablas permite la especialización de consultas a una base de datos específica. La estructura de tablas de una base de datos es también configurable, una característica que permite realizar ajustes de desempeño para aplicaciones especificas. El almacenamiento en Berkeley DB, al igual que el almacenamiento relacional, es persistente.

2.2.4 Visual Paradigm for UML.

Visual Paradigm for UML es una poderosa herramienta CASE que utiliza UML para el modelado; es la herramienta por excelencia para ser utilizada en un ambiente de software libre. Permite crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue.

Visual Paradigm ofrece:

- Un entorno de creación de diagramas para UML 2.0.
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.

- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas.

Fig. #12 Visual Paradigm for UML.

2.2.5 Lenguaje de programación PHP.

PHP significa “PHP: Hypertext Preprocessor”. PHP es un lenguaje de script del lado del servidor para crear páginas Web dinámicas. Crear scripts PHP para realizar un seguimiento de las actividades de los visitantes en el sitio, enviar correos electrónicos a los clientes, que los usuarios tengan la opción subir archivos o imágenes al sitio, y manejar contenido dinámico utilizando una base de datos. Las posibilidades son infinitas. La mayoría de los sitios Web de redes sociales están escritos en lenguaje. PHP es un lenguaje poderoso y sencillo (Gómez).

Fig. #13 PHP.

¿Cuáles son los beneficios de PHP para esta aplicación?

PHP es un lenguaje de programación de código abierto, lo que representa que no se deben pagar miles de dólares por concepto de derechos de licencia. Lo más favorable es la sencillez de su instalación. La característica más admirable es que es fácil de aprender. PHP es utilizado por miles desarrolladores en todo el orbe y existen miles de sitios web en Internet que están desarrollados en PHP. Un ejemplo de esto es la plataforma GeneSIG, la cual maneja su

negocio a través de este lenguaje. Por estas potencialidades el autor de esta investigación optó por este lenguaje para sea uno de los que utilizará en la solución.

2.2.6 JavaScript.

Es un lenguaje de programación utilizado para crear pequeños programas delegados de efectuar acciones dentro del ambiente de una página web. Consiste en un lenguaje de programación del lado del cliente, ya que es el navegador web el que tolera la carga de procesamiento. Debido a su compatibilidad con la generalidad de los navegadores modernos, es el lenguaje de programación del lado del cliente que más se utiliza.

Fig.14 JavaScript.

Con JavaScript se pueden crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones JavaScript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador. Entre las acciones típicas que se pueden realizar en JavaScript están: por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo, por el otro, JavaScript permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que se pueden crear páginas interactivas con programas como calculadoras, agendas, o tablas decálculo. JavaScript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, complejas estructuras de datos, etc. (Angel Álvarez)

AJAX (Asynchronous JavaScript and XML) (Correa)**:** Es un conjunto de tecnologías dedicadas con carácter delimitado, que permite crear aplicaciones web más eficaces en la interacción con el usuario. AJAX unifica estas tecnologías:

- Estructura y presentación de la información basada en estándares mediante XHTML y CSS.

- Document Object Model (DOM) para interactuar dinámicamente con los datos.
- Intercambio y manipulación de datos usando XML and XSLT.
- Recuperación de datos asincrónica usando XML-HttpRequest y JavaScript.

AJAX permite que hoy día sea utilizado como una iniciativa más dentro del desarrollo de aplicaciones web como método más rápido y eficiente de navegación.

Ext JS (SlideShare)**:** Consiste de una excelente librería para la creación de interfaces de usuario, Ext JS agiliza mucho el trabajo a los desarrolladores, ya que en unas cuantas líneas de JavaScript se pueden crear interfaces realmente intuitivas. Brinda una gran cantidad de componentes para interfaces de usuarios complejas y permite construir aplicaciones de escritorio en la web.

Ventajas:

- Código reutilizable.
- Independiente o adaptable a frameworks diferentes.
- Orientada a la programación de interfaces tipos desktop en la web.
- El API es homogenizado independientemente del adaptador usado. Los controles siempre se verán igual.
- Soporte Comercial y Open Source.
- Una extensa comunidad de Usuarios.

Fig. #15 Ext JS.

2.2.7 Lenguaje de programación Java.

Lenguaje de programación de alto nivel, muy extendido en los últimos años, con el cual se pueden realizar prácticamente cualquier tipo de aplicación. La principal característica con la que cuenta es que es independiente de la plataforma sobre la cual trabaje, ya sea Windows, Linux o Mac OS X. Su portabilidad entre dichas plataformas se logra a través de la Máquina Virtual de Java que actúa como enlace entre el Sistema Operativo y el programa de Java posibilitando que este sea perfectamente entendible.

Fig. #16 Java.

La programación en lenguaje Java permite el desarrollo de aplicaciones bajo la arquitectura cliente-servidor, dentro de lo que se encuentra la llamada computación distribuida. Esta característica permite que en resultado de esta investigación se muestra implementación de una aplicación servidora en Java que maneje las librerías del razonador Jena y publique un servicio que será consumido desde otra aplicación web en PHP + JavaScript (Ext JS & AJAX), esta última ajustándose al diseño de la plataforma GeneSIG como herramienta para la misma.

El autor de esta investigación considera que la utilización de estas tecnologías garantiza la que la interacción usuario-aplicación en la solución será más intuitiva, logrando sencillez y fácil maniobrabilidad sobre la misma, además de que haciendo uso de las mismas se logra mejor integración con la interfaz de la plataforma GeneSIG.

2.3 Lenguaje Unificado de Modelado (UML)

El lenguaje unificado de modelado (UML) por sus siglas en inglés, es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo del software. UML entrega una forma de modelar unidades conceptuales como son los procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de Base de Datos y componentes de software reusables. (Patricio-Salinas, y otros)

UML ha nacido como un lenguaje, pero es mucho más que un lenguaje de programación. Aunque en su génesis se parece a C++ o a Java, en realidad se ha diseñado y construido un lenguaje que ha nacido con una madurez muy acentuada si se le compara, incluso, con los últimos desarrollos de HTML, Java y XML, los lenguajes por excelencia del mundo Internet. (Rumbaugh, y otros, 1998)

Fig. #17 Lenguaje Unificado de Modelado.

También se puede afirmar que UML es el único lenguaje de modelado estándar, que permite un consenso con la mayoría de los expertos. Hoy día existen variadas herramientas que permiten realizar los distintos diagramas propuestos por UML, efectuándolos de una forma eficaz, fácil y rápida, economizando tiempo y evadiendo errores que se puedan cometer en el diseño de estos.

Al ser GeneSIG un producto de gran envergadura, UML juega un papel muy importante en todo su proceso de desarrollo, por lo que la integración del resultado de esta investigación con ese producto necesita tener como guía todo el proceso de desarrollo de este.

Las principales características de UML son:

- Lenguaje unificado para la modelación de sistemas.
- Tecnología orientada a objetos.
- El cliente participa en todas las etapas del proyecto.
- Corrección de errores viables en todas las etapas.
- Aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor.

2.4 Proceso Unificado de Desarrollo Ágil (AUP)

Versión simplificada de Proceso Unificado de Desarrollo (RUP), fácil de entender y aplicable al desarrollo de software comercial, utilizando las tendencias y conceptos que aún permanecen fieles a RUP. Se ha tratado de mantener el AUP tan simple como sea posible, tanto en su enfoque y en su descripción. (Mazzoni, y otros)

Fig. #18 Proceso Unificado de Desarrollo Ágil.

AUP como también se le conoce en el mundo de la ingeniería de software (IS), se preocupa especialmente de la gestión de riesgos. Plantea que aquellos elementos con alto riesgo obtengan prioridad en el proceso de desarrollo y sean afrontados tempranamente durante dicho proceso. Para lograrlo, se crean y mantienen listas identificando los riesgos desde etapas iniciales del proyecto. Especialmente relevante en este sentido es el desarrollo de prototipos ejecutables durante la base de elaboración del producto, donde se demuestre la validez de la arquitectura para los requisitos clave del producto y que determinan los riesgos técnicos (García-Expósito, y otros, 2009).

Las cuatro fases existentes en RUP también emergen en AUP, de manera contigua y ultimando como hitos alcanzados claramente.

El proceso AUP establece un modelo más simple que el que aparece en RUP por lo que reúne en una única disciplina las disciplinas de Modelado de Negocio, Requisitos y Análisis y Diseño. El resto de disciplinas (Implementación, Pruebas, Despliegue, Gestión de Configuración, Gestión y Entorno) coinciden con las restantes de RUP (García-Expósito, y otros, 2009).

A diferencia de RUP, que cuenta con 9 disciplinas, AUP presenta sólo 7, de las cuales algunas son la unión de otras en RUP. (Ver Fig. #18)

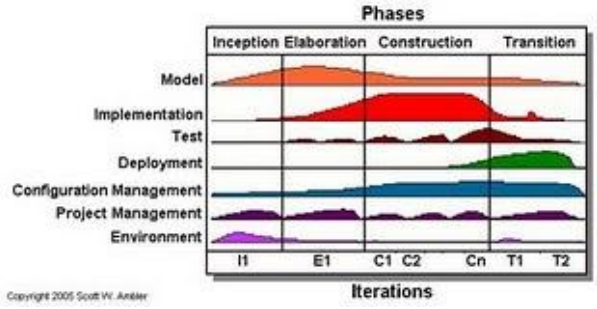


Fig. #19 Ciclo de vida de AUP.

Surge por una necesidad de acelerar el proceso de desarrollo de software para proyectos que sean pequeños. RUP Ágil es flexible, está orientada a equipos pequeños y presenta una significativa simplificación pero a pesar de ello no renuncia a las buenas prácticas ingenieriles para asegurar la calidad del producto. Propone los mismos roles, artefactos pero en una versión simplificada; es decir en RUP Ágil sólo se utilizan los artefactos que son imprescindibles y realmente necesarios para la realización del producto (García-Expósito, y otros, 2009).

Principios de RUP Ágil (García-Expósito, y otros, 2009)

- El personal necesita saber lo que está haciendo. La gente no va a leer la documentación de los procesos en detalle, sino que quieren una orientación de alto nivel y/o formación de vez en cuando. El producto AUP proporciona enlaces a muchos de los detalles si uno está interesado pero no obliga seguir los detalles.
- Simplicidad. Todo se describe concisamente usando unas páginas, no miles de páginas.
- Agilidad. El AUP se ajusta a los valores y principios de la Alianza Ágil.
- Centrarse en las actividades importantes. La atención se centra en las actividades que realmente cuentan.
- Herramienta de la independencia. Poder usar cualquier herramienta que desee utilizar con la AUP. Es recomendable utilizar herramientas que mejor se adapten para el trabajo, que a menudo son herramientas simples o incluso herramientas de código abierto.

- Querer adaptar este producto para satisfacer sus propias necesidades. El producto AUP es fácil de manejar a través de cualquier herramienta de edición de HTML. Usted no necesita comprar una herramienta especial, o tomar un curso, para adaptar el AUP.

El autor de esta investigación opta por la elección de esta metodología ya que la misma se encuentra diseñada para el trabajo de grupos pequeños de desarrollo, es independiente a las herramientas que se puedan utilizar, sobre todo las de código abierto, reforzando mejor los artefactos más importantes del sistema. Concuerda con la solución de esta investigación porque también está diseñada para proyecto donde la cantidad de casos de uso es chica. Por lo tanto fue la escogida porque se ajusta a las necesidades de esta investigación.

2.5 Conclusiones Parciales

Luego de un profundo análisis de las herramientas existentes se seleccionaron que intervienen en la construcción de la solución:

- NetBeans como Entorno Integrado de Desarrollo, que presenta grandes potencialidades para el desarrollo de aplicaciones, cuenta con soporte y ser un de lo más utilizados por los desarrolladores actuales.

- Java, PHP y JavaScript como lenguajes de programación para solucionar y lograr un mejor producto de software en su primera versión.

- Ajax, EXT JS y Jena como tecnologías de desarrollo que se integraron para lograr una estructura sólida del producto.

- Visual Paradigm for UML como herramienta CASE para la realización de los Diagramas de Casos de Uso del Sistema, Diagramas de Clases, Diagramas de Secuencia, Diagrama de Despliegue y Diagramas de Componentes de Componentes.

- UML como lenguaje de modelado para identificación de las Clases y modelación del problema.

- AUP como metodología de desarrollo, por su flexibilidad e independencia, generando sólo los artefactos necesarios para el desarrollo del producto.

Capítulo #3 Levantamiento de requisitos y Modelo del Sistema.

3.1 Introducción.

Durante el desarrollo de este capítulo se realiza el levantamiento de los requisitos funcionales y no funcionales del editor de consultas para lograr los objetivos planteados y a partir de la recopilación de dichos requisitos la construcción del Modelo de Casos de Uso del Sistemas.

3.2 Requisitos Funcionales.

Los requerimientos funcionales son condiciones o capacidades que el sistema debe cumplir. (Fase de inicio. Flujo de Trabajo de Requerimientos., 2010-2011) A continuación se listan los requerimientos funcionales:

- Requisito Funcional #1: El sistema de permitir seleccionar una ontología Ontología.
- Requisito Funcional #2: El sistema debe permitir crear consultas en lenguaje Sparql.
- Requisito Funcional #3: El sistema debe permitir ejecutar las consultas.
- Requisito Funcional #4: El sistema debe permitir chequear la sintaxis de Sparql.
- Requisito Funcional #5: El sistema debe permitir mostrar los resultados de las consultas.

3.3 Requisitos No funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. (Fase de inicio. Flujo de Trabajo de Requerimientos., 2010-2011) A continuación se listan los requerimientos no funcionales que el editor de consultas debe tener:

Requisitos de usabilidad:

- El sistema deberá ser utilizado por especialistas que posean al menos conocimientos básicos del lenguaje de consultas a ontologías SparQL.

- Debe contar con una interfaz amigable, con los colores básicos de la interfaz de la plataforma GeneSIG.

Requisitos de interfaz externa:

- El sistema debe tener una apariencia profesional y un diseño gráfico sencillo, con la utilización de las tonalidades de los colores azul celeste, blanco y gris fundamentalmente pues son los colores representativos de la plataforma GeneSIG.

Requisitos de operatividad y portabilidad:

- La aplicación debe ser compatible con los Sistemas Operativos: Windows 2000 NT, Windows XP y GNU/Linux.

Requerimientos de software:

- Las estaciones de trabajo que utilizarán el sistema deberán tener instalado:
  - Microsoft Windows ó GNU/Linux en cualquier distribución.
  - Navegador Web que cumpla con los stándares W3C(Mozilla Firefox, Opera, etc).
  - Contará con un nodo servidor con Apache 2 Server donde se alojará el núcleo OntoClient, y en otro nodo se alojará el núcleo OntoCore-Server, que necesita JVM 6 para poder ejecutarse.

Requerimientos de hardware:

- Las computadoras que utilizarán el software a desarrollar deberán tener 256 MB de Memoria tipo RAM como mínimo.

3.4 Descripción del sistema propuesto

La propuesta de solución es un sistema que permita al usuario crear y ejecutar consultas a geo-ontologías y devolver un resultado. Este sistema se desplegará cómo una herramienta para los desarrolladores de la plataforma GeneSIG, y estará compuesto por una aplicación Web y un servidor de servicios Web ontológicos, este mismo permitirá publicar la geo-ontología a la cual

se consultará. Permitirá además guardar la consulta en un fichero y a su vez cargar un fichero de consultas.

3.4.1 Descripción del actor.

Los actores del sistema son descripciones abstractas de las entidades externas a este; son subsistemas o clases que interactúan directamente con el sistema. Un actor participa en un caso de uso o en un conjunto vinculado de casos de uso para lograr un propósito global. Un actor determina las interacciones que los usuarios exteriores pueden tener sobre el sistema. En la siguiente tabla se enuncia la descripción del actor este sistema:

Actor	**Descripción**
Usuario	Persona que interactúa con el sistema para crear, ejecutar y guardar las consultas realizadas a las geo-ontologías, así como cargar un fichero de consultas.

Tabla #3: Descripción del actor de Sistema.

3.4.2 Casos de Uso del Sistema

Seguidamente se muestra el Diagrama de Casos de uso del Sistema (ver Fig. #10) donde se encuentra representado el actor que interactúa con el sistema a través de los casos de uso del sistema, seguido de la descripción textual de cada uno de estos.

3.4.2.1 Diagrama de Casos de Uso del Sistema.

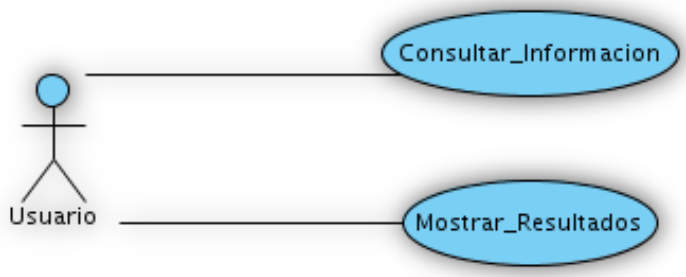


Fig. #20: Diagrama de Casos de Uso del Sistema

3.4.2.2 Descripción textual de los casos de uso del sistema.

Caso de Uso "Consultar Información".			
Actor	Usuario		
Resumen	Este caso se inicia cuando el usuario elige la opción "Cargar Ontología"		
Precondición	La consulta debe estar creada previamente.		
Referencias	Requisito Funcional #1: Seleccionar Ontología. Requisito Funcional #2: Crear consultas. Requisito Funcional #3: Ejecutar consulta. Requisito Funcional #4: Chequear sintaxis.		
Prioridad	Crítico		
Flujo Normal de Eventos			
Acción del Actor		Respuesta del Sistema	
1	El usuario selecciona la opción "Seleccionar Ontología".	1.1	El sistema muestra al usuario un listado de ontologías disponibles para consultar.
2	El usuario selecciona la ontología.	2.1	El sistema carga la ontología seleccionada.
3	El usuario procede a construir la consulta, haciendo uso del área de texto, puede escribir el código manualmente o	3.1	El sistema visualiza el proceso de construcción de la consulta.

	haciendo uso de los botones de inserción de las sentencias básicas.		
4	El usuario pulsa el botón “Ejecutar Consulta”.	4.1	El sistema chequea la sintaxis y ejecuta la consulta.
Flujo alterno de eventos			
3		3.1	El sistema muestra errores sintácticos en la sentencia de la consulta.
4	El usuario corrige los errores y pulsa el botón “Ejecutar Consulta”.		

Caso de Uso “Mostrar Resultados”.			
Actor	Usuario		
Resumen	Este caso se inicia cuando el usuario elige la opción “Guardar Consulta”		
Precondición	La consulta debe estar creada previamente.		
Referencias	Requisito Funcional #3: Mostrar Resultados.		
Prioridad	Crítico		
Flujo Normal de Eventos			
Acción del Actor		Respuesta del Sistema	
1		1.1	El sistema muestra al usuario el resultado de la consulta ejecutada.

Tabla #6 Descripción textual Caso de Uso “Mostrar Resultados”.

3.5 Conclusiones Parciales.

En el presente capítulo se realizó el Modelo del Sistema, comenzando por el levantamiento de los requerimientos funcionales y los requerimientos no funcionales del sistema que constituye el resultado de la presente investigación.

El levantamiento de requerimientos constituye la senda fundamental para irrumpirse en el Modelo del Sistema. Luego de un análisis de estos requerimientos, se confeccionó el Diagrama de Casos de Uso del Sistema y la respectiva descripción textual de cada uno de los casos de uso, haciendo indudable la manera en la que queda estructurado el sistema a construir.

Capítulo #4 Construcción de la solución propuesta.

4.1 Introducción.

El presente capítulo estará centrado en el proceso de desarrollo del sistema propuesto. Para ello primeramente se da paso a la construcción del modelo del diseño. Además se hace la propuesta de diagramas de clases y de secuencia del diseño para cada uno de los casos de uso identificados, aplicando el patrón de diseño centrado en componentes. Estos diagramas serán la base del diagrama de despliegue y el modelo de implementación que también son representados ene este capítulo.

4.2 Modelo del diseño.

El modelo del diseño constituye un modelo de objetos que representa la elaboración física de los casos de uso, ajustándose a cómo los requisitos funcionales y no funcionales en conjunto a las otras limitaciones correspondientes al entorno de implementación, tienen impacto en el sistema, siendo la principal vía de acceso en la actividad de implementación.

La capa OntoClient está compuesta por una arquitectura basada en componentes, donde se evidencia el patrón GoF Command en las peticiones y respuestas por AJAX a la capa OntoCore-Server.

En la capa OntoCore-Server se manifiesta una arquitectura en 3 capas donde existen 3 paquetes, un paquete **common** donde están las clases **Loader.java** y **Sparql.java** que se encargan de interactuar con las entidades de las ontologías, un paquete **control**, donde está la clase **MainControl.java** que se comporta como un servicio web y maneja todas las funcionalidades principales de la aplicación y un paquete **ontocorecli** donde está el programa principal **Main.java** que publica una interfaz SOAP en WSDL como instancia única (patrón Singleton) de la clase servicio web MainControl.java.

A continuación se muestra los diagramas de Clases del Diseño de las dos aplicaciones que conforman el resultado de esta investigación, denominadas OntoCore-Server (Fig. #21), que constituye en un núcleo servidor de servicios web de ontologías y OntoClient (Fig. #22), que constituido por la interfaz de usuario y el cliente que consume dichos servicios.

4.2.1 Diagramas de clases del diseño.

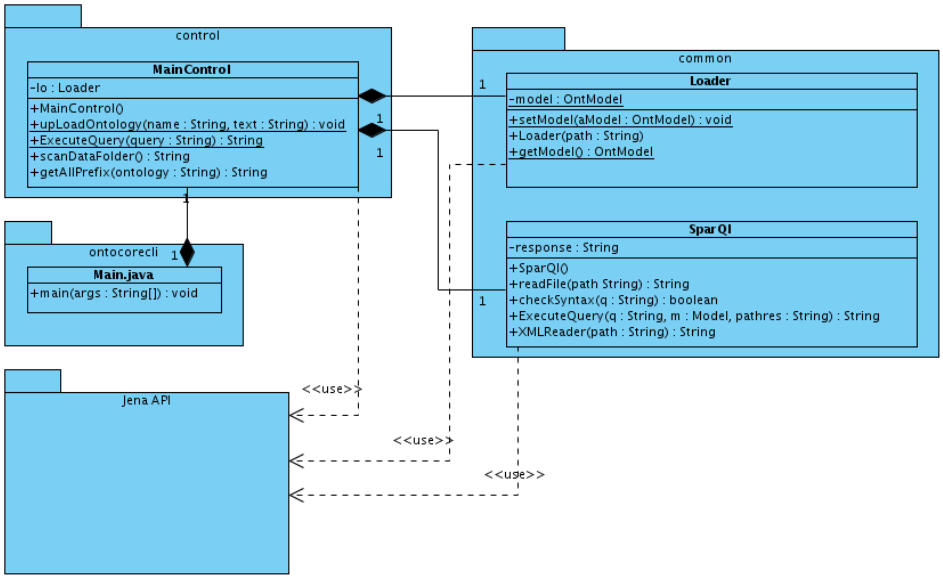


Fig. #21 Diagrama de clases del diseño del OntoCore-Server.

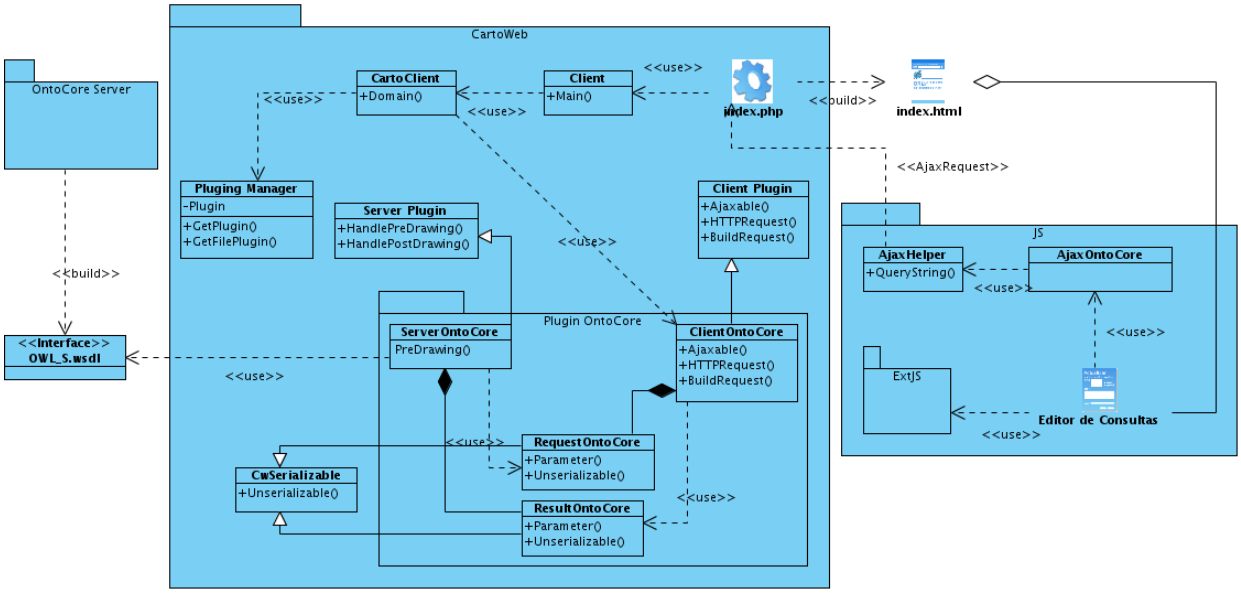


Fig. #22 Diagrama de clases del diseño del Onto-Client.

4.2.2 Diagramas de Secuencia del Diseño.

Los diagramas de secuencia muestran el intercambio entre distintos objetos en una situación determinada. Los objetos son instancias dela las clases. Los diagramas de secuencia se centran principalmente en el orden de las veces en que los mensajes son enviados.

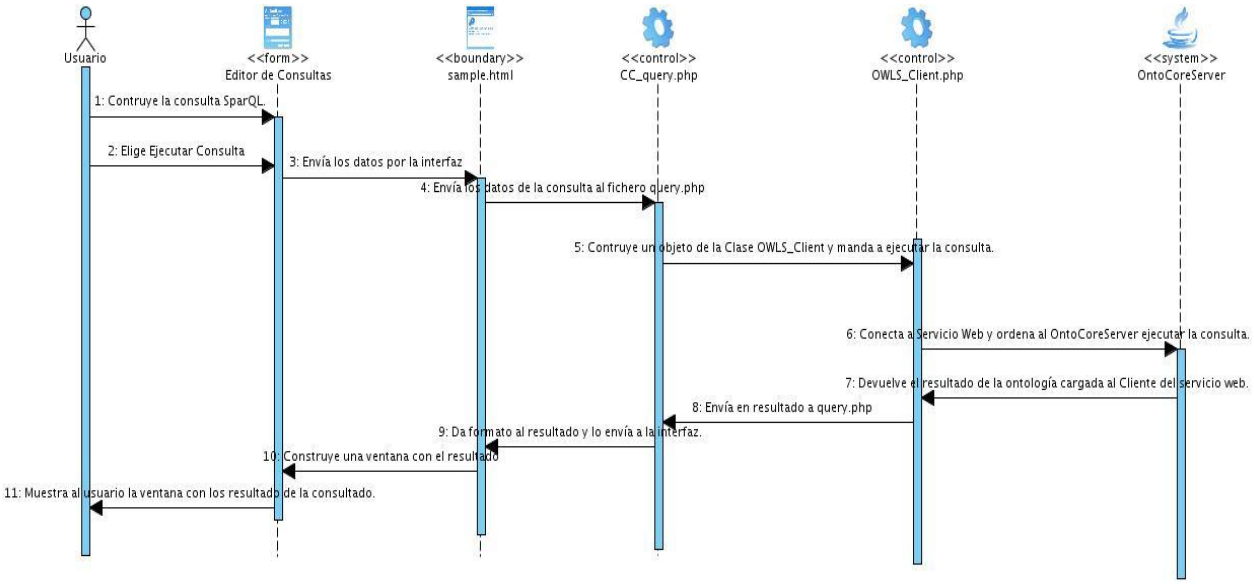


Fig. #23 Diagrama de Secuencia del Diseño.

La persistencia de los datos de los resultados de las consultas se encuentra dada a la utilización de ficheros de texto temporales y aleatorios, que se crean y destruyen en tiempo de ejecución, guardando su resultado en un objeto de tipo String antes de ser destruidos.

4.3 Modelo de despliegue

El propósito del modelo de despliegue es capturar la configuración de los elementos de procesamiento, y las conexiones entre estos elementos en el sistema. El modelo consiste en uno o más nodos, dispositivos, y conectores, entre estos. El modelo de despliegue también mapea procesos dentro de estos elementos de procesamiento, permitiendo la distribución del comportamiento a través de los nodos que son representados. A continuación se muestra el diagrama de despliegue modelado para la aplicación a desarrollar.

4.3.1 Diagrama de Despliegue

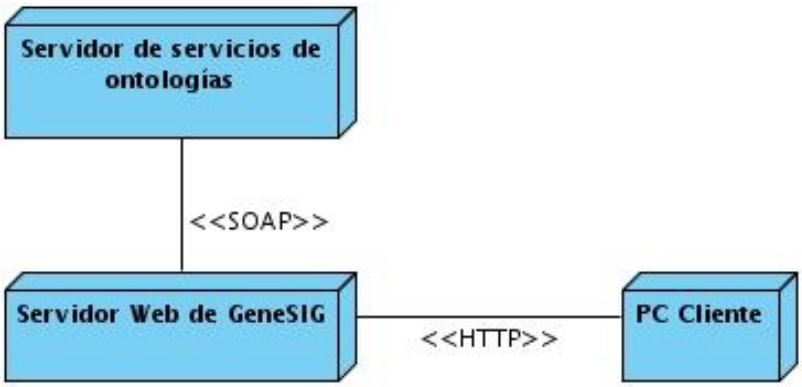


Fig. #24 Diagrama de despliegue.

4.4 Modelo de implementación.

El modelo de implementación está constituido por una colección de componentes, subsistemas de implementación, así como paquetes utilizados para agrupar elementos del modelo. Los componentes son la parte modular del sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces, y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos, estos pueden ser un fichero de código fuente, scripts, ficheros de código binario, ejecutables y similares. Los diagramas de componentes son utilizados para modelar la vista estática del sistema, mostrando la organización y las dependencias lógicas entre los componentes.

4.4.1 Diagramas de componentes

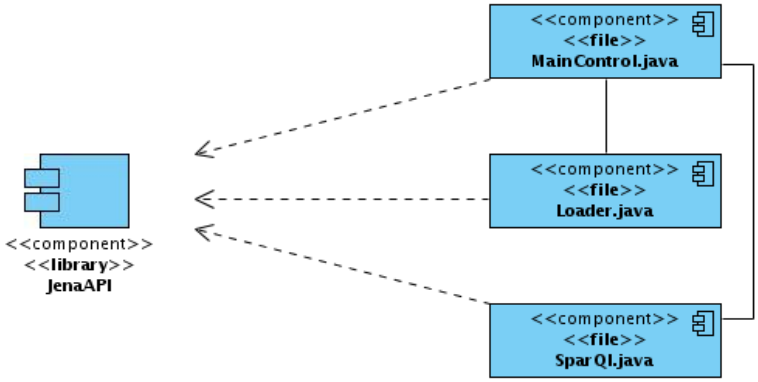


Fig. #25 Diagrama de componentes OntoCore Server.

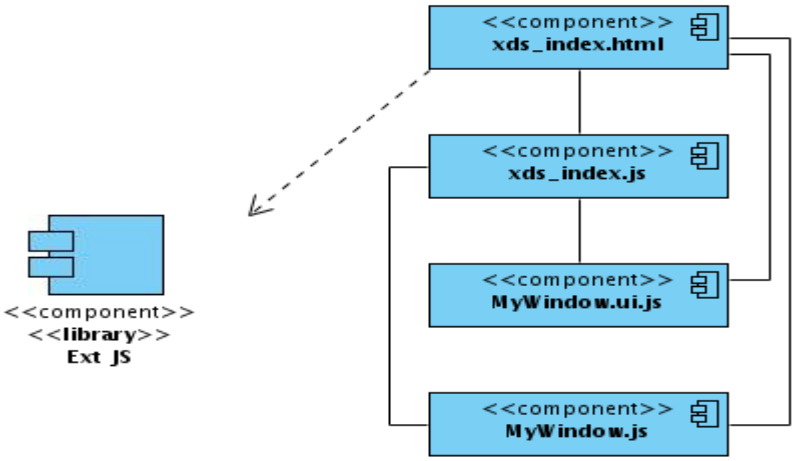


Fig. #26 Diagrama de Componentes OntoClient.

4.5 Conclusiones Parciales.

En el presente capítulo se desarrollaron las descripciones de la construcción de la solución propuesta. Se presentaron los Diagramas de Clases del Diseño de las aplicaciones OntoClient y OntoCore-Server, cuya integración da solución al problema planteado y el diagrama de secuencia del Editor de consultas. También se elaboraron los Modelos de Despliegue e Implementación del sistema.

Capítulo #5 Casos de pruebas y validación de la solución propuesta.

5.1 Introducción.

La realización de casos de pruebas es un de las fases imprescindibles para cotejar la calidad y correcto funcionamiento de un software. La prueba es el proceso de ejecución de una aplicación con el propósito de comprobar que el producto satisfaga los requerimientos enunciados y que el mismo tenga el comportamiento esperado. Para que las pruebas sean exitosas se hace necesario confeccionar casos de pruebas que tengan probabilidades de descubrir los errores del sistema utilizando técnicas que rijan el proceso de la prueba. Para probar el sistema propuesto se utilizó la técnica de caja negra.

5.2 Pruebas de caja negra.

Las técnicas de caja negra o funcionalidades son las que se aplican a través de la interfaz de la aplicación de software, concibiendo como interfaz las entradas y salidas de dicha aplicación. Para ello no se hace necesario conocer la lógica del negocio que maneja la aplicación, sólo se concentra en lo que la funcionalidad debe realizar. Las pruebas de caja negra, también conocidas como pruebas de comportamiento, se basan en la especificación de la aplicación o componente a ser probado para la elaboración de los casos de pruebas.

5.2.1 Caso de Prueba 1.

a. Descripción general.

Para la elaboración de las pruebas haciendo uso de las técnicas de caja negra se elige el caso de Uso “Consultar Información” a través de la interfaz del Editor de Consultas (Fig. #27), que consiste en un una ventana que contiene un formulario con un área de texto donde se escriben las sentencias de las consultas, una barra de botones que son los Componentes SparQL para insertar fragmentos de código correctos sintácticamente, una barra de herramientas donde está el botón Ejecutar Consulta, un botón desplegable donde se seleccionan las plantillas del tipo de consulta a realizar (SELECT y ASK), un combo dinámico donde se encuentran los prefijos de la ontología cargada en ese instante y un botón para adicionar el prefijo seleccionado en el combo dinámico.

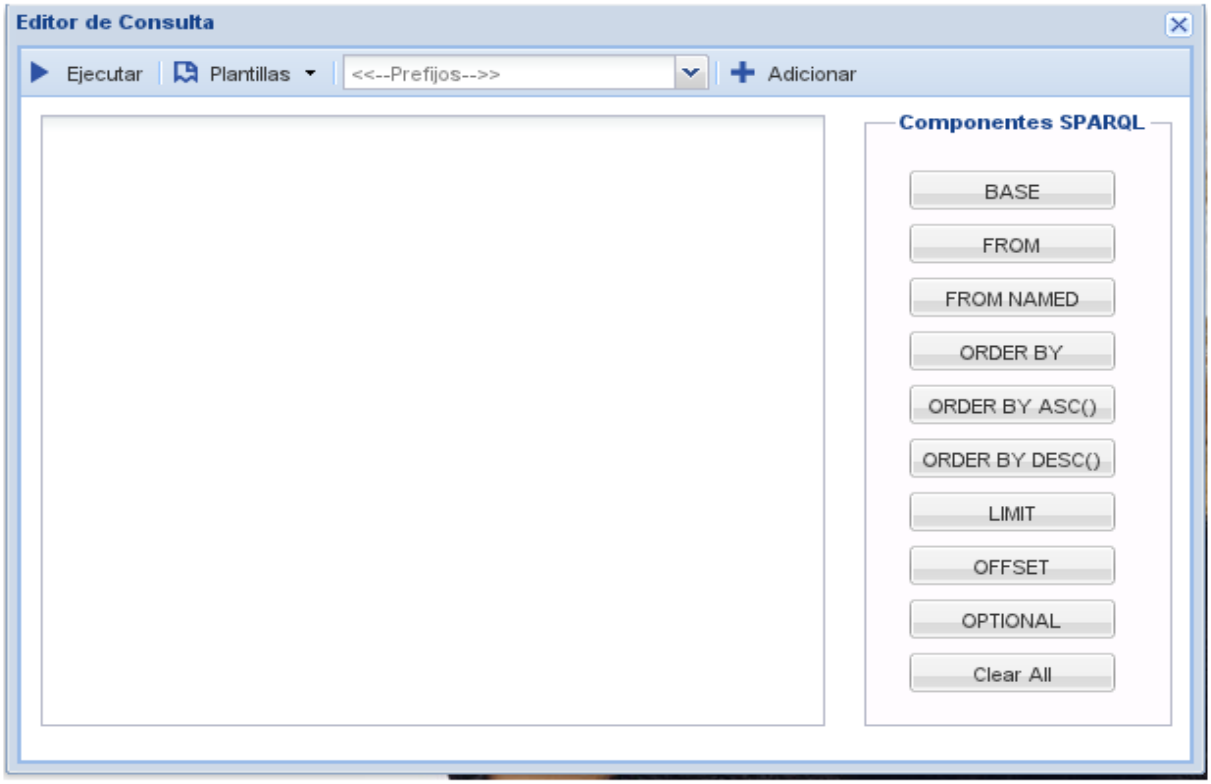


Fig. #27 Interfaz gráfica del Editor de consultas.

b. Condición de ejecución:

- Debe haber una ontología cargada previamente para poder consultarla.

c. Secciones de prueba del Caso de Uso "Consultar Información".

Nombre de la sección.	Escenario de la sección.	Descripción de la funcionalidad
SC 1: Consultar Información.	EC 1: Desplegar lista de prefijos.	El sistema muestra la lista de prefijos de la ontología cargada.
	EC 2: Seleccionar el prefijo y pulsar el	El sistema adiciona al área de texto el prefijo seleccionado.

	botón Adicionar.	
	EC 3: Seleccionar la plantilla del tipo de consulta a realizar.	El sistema muestra las plantillas y adiciona la seleccionada.
	EC 4: Insertar código en lenguaje SparQL.	El sistema inserta el código haciendo uso de la Barra de Componentes SparQL o el introducido por teclado.
	EC 5: Ejecutar la consulta.	Se da click en el botón Ejecutar y el sistema procede a la ejecución del código insertado en el área de texto, en caso de errores el sistema muestra el respectivo mensaje.

d. Descripción de las variables.

No.	Nombre del campo	Clasificación	Valor Nulo	Descripción
1	Editor	Área de texto	Sí	Se escribe el código SparQL a consultar.
2	Selector de Prefijos	Listado desplegable	No	Se debe seleccionar un prefijo de la lista.
3	Selector de Plantillas	Botón desplegable	No	Se debe seleccionar una plantilla de tipo de consulta.

e. Matriz de Datos.

Escenario	Variable 1	Variable 2	Variable 3	Respuesta del sistema	Resultado de la prueba	Flujo Central
EC 1.1	Editor	Selector de prefijos	Selector de plantillas	El sistema visualiza en el editor los prefijos y la plantillas seleccionados.		Crear un consulta SparQL.

5.2.2 Caso de Prueba 2.

Este es caso de prueba de aceptación con el cliente para validar el correcto funcionamiento de la aplicación:

Índice.	**CP2**
Descripción	El cliente realiza una consulta para conocer en la ontología Clasificacion_V3.owl qué algoritmo es el más preciso.
Valores establecidos	➢ ?Algoritmo ➢ ?Precision
Instrucciones	PREFIX a: <http://localhost/default#> PREFIX algor: <http://localhost/default#> SELECT DISTINCT ?Algoritmo ?Precision WHERE { ?Algoritmo algor:precision ?Precision . OPTIONAL { ?s algor:precision ?otraPrec } .

	Filter (?otraPrec < ?Precision) . Filter (bound(?otraPrec)) . } ORDER BY DESC(?Precision) LIMIT 1	
Resultados esperados.	Algoritmo	Precisión
	http://localhost/default#ID3	9

5.3 Conclusiones Parciales.

En el presente capítulo se procedió a la realización de Casos de Pruebas para la validación del producto de esta investigación. Se realizó un caso de prueba de caja negra al Caso de Uso "Consultar Información", estableciendo las correspondientes secciones con sus escenarios y la descripción de la funcionalidad, así como la descripción de las variables y la matriz de datos asociada a los mismos. También de expusieron los detalles de un caso de prueba de aceptación con el cliente para validación del correcto funcionamiento de la aplicación.

Conclusiones generales.

- La aplicación de la metodología de la investigación permitió determinar el estado del arte del objeto de estudio.
- Los procesos de manejo de información a través de la utilización de las ontologías en el marco nacional y foráneo, proporcionaron los elementos fundamentales para sostener teóricamente la investigación realizada.
- El estudio de las diversas herramientas, tecnologías y metodologías de desarrollo de software actuales, permitió determinar las factibles en el modelado e implementación del producto de esta investigación.
- El modelado del ciclo de vida del software agilizó la visión del sistema construido, garantizando la calidad de las actividades planificadas.
- Se obtuvo un producto capaz de realizar consultas en lenguaje SparQL a ontologías en formato ".owl".
- Se logró que el sistema cumpla con los requerimientos funcionales y no funcionales enunciados, siendo una herramienta capaz de devolver los resultados esperados.
- Se detallaron los artefactos propuestos por el Proceso Unificado de Desarrollo Ágil cómo metodología de desarrollo de software.

El producto de esta investigación es una herramienta cuyo aporte más significativo es capacitar a la plataforma GeneSIG de manejar metadatos ontológicos a través de consultas semánticas haciendo uso de una interfaz sencilla e intuitiva. Por tanto el objetivo general que desde la plataforma GeneSIG se puedan generar consultas semánticas a las ontologías como una componente más, y a su vez esta se retroalimente de dicha expresividad, convirtiéndose entonces en una plataforma para la creación de Sistemas de Información Geográfica Gobernados por Ontologías.

Bibliografía y referencias

Aguilar Ortiz, Roimel Rafael y Entenza Escoba, Yadira. 2009. *Sistema Web de Gestión de Préstamos externos de libros de literatura en la biblioteca de la Universidad de las Ciencias Informáticas.* La Habana : s.n., 2009.

Alcántara Caballano, José Luis. *El prisma.* [En línea] [Citado el: 20 de Octubre de 2009.] http://www.elprisma.com/apuntes/administracion_de_empresas/gestionempresarialrecursoshumanos/.

Angel Álvarez, Miguel. Desarrollo Web. *Desarrollo Web.* [En línea] [Citado el: 11 de febrero de 2011.] http://www.desarrolloweb.com/articulos/25.php.

Artidiello y Zarragoitia Alonso, Ileana. gestiopolis. [En línea] [Citado el: 15 de Octubre de 2009.] http://www.gestiopolis.com/Canales4/rrhh/formages.htm..

Berners-Lee. W3C. [En línea] http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide11-0.html.

Bosque-Sendra, J. 1992. *Sistemas de Información Geográfica.* Madrid : Rialp., 1992.

Braken, I. & Webster, C. 1992. *Information technology in geography and planning.* Londres & New York : Routledge, 1992.

Canós Darós, Lourdes. 2007. *Un algoritmo fuzzy para la selección de personal basado en agregación de competencias.* Universidad Politécnica de Valencia : s.n., 2007.

Canós Darós, Lourdes y Liern Carrión, Vicente. 2007. *Algunas aplicaciones de mateáticas fuzzy a la gestión de recursos humanos.* Universidad de Alicante : s.n., 2007.

Carrión, Juan. gestión del conocimiento. [En línea] [Citado el: 18 de Octubre de 2009.] http://www.gestiondelconocimiento.com/conceptos_conocimiento.htm..

Casinelli Esviza, Ignacio. rrhhweb. [En línea] [Citado el: 21 de Enero de 2010.] http://www.rrhhweb.com/ .

Cebrian, J. A. 1994. *GIS Concepts.* Cáceres : Departamento de Geografía y Ordenación del Territorio de la Universidad de Extremadura., 1994.

Correa, Leonardo. NovaWeb. *NovaWeb.* [En línea] [Citado el: 11 de febrero de 2011.] http://www.webnova.com.ar/articulo.php?recurso=357.

Crece_Negocios. 2008. *Crece Negocios.* [En línea] 18 de Diciembre de 2008. [Citado el: 15 de Noviembre de 2009.] http://www.crecenegocios.com/concepto-y-funciones-del-area-de-recursos-humanos.

DaeDalus. DaeDalus. [En línea] [Citado el: 14 de Noviembre de 2009.] http://www.daedalus.es/inteligencia-de-negocio/gestion-del-conocimiento/que-es-el-conocimiento/.

Definición.de. 2008. *definición.de.* [En línea] 2008. [Citado el: 2 de 25 de 2010.] http://definicion.de/competencia/.

—. 2008. Definicón.de. [En línea] 2008. [Citado el: 15 de Noviembre de 2009.] http://definicion.de/gestion/.

Delgado Morales, Dairai y Labrada Cruz, Amaury. 2008. *Software para la Selección del Personal por Gestión de Competencias utilizando Técnicas Matemáticas Multicriteriales.* La habana : s.n., 2008.

Diccionario_Ecológico. Diccionario_Ecológico. [En línea] [Citado el: 15 de 12 de 2009.] http://www.peruecologico.com.pe/glosario_g.htm..

1999*El proceso Unificado de Desarrollo del Software*1999

Escat Cortés, María. 2007. AreaRH. [En línea] 2007. [Citado el: 15 de noviembre de 2009.] http://www.arearh.com/rrhh/gestionrh.htm.

Espina, F. *El mundo de la Ontología.* Navarra : s.n.

EVA-UCI. *Eva UCI.* [En línea] [Citado el: 3 de 12 de 2009.] http://eva.uci.cu/file.php/46/Materiales_Complementarios/Sistemas_Basados_en_el_conocimiento./TEMA_01_SBC.pdf.

Experiencias en la gestión semántica de proyectos en la Universidad de las Ciencias Informáticas. **Román-Durán, M., Pérez-Olmos, Y. y Mederos-López-del-Castillo, J. 2010.** 2010, Revista de Inteligencia Empresarial., págs. 16-17.

Fase de inicio. Flujo de Trabajo de Requerimientos. **I., Conferencia 4: Ingeniería de Software. 2010-2011.** Universidad de las Ciencias Informáticas. : s.n., 2010-2011.

Fundacite-Mérida. Fundacite-Mérida. [En línea] [Citado el: 7 de Febrero de 2010.] http://sistemas.fsl.fundacite-merida.gob.ve/docman/view.php/84/420/compracion_de_framework_mvc.pdf.

García-Expósito, Raúl José y Rodriguez Luis, Zaily. 2009. *Intranet Petrosoft.* La habana : s.n., 2009.

Garea, E. y Vera, F. 2009. *ALINEAMIENTO DE ONTOLOGÍAS EN EL DOMINIO GEOESPACIAL.* 2009.

Garea-Llano, E, y otros. *SISTEMAS DE INFORMACIÓN GEOGRÁFICA COMO HERRAMIENTA PARA LA INTEGRACION E INTERPRETACIÓN SEMÁNTICA DE LA INFORMACIÓN ESPACIAL.* Habana : s.n.

Garea-Llano, E. y Gil-Rodríguez, J. L. Febrero 2007. *Los Sistemas de Información Geográfica Gobernados por Ontologías como Herramienta para la Interpretación Semántica de la Información Espacial y su integración a la IDERC.* La Habana, Cuba : V Congreso Internacional de Geomática, Febrero 2007.

Gómez, Hugo. http://www.articulo.org. [En línea] [Citado el: 10 de Febrero de 2011.] http://www.articulo.org/articulo/15539/introduccion_a_php.html.

Gracia Aguila, Adrián. 2009. *Análisis y diseño de un sistema automatizado para el control de los recursos humanos en los polos productivos de la facultad 9.* La Habana : s.n., 2009.

Gruber, T. 1993. *A Translation Approach to Portable Ontology Specifications.* 1993.

Gruber, T. R. 2001. What is an Ontology? *http://www-ksl.standford.edu/kst/what-is-an-ontology.html.* [En línea] 2001.

Gruber, T. 1993. *Toward Principles for the Design of Ontologies Used for Knowledge Sharing.* Standford University, CA. : s.n., 1993.

GSIG. 2008. *Presentación Oficial de la Plataforma GeneSIG.* Habana, Cuba : s.n., 2008.

Guarino, N. and Giaretta, P. 1995. *Ontologies and knowledge bases: towards a terminological clarification.* Trento, Roma. : Laboratory for Applied Ontology., 1995.

Guarino, N. 1998. *Formal ontology and information systems.* Trento, Roma. : Laboratory for Applied Ontology (LOA)., 1998.

Gutiérrez, Javier J. 2008. www.lsi.us.es. [En línea] 2008. [Citado el: 9 de Febrero de 2009.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

Herman, Ivan. W3C. [En línea] http://www.w3.org/2007/Talks/0423-Stavanger-IH.

Hernández Cabrera, ING. Jose Luis. 2006. Gestiopolis. [En línea] Junio de 2006. [Citado el: 15 de Octubre de 2009.] http://www.gestiopolis.com/canales7/rrhh/historia-y-desafios-de-la-gestion-de-los-recursos-humanos.htm..

Integración Ontológica de Datos Metadatos y Conocimiento en Sistemas de Información Geográfica como Herramienta para la Interpretación Semántica de la Información Espacial. **Garea-Llano, E., Oliva-Santos, R. y Costales-Llerandi, C. 2009.** 2009.

It-review. 2009. [En línea] 16 de 12 de 2009. [Citado el: 2010 de 4 de 2.] http://www.it-review.cl/?p=1359.

KDE.org, Documentacion. Documentacion KDE.org. [En línea] http://docs.kde.org/stable/es/kdesdk/umbrello/introduction.html.

Larin, R. y Garea, E. *INTEGRACIÓN SEMÁNTICA DE DATOS ESPACIALES CON SISTEMAS DE INFORMACIÓN GEOGRÁFICA.* Habana : s.n.

Larman, Craig. 1999. *UML y Patrones.* México : s.n., 1999.

Letelier, Patricio y Penadés, Msc Carmen. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* Universidad Politécnica de Valencia : s.n.

Lutz, M. y Klien, E. *Ontology-Based Retrieval of Geographic Information.* s.l. : Int. Journal of Geographical Information Science (IJGIS). págs. 233-260.

Martínez-Terrero, Ing. Alexis y Garea-Llano, Dr. C. Eduardo. octubre 2010. *Métodos de recuperación semántica de datos espaciales en formato vectorial.* Ciudad Habana : CENATAV, octubre 2010.

Mazzoni, Diego, Magliano, Román y Miretti, Andrés. *Utni-soo.* [En línea] [Citado el: 18 de enero de 2010.] http://sites.google.com/site/utnisoo/desarrollo-metodologias/Practico-Metodologia-2008---AUP.

NCGIA. 1990. *National Center for Geographical Information and Analysis, vol.1.* Santa Bárbara, Universidad de California : s.n., 1990.

Oliva-Santos, Rafael. 2009. *Anotaciones Semánticas de Datos Geográficos.* Ciudad Habana : Univesidad de la Habana, 2009.

OWL, Descripción de la ontología. **Gómez-López, A.** Recuperación de Información OWL.

Patricio-Salinas, Caro y Histchfeld- K., Nancy. DCC. [En línea] [Citado el: 1 de febrero de 2010.] http://www.dcc.uchile.cl/~psalinas/uml/introduccion.html.

Portal-UCI. La Produccion en la UCI. [En línea] [Citado el: 18 de Octubre de 2009.] http://www.uci.cu/?q=node/46.

Potencier, Fabien y Zaninotto, François. 2008. *Symfony, la guía definitiva.* 2008.

Reynoso, Carlos y Kicillof, Nicolás. 2004. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Universidad de Buenos Aires : s.n., 2004.

Rodil-Garrido, A. 2006. Universitat Oberta de Catalunya : s.n., 2006.

Rumbaugh, Jacobson y Booch. *El lenguaje Unificado de Modelado. Manual de Referencia.*

Rumbaugh, James, Jacobson, Ivar y Booch, Grady. 1998. *El lenguaje unificado de modelado.* 1998.

Sánchez García, Alberto y Puig Pinto, Jorge Carlos. 2009. *Sistema para la Gestión de la Información de Laboratorios de la Dirección de Calidad del Centro de Ingeniería Genética y Biotecnología: Implementación del Módulo Análisis Químico.* La habana : s.n., 2009.

SARABIA-LÓPEZ, GERARDO. Diciembre 2008. *BÚSQUEDA Y PONDERACIÓN DE INFORMACIÓN CONTENIDA EN BASES DE DATOS ESPACIALES, UTILIZANDO JERARQUÍAS.* México D.F. : s.n., Diciembre 2008.

Slideshare. Slideshare. [En línea] [Citado el: 8 de Febrero de 2009.] http://www.slideshare.net/alexmerono/sistemas-gestores-de-bases-de-datos.

SlideShare. SlideShare Inc. *SlideShare Inc.* [En línea] [Citado el: 11 de febrero de 2011.] http://www.slideshare.net/almarag/ext-js-y-frameworks-javascript.

SolusS.A. 2007. *Guía de Usuario de Enterprise Architect 7.0.* [En línea] 2007. [Citado el: 28 de 4 de 2010.] http://www.sparxsystems.com.ar/download/ayuda/index.html?componentdiagram.htm.

Swartout, B., Patil, R. and Knight, K. 1997. *Toward distributed use of large-scale ontologies. .* s.l. : In AAAI-97 Spring Symposium Series on Ontological Engineering., 1997.

Valencia-Castillo, E. *Recuperación y organización de la información a través de RDF usando SPARQL.*

Vasconcelos, Karine F., y otros. *OntoEditor: a Web Tool for Manipulating Ontologies Stored in Database Servers.* Av. Aprigio Veloso, 882 Bodocongó, 58109-970 Campina Grande – Pb Brasil : Universidade Federal de Campina Grande; Departamento de Sistemas e Computação.

Vckovski, A., K. Brassel, and H. J. Schek. 1999. *Preface. Interoperating Geographic Information Systems.* Zurich, Switzerland. : 2nd International Conference, INTEROP'99,, 1999.

Villalobos. 2006. Ciber Aula. [En línea] 2006. [Citado el: 1 de Febrero de 2010.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

Web, Maestros del. http://www.maestrosdelweb.com/. [En línea]