



Instituto Politécnico Nacional
*Unidad Profesional Interdisciplinaria en
Ingeniería y Tecnologías Avanzadas*



*Módulo de consultas federadas geoespaciales en el contexto
de la Web de Linked Data para el triple store Apache
Marmotta*

Alumno: Oswaldo Emmanuel Páez Ortega

Asesores:

Nombre: Luis Manuel Vilches Blázquez

Procedencia: Centro de Investigación en Computación (CIC)

Nombre: Cyntia Eugenia Enríquez Ortiz

Procedencia: Unidad Profesional en Interdisciplinaria en Ingeniería y Tecnologías Avanzadas
(UPIITA)

Septiembre marzode 2019

Resumen:

El presente proyecto terminal propone diseñar, implementar y caracterizar un módulo de consultas federadas de datos geoespaciales para un *triple store* que no tenga implementado dicho módulo, en específico, a la plataforma Apache Marmotta cuya arquitectura y funcionamiento están basados en los estándares SPARQL y GeoSPARQL. Además, con el propósito de que los usuarios finales visualicen e interactúen con los resultados desplegados por el módulo de consultas federadas geoespaciales, se usará una aplicación Web para que los resultados recuperados de la Web de *Linked Data* puedan ser visualizados y explorados.

Palabras clave: Módulo, consultas federadas, *triple store*, datos geoespaciales, Apache Marmotta, *Linked Data*, SPARQL, GeoSPARQL, Web Semántica, aplicación web

Contenido

1. INTRODUCCIÓN	4
JUSTIFICACIÓN	4
2. PLANTEAMIENTO DEL PROBLEMA	5
3. PROPUESTA DE SOLUCIÓN	7
3.1. ALCANCES (RESULTADOS ESPERADOS)	10
4. OBJETIVO GENERAL	11
4.1. <i>Objetivos específicos</i>	11
5. ESTADO DEL ARTE.....	11
6. MARCO TEÓRICO	16
6.1. LINKED DATA	17
6.2. WEB SEMÁNTICA	17
6.3. RDF (RESOURCE DESCRIPTION FRAMEWORK)	18
6.4. URI (UNIFORM RESOURCE IDENTIFIER)	18
6.5. RDF TRIPLE STORE	18
6.6. SPARQL	19
6.7. <i>SPARQL ENDPOINT</i>	19
6.8. <i>GEOSPARQL</i>	20
6.9. ARQUITECTURA SOA.....	21
6.10. PROTOCOLO HTTP	22
6.11. <i>REST</i>	23
6.12. <i>JSON</i>	23
7. ANÁLISIS DEL SISTEMA.....	25
7.1. DESCRIPCIÓN GENERAL	25
7.1.1. <i>Perspectiva del producto</i>	25
7.1.2. <i>Características de los usuarios</i>	25
7.1.3. <i>Restricciones</i>	25
7.1.4. <i>Suposiciones y dependencias</i>	25
7.2. REQUISITOS ESPECÍFICOS	25
7.2.1. <i>Requerimientos funcionales</i>	25
7.2.2. <i>Requerimientos no funcionales</i>	29
7.3. INTERFACES DE HARDWARE.....	30
7.4. INTERFACES DE SOFTWARE	30
7.5. INTERFACES DE COMUNICACIÓN	30
7.6. REQUERIMIENTOS FUNCIONALES	30
7.6.1. <i>Requerimiento funcional 1</i>	30

7.6.2.	<i>Requisito funcional 2</i>	30
7.6.3.	<i>Requisito funcional 3</i>	30
7.6.4.	<i>Requisito funcional 4</i>	30
7.6.5.	<i>Requisito funcional 5</i>	30
7.6.6.	<i>Requisito funcional 6</i>	31
7.6.7.	<i>Requisito funcional 7</i>	31
7.6.8.	<i>Requisito funcional 8</i>	31
7.6.9.	<i>Requisito funcional 9</i>	31
7.6.10.	<i>Requisito funcional 10</i>	31
7.6.11.	<i>Requisito funcional 11</i>	31
7.6.12.	<i>Requisito funcional 12</i>	31
7.7.	REQUISITOS NO FUNCIONALES.....	31
7.7.1.	<i>Disponibilidad</i>	31
7.7.2.	<i>Usabilidad</i>	32
7.7.3.	<i>Interfaz de la aplicación</i>	32
8.	DISEÑO DEL SISTEMA.....	33
8.1.	CASO DE USO	33
8.2.	DIAGRAMAS DE ESTADO	33
8.2.1.	<i>Diagrama de estado para la aplicación web</i>	33
8.2.2.	<i>Diagrama de estado para módulo de consultas en Apache Marmotta</i>	34
8.3.	DIAGRAMAS DE SECUENCIA.....	35
8.3.1.	<i>Diagrama de secuencia para aplicación web</i>	35
8.3.2.	<i>Diagrama de secuencia para módulo de consultas federadas geoespaciales</i>	36
8.4.	SECUENCIA DE INTERFACES	37
	REFERENCIAS.....	38

1. Introducción

La creación de la Web, llevada a cabo por Tim Berners Lee, y la popularidad que alcanzó provocó que los usuarios se interesaran en aportar contenido de toda índole en poco tiempo sin prestar atención a desarrollar un conjunto de buenas prácticas las cuales sirvieran como referencia para los usuarios al momento de crear y subir contenido a la Web. Debido a esta omisión, la posibilidad de tener una web inteligente se volvería difícil de lograr, esto a consecuencia de que las computadoras no son capaces de interpretar ni de hacer inferencias en el contenido de la Web [1]. Sin embargo, se propuso una evolución que le permitiría a la Web tener un contexto y significado en el contenido que alberga en ella; es aquí donde surge la Web Semántica. Con esta propuesta se pretende que el contenido en Web pueda ser interpretada por las computadoras a nivel semántico [2]. A partir de este acontecimiento surgen servidores de triple store cuya información que almacenan son tripletes en documentos del tipo Marco de Descripción de Recursos (RDF, por sus siglas en inglés) que describen entidades y relaciones en la Web Semántica a través de grafos y a su vez, surgen las plataformas de *Linked Data* (LDP, por sus siglas en inglés) las cuales son herramientas que son capaces de manipular dichas entidades y las relaciones existentes entre ellas. Las *triple stores* se basan en SPARQL [3] que es el lenguaje estandarizado de consultas para bases de datos de tipo RDF. Este concepto también es utilizado en el dominio de la Web Semántica para datos geoespaciales donde para la realización de consultas se utiliza el estándar GeoSPARQL [4]. Este estándar, en conjunto con tecnologías propias de la Web Semántica, ha sido aplicado en problemas de logística, hidrología, turismo, entre otros [5]. Estos ejemplos, con frecuencia, presentan propuestas donde se realizan consultas únicamente a un triple store y éste se encarga de devolver la información almacenada con características geoespaciales. No obstante, existen algunas propuestas donde se han realizado algunos ejemplos ad hoc de consultas federadas en el ámbito de los datos geoespaciales [6]. Sin embargo, el estado del arte actual presenta una importante limitación, ya que existen *triple stores* que no permiten la realización de consultas federadas a través de múltiples *triple stores* que presenten de información geoespacial (conforme a GeoSPARQL) en el contexto de la nube de *Linked Data*.

Justificación

Existen diversas empresas y organizaciones que se encargan de desarrollar herramientas para la Web Semántica y *Linked Data* para la manipulación y almacenamiento de datos semánticos. Una de estas organizaciones es la organización sin fines de lucro *Apache Software Foundation* (ASF) la cual ofrece diversas herramientas para diferentes necesidades en cuanto a software se refiere. Ejemplos de estas herramientas son servidores Web, *frameworks*, bases de datos, entre otras. Para el mundo del *Linked Data* y Web Semántica, ASF también tiene su plataforma y se le conoce como Apache Marmotta. La plataforma Apache Marmotta entre las diversas características que este software tiene, las 3 relevantes para este proyecto terminal son: es una LDP, un SPARQL *endpoint* y también es una base de datos para tripletes RDF, triple store. Para contextualizar a Apache Marmotta, una analogía con bases de datos SQL se muestra en la figura 1.

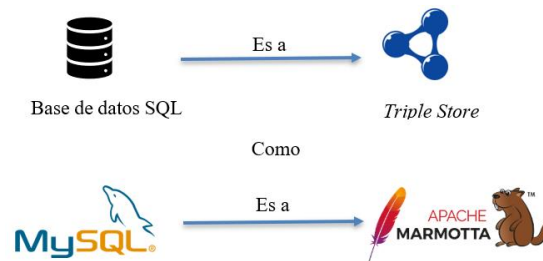


Fig. 1 Analogía entre MySQL y Apache Marmotta

A pesar de que es un software desarrollado por Apache, aún tiene características por incorporar al sistema. Ejemplo de estas características es que al estar basado en el estándar SPARQL 1.1, este documento estipula que hay 11 características por cumplir para que un triple store se considere completo conforme a esta versión del estándar (SPARQL 1.1). Apache Marmotta carece de 2 características, consultas federadas y regímenes de vinculación; las consultas federadas será la característica que será implementada en el presente proyecto terminal [7] con el fin de ofrecer una herramienta *open source* a desarrolladores e investigadores que usen la Web Semántica en sus trabajos e investigaciones.

2. Planteamiento del problema

Se ha demostrado que aproximadamente el 80% de los datos tienen relación con una ubicación geográfica [8]. Esta es una de las razones por la que las herramientas que traten con datos geoespaciales estén en constante actualización. El caso de la implementación del *Linked Data*, no es la excepción. Investigadores, empresas y organizaciones gubernamentales usan la nube del *Linked Data* para el estudio y administración de su información [9]. Actualmente existen diversos motores de *triple store* que no son capaces de hacer consultas a diversas fuentes de información geográfica a la vez, es decir, consultas federadas a *triple stores* de datos geoespaciales conforme a los estándares SPARQL 1.1 y GeoSPARQL.

Adicionalmente, otro aspecto a considerar para justificar el desarrollo del módulo propuesto en este proyecto es que las bases de datos no solo son del tipo distribuidas, sino que también existen aquellas que están bajo el esquema de bases de datos federadas. En el escenario de la Web Semántica, a este tipo de bases de datos se les denomina *federated triple store*. Con esta característica también se involucra el hecho de que los servidores estén en diferentes localizaciones geográficas, por lo que si no se realizan consultas federada a una *triple store* federada, se estaría obteniendo una porción de información. Por el contrario, si la consulta realizada es federada, se obtendría una respuesta completa que agrupa los resultados de las múltiples *triple store*. Además, considerando la característica de que la información proviene de distintos proveedores de información, el desarrollo de este módulo podría permitir que los expertos en las múltiples disciplinas que tiene asociada la información geográfica [6] puedan abordar sus problemáticas con una nueva herramienta que permita integrar y enriquecer sus análisis y estudio.

Ante este escenario, desarrollar e implementar un módulo que solucione dicha problemática sería clave para que los motores de *triple store* puedan ser usados de una mejor manera y obtener

mejores resultados en las búsquedas. En el caso particular de este proyecto, el trabajo se realizará sobre Apache Marmotta. Tal y como se indica en la figura 2, entre los elementos que contiene este *triple store*, no se encuentra un módulo de consultas federadas de datos geoespaciales.



Fig. 1 Elementos que contiene Apache Marmotta a considerar en el proyecto

La plataforma funciona con los estándares SPARQL y GeoSPARQL para establecer la manipulación correcta de datos en la nube del *Linked Data* y datos geoespaciales respectivamente. Esto quiere decir que tales características ya existen, pero lo que aún no se ha implementado es la capacidad de hacer consultas federadas. Como prueba, la figura 3 muestra una captura de pantalla del sitio oficial de Apache Marmotta¹. encerrado en un rectángulo rojo, indicando que la plataforma aún no es capaz de hacer consultas federadas.

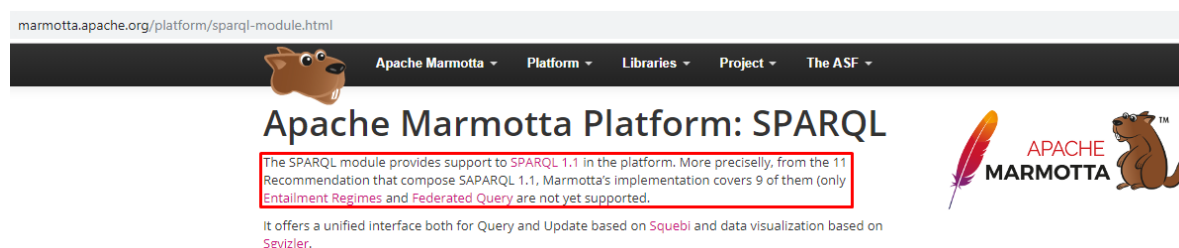


Fig. 2 Apache Marmotta no soporta consultas federadas.

Sin embargo, hay aspectos a cumplir para que este propósito sea realidad. El primer punto por tratar es que Apache Marmotta [10] está escrito en Java por lo que hay que comprender cómo es que está diseñado y construido. Las librerías, objetos y el paradigma de programación implementados son ejemplos de elementos a estudiar. De igual forma se debe de abordar los protocolos que son indispensables para que Apache Marmotta funcione.

Así mismo, puesto que se escribirá código y algoritmos, dicho código debe de ser eficiente y de fácil mantenimiento. Esto se irá mejorando con las múltiples pruebas descritas en escenario de pruebas.

¹ <http://marmotta.apache.org/platform/sparql-module.html>

Otros aspectos por considerar es la funcionalidad del módulo la cual consiste en permitir que las consultas federadas se desplieguen en la nube de *Linked Data* y que el usuario final pueda recuperar información geoespacial de forma distribuida. De esta manera, el módulo deberá ser capaz de combinar las respuestas de las diversas fuentes consultadas sin que existan resultados repetidos u omitidos, y cómo es que se manipulará los datos geográficos para poderlos desplegar en una aplicación web. En este sentido, existen diversas alternativas para lograr esta meta, un ejemplo es la herramienta *Map4RDF* cuya función es visualizar y explorar *datasets* RDF cuya información sea geométrica [11]. Por tanto, cabe aclarar que se reutilizará una herramienta de este estilo solo para visualizar los datos recuperados por el módulo de consultas federadas geoespaciales.

De esta forma, el proyecto a desarrollar pretende contribuir a las herramientas usadas en la Web Semántica. Con la intención de cumplir tal propósito la pregunta que surge es: ¿Cuánta información extra revelará una consulta geoespacial federada frente a una consulta a un único repositorio?

3. Propuesta de solución

En la actualidad, existen diversos motores *triple store*, aunque gran parte de ellos [12] solo pueden hacer consultas de forma individual sobre aquellos conjuntos de datos con características geoespaciales que tienen almacenados. Así, la herramienta a desarrollar en el proyecto terminal pretende solucionar este problema brindando la posibilidad de realizar consultas distribuidas a conjuntos de datos geoespaciales presentes en la nube de *Linked Data* para el *triple store* Apache Marmotta. De esta manera, se pretende aprovechar los beneficios que dicha plataforma ofrece para avanzar en el estado del arte y contribuir con el desarrollo y progreso de la Web Semántica geoespacial.

En la figura 4 se muestra en línea negra una consulta simple a un solo *triple store*, mientras que, en la línea roja punteada, funcionalidad que proporcionará este proyecto, representa la capacidad de consultar diversos *triple store* a la vez.

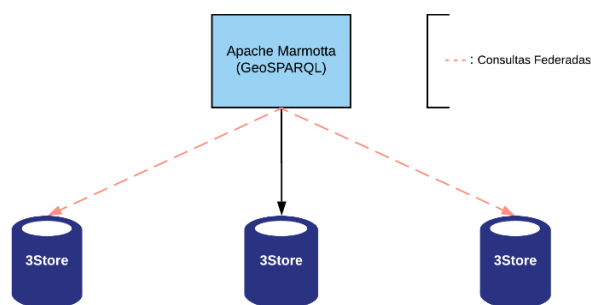


Fig. 3 Diagrama de Apache Marmotta haciendo una consulta individual (línea negra) y consultas federadas (líneas punteadas rojas)

Descomponiendo la figura 4, en la figura 5 se muestra un diagrama a bloques que detalla el proceso de una consulta geoespacial simple, es decir, una consulta a un solo *triple store*. La consulta, en función de los operadores usados, retornará una geometría o un booleano.

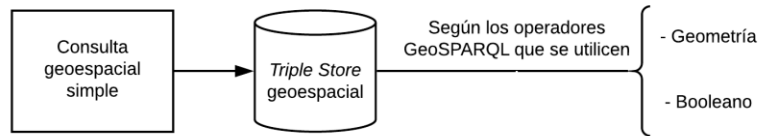


Fig. 4 Consulta geoespacial simple

La intención del módulo a desarrollar es que la consulta en Apache Marmotta no se centre en un solo *triple store*, sino que consulte a todas las *triple store* que estén disponibles y que presenten información con características geoespaciales. Una vez que se consulten a todas las bases de datos geoespaciales, las respuestas tendrán que ser unidas de tal forma que no exista datos repetidos. Este resultado será íntegro, ya que contendrá información no solo de un *triple store*, sino que la consulta será respondida con información de diferentes fuentes. La figura 6 muestra un diagrama a bloques de lo explicado.

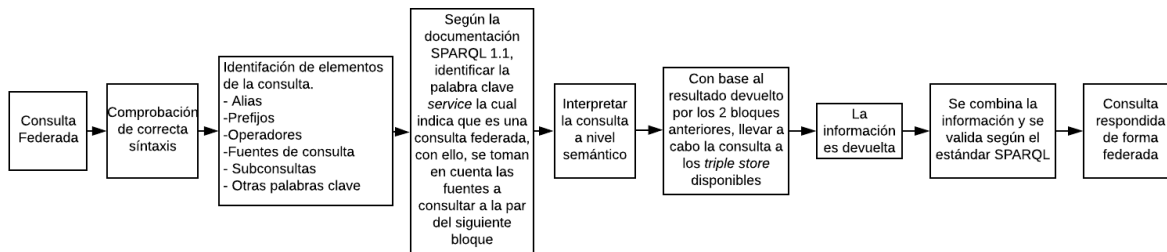


Fig. 5 Diagrama a bloques de una consulta geoespacial federada

El desarrollo e implementación del módulo de consultas federadas geoespaciales serán basados en los estándares SPARQL y GeoSPARQL, es decir, su funcionamiento, desarrollo y resultados deberán cumplir con las características estipuladas en los documentos *SPARQL 1.1 Federated Query* [3] y *A Geographic Query Language for RDF Data* [13].

Apache Marmotta está basado en el lenguaje de consultas de la Web Semántica SPARQL y es capaz de hacer consultas geográficas SPARQL cuya tecnología es mejor conocida como GeoSPARQL. El módulo que será desarrollado se integraría al código fuente del motor para lograr que éste sea capaz de hacer múltiples consultas en vez de solo una. En el proyecto se plantea desarrollar el módulo de consultas federadas para que funcione con GeoSPARQL. La figura 7 muestra una representación de los módulos de Apache Marmotta. En verde se encuentra el lenguaje y protocolo SPARQL mientras que en amarillo la extensión GeoSPARQL; en rojo se muestra el módulo de consultas federadas a implementar.

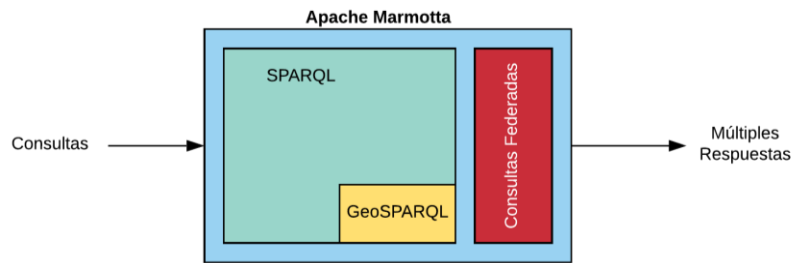


Fig. 6 Diagrama a bloques de la implementación del módulo de consultas federadas

Para caracterizar el módulo desarrollado, se propone llevar a cabo un conjunto de pruebas descritas en el escenario de pruebas. Esto a su vez permitirá hacer un *benchmarking* contra otras plataformas similares para determinar sus fortalezas y debilidades respecto dichas plataformas.

Los parámetros para tomar en cuenta en la caracterización estarán completamente determinados después de haber concluido el *benchmarking*, pero entre estos podrían estar los siguientes:

- Tiempo de carga.
- Tiempo de consulta.
- Consultas por segundo.
- Consultas mezcladas por segundo.
- Uso de operadores, modificadores y acceso a datos

Estos parámetros están detallados en los siguientes 3 documentos:

- *An Evaluation of Approaches to Federated Query Processing over Linked Data* [14]
- *SP2Bench: A SPARQL Performance Benchmark* [15]
- *The Berlin SPARQL Benchmark* [16]

Con los datos recabados y una vez desarrollado el módulo, se podrá redactar la documentación de dicha herramienta que permita su correcto uso, así como su compartición con la comunidad relacionada con esos temas.

Además, el módulo se complementará con una herramienta para que los usuarios finales puedan visualizar los resultados de las consultas geoespaciales federadas. En la actualidad ya existen herramientas que permiten visualizar datos geoespaciales, entonces con el único fin de demostrar que el módulo funciona correctamente, por ende, otra alternativa de comprobación de funcionamiento, este trabajo presentará los resultados obtenidos usando uno de esos recursos que permitan desplegar dicha información y analizarla de forma amigable como puede ser una aplicación web. Un resultado, similar al esperado, se muestra en la figura 8 donde se aprecian puntos en color azul que representan la información geoespacial recuperada de múltiples *triple store*.

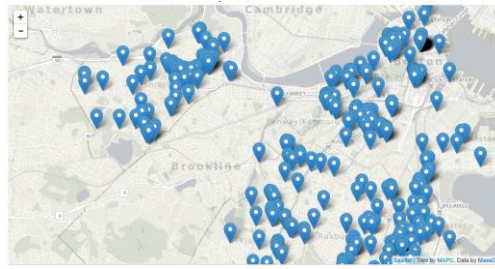


Fig. 7 Propuesta de visualización

3.1. Alcances (Resultados esperados)

Al no existir un módulo de consultas federadas geoespaciales para Apache Marmotta, ningunos de los alcances propuestos han sido elaborados hasta el momento, Así, los alcances fijados para este trabajo son:

- El *triple store* Apache Marmotta en conjunto con el módulo será capaz de hacer consultas federadas geoespaciales con base a los estándares SPARQL y GeoSPARQL.
- El *benchmarking* estará inspirado en trabajos presentes en el estado del arte, tales como [14], [15] y [16].
- La caracterización será delimitada con los resultados arrojados por el *benchmarking*.
- La comprobación y explotación de datos geoespaciales devueltos por el módulo de consultas federadas en Apache Marmotta serán mediante una aplicación Web.
- Una vez que se tenga el módulo desarrollado, implementado y después de haberlo puesto a prueba con las pruebas descritas en el escenario de pruebas, se propondrá a la organización *Apache Software Foundation* que el módulo sea incorporado en el sistema Apache Marmotta de forma oficial.

Limitaciones

- Se necesita una computadora con conexión a Internet, sistema operativo Unix-Linux Ubuntu para instalar Apache Marmotta y el módulo y un navegador web para visualizar la aplicación web.
- El módulo estará integrado al código fuente del motor Marmotta por lo que no será un *framework* ni una API. Su uso será completamente dentro del entorno Apache Marmotta.
- La herramienta por usar para la exploración de datos geoespaciales desplegados por el módulo de consultas federadas no será desarrollada, será seleccionada de las opciones actuales y usada.
- Se busca incorporar la funcionalidad a Apache Marmotta mas no competir contra otros *triple stores*.
- El proyecto no trata sobre una búsqueda semántica interespatial.

4. Objetivo general

Desarrollar un módulo de consultas geoespaciales federadas para el *triple store* Apache Marmotta, con el propósito de contribuir al avance de las tecnologías usadas en la Web Semántica y proveer una alternativa *open source* diferente a los *triple store* existentes.

4.1. Objetivos específicos

- Implementar, con base en los estándares SPARQL, GeoSPARQL, así como auxiliándose de otras tecnologías involucradas en la Web Semántica y *Linked Data*, un módulo de consultas federadas para el *triple store* Apache Marmotta.
- Evaluar el rendimiento de las consultas hechas por el módulo desarrollado (*benchmarking*).
- Comparar el *triple store* Apache Marmotta con otros *triple store* auxiliándose de la caracterización y *benchmarking* del módulo.
- Caracterizar el módulo de consultas federadas.
- Explotar las características geoespaciales de los datos obtenidos del despliegue de consultas federadas mediante una aplicación Web para poder visualizarlos y explorarlos.

5. Estado del arte

Los trabajos desarrollados que se presentan en este documento a nivel internacional son 12, a nivel nacional 3 y en UPIITA es 1. Los 6 trabajos que se presentan o bien usan las herramientas del *Linked Data* o dan un enfoque de cómo se usan las tecnologías, pero ninguna aborda una propuesta similar a la considerada en este trabajo.

Internacional

Querying Geospatial Data over the Web: a GeoSPARQL Interface

En este artículo se describe cómo es que Nancy, Ralph y Dave crearon e implementaron una interfaz para datos GeoSPARQL llamada *GeoQuery TOOL*. Esta interfaz intuitiva pretende hacer que las consultas geoespaciales sean más fáciles de hacer al implementar listas en su interfaz para poder escoger atributos y operadores espaciales. Con base en los datos de entrada que haya ingresado el usuario, *GeoQuery* genera código GeoSPARQL automáticamente, realiza la consulta usando el *triple store* Parliament y es desplegado en una aplicación web en vez de utilizar el Sistema de Información Geográfica (GIS, por sus siglas en inglés) [17]

Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL

Este trabajo de Robert y Dave se presentan razones por las cuáles hay que usar GeoSPARQL, su estado del arte en la industria y en la investigación, y su implementación de GeoSPARQL en el *triple store* Parliament. Explican conceptos geoespaciales tales como las diferencia entre una característica y una geometría, qué es un sistema de referencia de coordenadas (CRS, por sus siglas en inglés) y las relaciones topológicas que existen. En esta última mencionan 8 operaciones básicas y sus 2 variantes Egenhofer y RCC8. Ambas expresan las mismas operaciones al ser equivalentes. En general, este documento da las herramientas para comprender GeoSPARQL, así como su uso e implementación con la intención de que empresas u organizaciones consideren adoptar esta tecnología [18].

Strategies for Executing Federated Queries in SPARQL1.1

En esta propuesta se analizan diferentes estrategias para implementar consultas federadas con la intención de evitar los límites que un *endpoint* presenta. Las estrategias que proponen están basadas en la versión de SPARQL 1.1 mediante descomposición de consultas federadas. En este artículo se describe la sintaxis de SPARQL mediante teoría de conjuntos, así como la evaluación de las estrategias propuestas probando los teoremas propuestos. Por último, los autores muestran la mejora de resultados habiendo implementado sus estrategias [19].

Linking UK Government Data

En este trabajo se establecen los casos de uso para la adopción de los principios de *Linked Data* para la publicación de datos públicos del gobierno de Reino Unido. Además, los autores plantean los beneficios de usar *Linked Data*. En sí, el trabajo pretende convencer a empresas, centros de estudio y a desarrolladores a empezar a usar *Linked Data*. En el documento se abordan los temas de datos públicos del gobierno y la responsabilidad que deben existir para su publicación, patrones de diseño, tópicos imprescindibles para abordar la publicación de datos relacionados con información estadística y geoespacial. Por último, presentan las tecnologías disponibles en el contexto *Linked Data* para que desarrolladores de software puedan crear nuevas herramientas [20].

A parallel approach for improving Geo-SPARQL query performance

Esta investigación expone el problema actual que existe en las consultas geoespaciales que involucran complejas relaciones topológicas y que en conjunto a las bases de conocimiento las cuales no están indexadas, generan ineficiencia en consultas de sus datos. Entonces, en el documento a parte de exponer el problema, proponen una estrategia para disminuirlo, que consiste en el uso de cómputo en paralelo y la creación virtual de índices de la base de conocimiento. En el desarrollo del documento, se muestran el antes y el después de los resultados obtenidos al usar su estrategia sobre consultas en la ciudad de Connecticut, EUA [21].

DBpedia SPARQL Benchmark – Performance Assessment with Real Queries on Real Data

El documento propone una nueva forma de hacer un *benchmarking* con el propósito de demostrar que los *triples stores* existentes no son tan homogéneos como los otros *benchmarking* lo muestran, es decir, el rendimiento depende de los parámetros usados por lo que una consulta no puede mostrar la misma eficiencia respecto a otra consulta.

Se ponen a prueba 4 *triple store* populares Virtuoso, Sesame, Jena-TDB y BigOWLIM. A lo largo del documento muestran el desarrollo del *benchmarking*, desde la generación del conjunto de datos de DBpedia, hasta los resultados obtenidos. En este trabajos los autores miden la cantidad de consultas por segundo (QpS, por sus siglas en inglés) en 4 casos: 10%, 50%, 100% y 200% del conjunto de datos generados previamente.

A parte del *benchmarking*, se presenta una discusión donde se habla sobre las fortalezas y debilidades de cada *triple store* al hacer múltiples y simples consultas con ellos. También se presentan trabajos relacionados y los retos por vencer en el campo de los *benchmarking en triple stores* [22].

Explotación de información en el dominio geo-hídrico ecuatoriano utilizando tecnología semántica

El documento presenta una propuesta de cómo aprovechar los datos geográficos en el dominio geo hídrico. Se apoyan en la única herramienta *open source*, Parliament, para llevar a cabo las consultas geográficas de forma federada.

Presentan el escenario del dominio geo hídrico ecuatoriano y la manera de cómo explotan los datos mediante 3 repositorios de *triple store*.

Cuando terminan de obtener los resultados de las consultas, estos son visualizados con el apoyo de una herramienta llamada MAP4RDF

Cabe decir que en este trabajo no se hace ninguna implementación o desarrollo sobre un *triple store*, solo utilizan ejemplos de consultas y obtienen datos [6].

Sextant: Browsing and Mapping the Ocean of Linked Geospatial Data

En este documento presentan una herramienta Web llamada *Sextant* que permite la exploración de datos enlazados geoespaciales para crear, compartir y edición colaborativa mediante la combinación de datos geoespaciales enlazados y otro tipo de información disponible en archivos de formato OGC [23].

Answering geospatial queries over relational data

En este *paper* los autores parten del hecho de que los datos geoespaciales son comúnmente almacenados en sistemas manejadores de bases de datos, *DBMS* por sus siglas en inglés, del tipo geoespacial. Para llevar a cabo dicha tarea, se deben de convertir esos datos en datos RDF y almacenarlos en *triple store* cada vez que nuevos datos llegan. Esta labor resulta ser monótona, generando apatía en los administradores de dichas bases de datos para actualizarlas. Si bien existe el paradigma de administración de datos denominada Acceso a Datos Basado en Ontologías, *OBDA* por sus siglas en inglés, que se encargan de ofrecer consultas SPARQL en formato SQL, no existe soporte para datos geoespaciales; esto implica no hacer consultas actualizadas o no obtener

resultados de consultas geoespaciales. La solución por parte de los autores de habilitar consultas del tipo GeoSPARQL *en el aire* al no convertir los datos consultados a RDF y después almacenarlos en un *triple store*. Como resultado, se pueden hacer consultas sin tener que hacer conversiones y almacenamiento, solo puras consultas [24].

Benchmarking Commercial RDF stores with Publications Office Dataset

Los autores presentan un *benchmark* para RDF stores usando *datasets* de la oficina de publicaciones, PO por sus siglas en inglés. En la comparación se miden 4 características: *bulkloading*, escalabilidad, estabilidad y ejecución de consultas. De la misma PO se utilizaron *datasets* normalizados y no normalizados. Usando lo mismos *datasets* se construyeron nuevos datos para medir la escalabilidad; en estas consultas se dividieron en 2 categorías: consultas instantáneas, consultas que involucran operadores primitivos, y consultas analíticas con el propósito de medir la calidad de servicio y no solo la velocidad. Se concluye que el rendimiento no es homogéneo entre sistemas y que la calidad y velocidad de resultados dependen de diversos parámetros como el tipo de consultas, características de base de datos o *hardware* utilizado [25].

Geographica: A Benchmark for Geospatial RDF Stores

Se consideró por parte de los autores que no existía un *benchmark* que evaluara *triple store* geoespaciales que fuese usado de una manera conocida por lo que el propósito de haber desarrollado el *benchmark Geographica* fue contribuir al estado de arte. En *Geographica* se usaron datos sintéticos y datos del mundo real para probar la funcionalidad ofrecida y el rendimiento de RDF stores geoespaciales; se usaron en específico Strabon, Parliament y uSeekM los cuales eran los más completos para la evaluación. El desarrollo de esta comparativa fue con el fin de ofrecer una metodología que permitiera evaluar RDF stores de una mejor manera que propuestas previas. Ellos usaron 2 *workloads* de datos: sintético y del mundo real. Con ellos evaluaron eficiencia de funciones primitivas espaciales y el rendimiento de los RDF stores en *reverse geocoding*, *map search* y *browsing*. Los autores buscan, como trabajo futuro, expandir el *benchmark* para cubrir el estándar GeoSPARQL de una manera completa y probar a *Geographica* en un ecosistema centralizado y distribuido [26].

Geoyasgui: The GeoSPARQL query editor and result visualizer

Los autores abordan una problemática presente en los editores y evaluadores de consultas de GeoSPARQL al trabajar con el *Land Registry and Mapping Agency*, mejor conocido como *Kadaster*, quienes publican una gran cantidad de *datasets* entre los cuales son publicados de diversas maneras y en una cantidad considerable, muchos de esos datos son geoespaciales. Cabe decir que *Kadaster* publica sus datos basados el estándar GeoSPARQL como *Linked Open Data*. Básicamente lo que hace el sistema es evaluar las consultas al ser enviadas a un SPARQL *endpoint* donde se evalúa el álgebra respecto a la colección de datos almacenados en el *endpoint* esto con el objetivo de optimizar la consulta y después el *endpoint* devuelve los resultados de la consulta en un formato estandarizado (XML, JSON, CSV/TSV). En cuanto a la edición de las consultas, los autores se basaron en trabajos previos: *YASQE*, *YASR*, *YASGUI* pero usando los componentes de GeoSPARQL dando así una edición de consultas de datos geoespaciales ofreciendo un visualizador de consultas, un *feedback* directamente al usuario al autocompletar y resaltar sintaxis de la consulta, y un servicio Web que une los elementos anteriores [27].

Nacional

Linked Open Data en la Biblioteca Digital Semántica Académica

En este trabajo se describe y analiza cómo es que el *Linked Open Data* es aplicado en las bibliotecas digitales semánticas. En el documento se presenta un marco teórico explicando las tecnologías que son necesarias para el trabajo tales como XML, RDF, SPARQL, *triple store*. Además, se describen las herramientas que cuentan en la UNAM para el desarrollo de la biblioteca digital. También presentan iniciativas de *Linked Open Data* en Alemania, Reino Unido y en México. Para concluir su documento, dan una justificación del porque su proyecto es innovador al usar nuevas alternativas de publicación, búsqueda y recuperación de información [28][24].

Enfoque semántico para el descubrimiento de recursos sensible al contexto sobre contenidos académicos estructurados con OAI-PMH

En este trabajo describen un enfoque que considera los recursos de información estructurados con el Protocolo para Cosecha de Metadatos de la Iniciativa de Archivos Abiertos, OAI-PMH por sus siglas en inglés, representación ontológica y su uso en aplicaciones de recuperación de información. Los autores presentan los conceptos a tomar en cuenta como son OAI-PMH, Dublin-Core, sensibilidad al contexto, ontologías. En el trabajo se usó Apache Jena el cual es un *triple store* donde llevaron a cabo sus pruebas. Se presenta una metodología para consultar y obtener información. Después muestran sus resultados mostrando un grafo que describe la relación entre las instancias de ejemplo que propusieron. Para finalizar, abordan trabajos relacionados, conclusiones y el trabajo a futuro para mejorar su propuesta [29].

Facilitador de contenido móvil para el viajero basado en servicios de localización y Web Semántica

Esta tesis desarrollada en el CIC del IPN donde proponen el diseño e implementación de un facilitador de contenido móvil usando técnicas de la Web Semántica y datos geográficos con GeoSPARQL para recomendaciones de alimentación, hospedaje y sitios de interés a turistas. El proyecto se basó en lenguajes de la Web Semántica como RDF, OWL y SPARQL. Fue desarrollado sobre un *servlet* de Java en conjunto a servicios basado en geolocalización. Se basó en una metodología dividida en 5 etapas: conceptualización, recuperación de términos, recuperación de información turística y su presentación. Al ser un proyecto que pretende estar disponibles en cualquier dispositivo móvil, se hicieron pruebas experimentales de su servicio web, de la aplicación móvil, de la recuperación de información turística y la forma de presentar los datos. Por último, en la conclusión hace una comparación de su proyecto respecto a una popular aplicación llamada *TripAdvisor* con la principal característica de que en el proyecto desarrollado muestra resultados precisos, limitados, pero correspondían a la zona donde el usuario se encuentra, mientras que la aplicación comercial en diversos casos devolvía resultados que no estaban relacionados a la consulta hecha [30].

UPIITA

Recuperación de información geográfica utilizando similitud semántica

Trabajo hecho por 2 estudiantes donde proponen el uso de un sistema de información geográfica (GIS) y un sistema de geoposicionamiento global (GPS) para identificar sitios de interés alrededor de la zona de donde se encuentre el usuario usando técnicas de similitud semántica, en específico la teoría de confusión. El proyecto fue probado en los alrededores de la IPN Zacatenco para encontrar banco, hospitales y lugares para comer y entretenerse [31].

Software similar

A continuación, se muestran 5 *triple store* que existen en la actualidad. Cabe decir que ninguno de ellos tiene la capacidad que se pretende implementar en Apache Marmotta.

- *Oracle Spatial* [32]
- *ClioPatria* [33]
- *Mulgara* [34]
- *Marklogic* [35]
- *Blazegraph* [36]

6. Marco teórico

Los conceptos descritos en esta sección son fundamentales en el presente proyecto terminal para mostrar qué compone este proyecto. RDF, URI, *triple store* y SPARQL *endpoint* son conceptos que se deben manejar sí o sí para el desarrollo del módulo ya que estos son elementos indispensables para su desarrollo; tales conceptos explican los objetos con los que se trabajaran a lo largo de este proyecto terminal.

Linked Data y Web Semántica son conceptos por tener presentes ya que no se debe de perder de vista sobre qué contexto se está desarrollando el módulo, porque bien se podría hablar de datos cualquiera pero no es así, el proyecto terminal estará construido bajo estos conceptos.

SPARQL y GeoSPARL son los estándares por seguir y cumplir. Si no fuera así, simplemente no se podrá construir nada que funcione con el *triple store* Apache Marmotta o cualquier otra plataforma que se basen en cualquiera de los 2 estándares.

6.1. Linked Data

El *Linked Data* es un conjunto de buenas prácticas para publicar y conectar datos estructurados en la web. Los principios que tiene asociado son URI para identificar entidades en el mundo, HTTP que es el mecanismo por donde se recuperan recursos o descripciones de recursos y RDF para estructurar y enlazar datos que describen cosas en el mundo [37].

Ya que el *Linked Data* está basado en web, la diferencia entre sitios de Internet comunes y los basados en *Linked Data* es que mientras que los HTML simples en la web son conectados mediante hipervínculos comunes, *Linked Data* se basa en documentos que albergan datos en formato RDF.

Web Semántica

6.2. Web semántica

Es una web de datos de cualquier índole. En la figura 9 se muestra la pila que Tim Berners-Lee propone para describir la Web Semántica en cuanto a elementos que lo conforman.

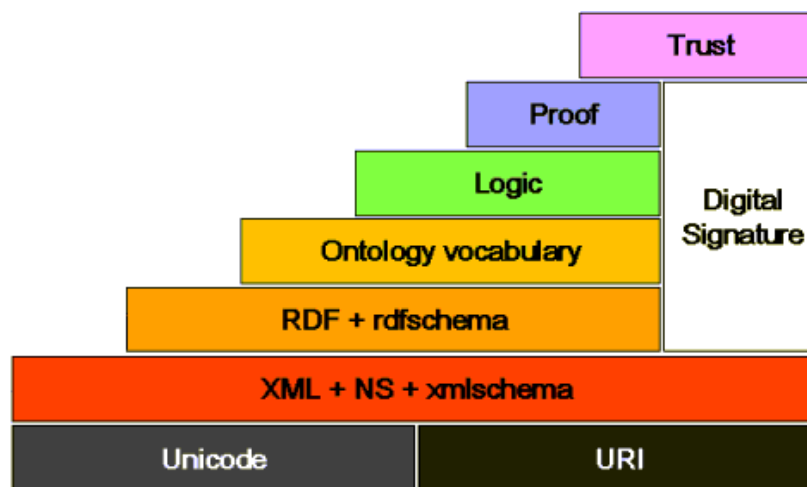


Fig. 8 Diagrama propuesto por Tim Berners-Lee mostrando las tecnologías que conforma la Web Semántica

Describiendo de abajo hacia arriba, en el primer nivel se encuentran las tecnologías que permiten la identificación de cada entidad (Unicode y URI). En el segundo nivel se encuentran las tecnologías que le dan estructura, pero no significado, al documento publicado en la web (XML, NS, XML schema). A partir del tercer nivel a la sexto, la pila muestra una firma digital (color blanco) para garantizar la autenticidad de cada entidad. En el tercer nivel están las tecnologías que describen el contenido del documento tal que permiten a la mayoría de las computadoras entender el significado de la entidad en cuestión (RDF y RDF schema). En el cuarto nivel, si bien no es una tecnología, se encuentra un documento o archivo que define las relaciones entre entidades mediante ontologías

y reglas de inferencia (comúnmente se implementa el lenguaje OWL para lograrlo). En el quinto nivel se encuentra la lógica la cual reúne las diversas ontologías usadas al igual que las reglas de los lenguajes que se usaron para describir y estructurar la entidad; en este nivel se llevan a cabo las inferencias y se les da un significado a los datos. En el sexto nivel, se realiza la prueba, es decir, el cómo se hicieron las inferencias y el origen de los datos. Por último, se encuentra la confianza, la confianza de que el sistema es capaz de funcionar correctamente, de que el sistema pueda explicar que hace, del origen de las fuentes de datos y servicios, así como la tecnología e interfaz de usuario.

La Web Semántica se puede ver como un todo y el *Linked Data* como los elementos que la conforman puesto que proporciona sustento y las bases para que la Web Semántica sea construida correctamente [38].

6.3. RDF (Resource Description Framework)

El *framework* de descripción de recurso, RDF por sus siglas en inglés, es un modelo estándar para el intercambio de datos en la web.

RDF extiende las estructuras de enlaces de la web al usar URI tanto para nombrar relaciones entre cosas como para los puntos finales de las relaciones, a veces referido como “triple”. El modelo RDF permite representar los datos como sujeto, predicado y objeto. El sujeto y el predicado de un *triple* son URI que identifican a cada uno. El predicado especifica como el sujeto y el objeto están relacionados, y también es representado por un URI. Esta característica provee un modelo de datos basado en grafos [39].

6.4. URI (Uniform Resource Identifier)

Identificador de recursos uniforme, URI por sus siglas en inglés, es una cadena ASCII que identifica recursos de información en la Web Semántica.

Tal y como se observa en la figura 10, una URI puede estar compuesto de un localizador de recursos uniforme (URL, por sus siglas en inglés), de un nombre de recursos uniforme (URN, por sus siglas en inglés) o de ambos [40].

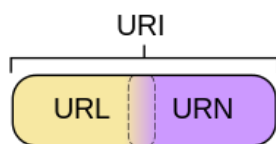


Fig. 9 Diagrama de cómo está compuesto un URI

6.5. RDF triple store

Es un tipo de base de datos basada en grafos de tripletas RDF.

Al ser una base de datos basada en grafos, el *triple store* puede ser vista como una red de objetos enlazados. El *triple store* al ser una herramienta de la Web Semántica, las entidades que conforman a la base de datos, tripletas RDF, son representadas como sujeto, predicado y objeto o también puede ser considerada como sujeto, predicado y etiqueta [41].

6.6. SPARQL

SPARQL es un acrónimo para el Protocolo SPARQL y Lenguaje de Consultas RDF, por sus siglas en inglés, y es un protocolo y lenguaje de consultas para *Linked Data* en la web o bases de datos semánticas basadas en grafos (*RDF triple stores*). SPARQL está diseñado y respaldado por el consorcio de la web (W3C).

El siguiente código muestra cómo hacer una consulta SPARQL de los músicos mexicanos famosos que ya han muerto y que están sobre el *triple store* de Dbpedia.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbp: <http://dbpedia.org/ontology/>
SELECT ?musico ?nombreMusico ?fechaFallecimiento
WHERE {
    ?musico dcterms:subject
    <http://dbpedia.org/resource/Category:Mexican_musicians>;
    rdfs:label ?nombreMusico ;
    dbp:birthDate ?fechaNacimiento ;
    dbp:deathDate ?fechaFallecimiento .
    FILTER (LANG(?nombreMusico) = "es")
}
```

La figura 11 muestra el resultado de la consulta

musico	nombreMusico	fechaFallecimiento
http://dbpedia.org/resource/Juan_García_Esquivel	"Juan García Esquivel"@es	2002-01-03
http://dbpedia.org/resource/Juan_García_Esquivel	"Juan García Esquivel"@es	"2002-1-3"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Cornelio_Reyna	"Cornelio Reyna"@es	1997-01-22
http://dbpedia.org/resource/Cornelio_Reyna	"Cornelio Reyna"@es	"1997-1-22"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Manolo_Muñoz	"Manolo Muñoz"@es	"2000-3-14"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Carlos_Gómez_Barrera	"Carlos Gómez Barrera"@es	"1996-1-1"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Tito_Guizar	"Tito Guizar"@es	1999-12-24
http://dbpedia.org/resource/Lorenzo_Barcelata	"Lorenzo Barcelata"@es	"1943-7-13"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Consuelo_Velázquez	"Consuelo Velázquez"@es	"2005-1-22"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Chavela_Vargas	"Chavela Vargas"@es	2012-08-05
http://dbpedia.org/resource/Chavela_Vargas	"Chavela Vargas"@es	"2012-8-5"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/José_Mojica	"José Mojica"@es	1974-09-20
http://dbpedia.org/resource/José_Mojica	"José Mojica"@es	"1974-9-20"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Rafael_Méndez	"Rafael Méndez (trompetista)"@es	"1981-9-15"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Chico_Che	"Chico Che"@es	"1989-3-29"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Macedonio_Alcalá	"Macedonio Alcalá"@es	1869-08-24
http://dbpedia.org/resource/Macedonio_Alcalá	"Macedonio Alcalá"@es	"1869-8-24"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Fernando_Valadés	"Fernando Valadés"@es	1978-12-14
http://dbpedia.org/resource/Ángel_Tavira	"Ángel Tavira"@es	"2008-6-30"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Enrique_Ballesté	"Enrique Ballesté"@es	"2015-9-19"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/José_Agustín_Ramírez_Altamirano	"José Agustín Ramírez Altamirano"@es	"1957-9-12"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Zacarías_Salmerón	"Zacarías Salmerón"@es	"2011-1-29"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Silvestre_Vargas	"Silvestre Vargas"@es	"1985-10-7"^^<http://www.w3.org/2001/XMLSchema#date>
http://dbpedia.org/resource/Cirilo_Marmolejo	"Cirilo Marmolejo"@es	"1960-1-1"^^<http://www.w3.org/2001/XMLSchema#date>


Fig. 10 Resultado de la consulta en SPARQL

Actualmente se encuentra en su segunda versión, SPARQL 1.1, y ya describe una extensión para explícitamente delegar subconsultas a diferentes SPARQL *endpoint*. Esta característica es conocida como consulta federada.

6.7. SPARQL endpoint

Un SPARQL *endpoint* es un servicio web que acepta consultas SPARQL [42].

DBpedia es quizá el SPARQL *endpoint* más famoso, el cual aloja la información que se ve, a veces, en la parte derecha de las páginas de Wikipedia, estas son conocidas como *infoboxes*. En la figura 12 se ven datos extraídos desde DBpedia (cuadro rojo) y usados por Wikipedia.



The screenshot shows the Wikipedia page for "Akita Inu". The infobox, which is highlighted with a red border, contains the following information:

Akita Inu	
Otros nombres	Akita Ken (秋田犬) Japanese Akita Akita-
Región de origen	 Japón
Características	
Tipo	perro
Dimensiones	Machos: 67 cm, hembras: 61 cm
Tamaño	Grande.
Peso	Machos: 34 a 53 kg, hembras: 30 a 49 kg
Ojos	pequeños, casi triangulares y oscuros.
Orejas	triangulares, redondeadas en la punta, erguidas e inclinadas hacia delante.
Cola	alta, gruesa y enroscada sobre la espalda.
Carácter	temperamento calmado, fiel, dominante Kennel Council

Below the infobox, there is a link to "[editar datos en Wikidata]".

Fig. 11 Datos de DBpedia usados por Wikipedia

6.8. GeoSPARQL

Es un lenguaje de consultas geográficas para datos RDF en la Web Semántica estandarizado por el *Open Geospatial Consortium* (OGC). GeoSPARQL define un vocabulario para la representación de datos en RDF y también define una extensión para el lenguaje de consultas geoespaciales SPARQL.

Un ejemplo de consulta es el siguiente: Determinar los objetos que estén contenidos en la figura A de la figura 13

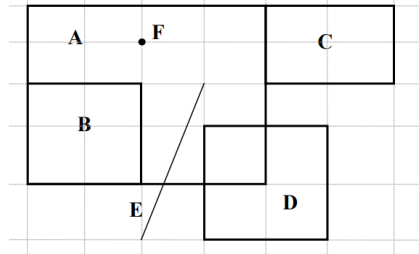


Fig. 12 Representación de datos espaciales

A continuación, la consulta SPARQL asociada para resolver ese interrogante:

```
PREFIX my: <http://example.org/ApplicationSchema#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX geof: <http://www.opengis.net/def/function/geosparql/>
SELECT ?f
WHERE {
  my:A my:hasExactGeometry ?aGeom .
  ?aGeom geo:asWKT ?aWKT .
  ?f my:hasExactGeometry ?fGeom .
  ?fGeom geo:asWKT ?fWKT .
  FILTER (geof:sfContains(?aWKT, ?fWKT) &&
    !sameTerm(?aGeom, ?fGeom))
}
```

El resultado se muestra en la figura 14.

?f
my:B
my:F

Fig. 13 Resultado de la consulta en GeoSPARQL

Como era de esperarse, el resultado dice que 'A' contiene tanto a 'B' como al punto 'F'

6.9. Arquitectura SOA

La arquitectura orientada a servicios, SOA por sus siglas en inglés, es un concepto en la ingeniería de software cuya intención es reducir costos de implementación servicios para clientes innovadores, agilidad para adaptar cambios del sistema [43]. Sus características son las siguientes

- Objetivos de negocio ligados a la infraestructura de tecnologías de información (TI)
- Orientada a la arquitectura de sistemas. Esto busca que la lógica de procesos de negocio no intervenga con la lógica del software de un sistema.
- Separación de objetivos: Dividir objetivos primarios en diferentes características con funcionalidades estrechas tan pequeñas como sean posible.
- Modularidad: La aplicación estará dividida en piezas distinguibles las cuales, desempeñarán una función en específico en el sistema.

- Bajo acoplamiento: Los atributos de los componentes de un sistema no tienen y/o no hacen uso del conocimiento de otros componentes independientes.
- Encapsulación: El acceso a datos con sus respectivas instrucciones de manipulación, las cuales estarán dentro de un paquete, es posible mediante una interfaz independiente.
- Interfaces: Implementación de un pequeño conjunto de interfaces que son mantenidas de manera separada.
- Mensajes: Uso de mensaje que contengan información a ser intercambiada a través de las interfaces mediante una estructura y vocabulario delimitado por un esquema.
- Reutilización: Es la acción de reutilizar un componente múltiples veces.
- Composabilidad: Capacidad de seleccionar componentes y ensamblarlos de diversas maneras que puedan cumplir el objetivo de la aplicación

6.10. Protocolo HTTP

El protocolo HTTP es un protocolo a nivel de aplicación para colaborar, distribuir sistemas de información de tipo hipermedia (texto, imagen, audio, video, mapas). La comunicación entre sistemas, según el protocolo, está basado en respuestas y peticiones [44].

- El cliente hace una petición HTTP
- El servidor recibe la petición
- El servidor procesa la petición
- El servidor lleva a cabo una respuesta HTTP con la información solicitada o con un mensaje de error.
- El cliente recibe la respuesta.

Así mismo, el protocolo HTTP establece un grupo de métodos de petición los cuales indican que acción de quiere llevar a cabo sobre un específico recurso. Los métodos son los siguientes [45]

- GET: Pide una representación de un específico recurso. Este método solo debe ser capaz de recuperar datos.
- POST: Crea un nuevo recurso en el servidor.
- PUT: Las representaciones actuales de un recurso son sustituidas por la información que lleva el mensaje, y en caso de no existir, lo crea.
- DELETE: Elimina un recurso determinado.
- HEAD: Petición similar a la de GET, pero sin cuerpo de la respuesta.
- CONNECT: Establece comunicación en 2 vías con el servidor del recurso solicitado. Comúnmente es usado para llevar a cabo una comunicación túnel.
- TRACE: Lleva a cabo un mensaje de prueba de ida y vuelta en toda la ruta hasta el recurso objetivo con el fin de ser un mecanismo de depuración.
- OPTIONS: Método que permite describir las opciones de comunicación para un recurso en específico.

- PATCH: Permite modificar un recurso de manera parcial, a diferencia de que el método PUT lo hace sobre todo el recurso.

6.11. *REST*

La transferencia de estado representacional, REST por sus siglas en inglés, es un tipo de arquitectura define reglas de comunicación entre sistemas computacionales en la Web [46].

A pesar de que *REST* no es un estándar, si hace uso de ellos

- URL
- HTTP
- XML, GIF, JPG, etc. (representación de recursos)
- Extensiones multipropósito de correo de Internet (MIME)

Las características de *REST* son

- Basado en cliente servidor: Consumo de componentes mediante peticiones
- Sistemas independientes: Los sistemas computacionales no deben de estar desarrollados en el mismo lenguaje o paradigma de programación mientras estos cumplan el estilo de arquitectura *REST* se puede establece comunicación entre los sistemas.
- Uso de memoria caché: Con el fin de mejorar la eficiencia de respuestas, los sistemas deben de ser capaces de decidir si las respuestas son o no parte de la memoria caché.
- Recursos etiquetados: Los sistemas deben de identificar a sus recursos con una URL.
- Interfaz uniforme: Cualquier recurso del sistema puede accederse mediante una interfaz genérica. Ejemplo: métodos HTTP
- Representación de recursos enlazados: Las representaciones de los recursos deben de estar conectados entre sí para que el usuario sea capaz de ir de una representación a otra.
- Capas entre los componentes: Con el fin de otorgar servicios al usuario como seguridad, privacidad, eficiencia de servicio, entre otros, pueden usarse sistemas intermediarios como *gateways*, servidores *proxy* o servidores *caché* por mencionar algunos.

6.12. *JSON*

La notación de Objetos de JavaScript, JSON por sus siglas en inglés, es un formato de intercambio de información basado en texto e independiente de lenguaje. El formato fue un derivado del estándar del lenguaje de programación ECMAScript [47].

JSON está conformado por 2 estructuras

- Colección de pares nombre/valor: En diversos lenguajes de programación implementan esta estructura como un objeto, diccionario, registro, tabla *hash*, arreglos asociativos o lista de claves.
- Lista ordenada de valores: Se implementa en los lenguajes de programación como vectores, arreglos, secuencias o listas.

Ejemplo de un objeto *JSON* es

```
{
  "Image": {
    "Width": 800,
    "Height": 600,
    "Title": "View from 15th Floor",
    "Thumbnail": {
      "Url": "http://www.example.com/image/481989943",
      "Height": 125,
      "Width": 100
    },
    "Animated" : false,
    "IDs": [116, 943, 234, 38793]
  }
}
```


7. Análisis del sistema

7.1. Descripción general

7.1.1. Perspectiva del producto

La aplicación Web permite llevar a cabo las consultas federadas geoespaciales y usará la herramienta Map4RDF para visualizar e interactuar con los resultados en un mapa.

7.1.2. Características de los usuarios

Tipo de usuario	Anónimo
Formación	Estudiante, profesor y/o investigador
Habilidades	Conocimiento SPARQL y GeoSPARQL
Actividades	Consultar, visualizar datos, filtrar datos, descargar resultados

Tipo de usuario	Administrador
Formación	Estudiante, profesor y/o investigador
Habilidades	Conocimiento SPARQL y GeoSPARQL
Actividades	Configurar Apache Marmotta para que funcione bajo uno de los 3 perfiles: <i>simple</i> , <i>estándar</i> y <i>restricted</i> .

7.1.3. Restricciones

- Conexión a Internet
- Consultas basadas en los protocolos *SPARQL 1.1* y *GeoSPARQL*
- Las consultas pueden ser federadas
- Consultas a la nube *Linked Data*
- Los lenguajes de programación y de marcado para la plataforma serán
 - HTML
 - CSS
 - JavaScript
 - Java
- La aplicación estará basada en métodos HTTP, en REST y JSON

7.1.4. Suposiciones y dependencias

- La herramienta Map4RDF en próximas versiones seguirá usando las tecnologías mencionadas para la aplicación Web.
- Se asume que el módulo de consultas federadas estará disponible cuando la aplicación Web esté lista.

7.2. Requisitos específicos

7.2.1. Requerimientos funcionales

Número de requisito	RF01
Nombre de requisito	Establecer comunicación
Tipo	Requisito
Características	El usuario debe de inicializar la aplicación Web para poder usarla.
Descripción del requerimiento	Se debe de establecer comunicación entre el software Apache Marmotta ya que, sin ellos la aplicación Web no puede funcionar.

Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	Alta/Esencial

Número de requisito	RF02
Nombre de requisito	Valida conexión
Tipo	Requisito
Características	La aplicación Web debe validar la conexión.
Descripción del requerimiento	Para usar la aplicación Web, se debe de conectar la aplicación con Apache Marmotta e Internet con el fin de avanzar o mostrar un mensaje de error de conexión en la aplicación.
Requerimiento no funcional	RNF01
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF03
Nombre de requisito	Selección de modo
Tipo	Requisito
Características	El usuario deberá escoger el modo de uso de la aplicación: Consultar <i>dataset</i> o Realizar consulta.
Descripción del requerimiento	La aplicación Web da la opción de escoger 2 opciones de operación. En la primera, existirán varios <i>datasets</i> precargados en Apache Marmotta mientras que, en la segunda opción, el usuario podrá ingresar una consulta que quiera ser ejecutada en tiempo real. Los resultados de ambas opciones se podrán visualizar en la aplicación Web con la ayuda de la herramienta Map4RDF.
Requerimiento no funcional	RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF04
Nombre de requisito	Modo dataset
Tipo	Requisito
Características	La aplicación Web debe mostrar los <i>datasets</i> disponibles en Marmotta.
Descripción del requerimiento	Los <i>datasets</i> disponibles en Apache Marmotta serán desplegados en la aplicación Web para que usuario escoja cual quiere explorar.
Requerimiento no funcional	RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF05
Nombre de requisito	Selección <i>dataset</i>
Tipo	Requisito
Características	El usuario podrá escoger los <i>dataset</i> disponibles en el sistema para luego ser visualizados

Descripción del requerimiento	El sistema Web en conjunto con Apache Marmotta, deberán mostrar <i>datasets</i> disponibles de consultas federadas previas y posteriormente, con la herramienta Map4RDF, visualizar e interactuar con los resultados de la consulta asociada a dicho <i>dataset</i> .
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF06
Nombre de requisito	Cargar datos
Tipo	Requisito
Características	Se carga el <i>dataset</i> en la aplicación Web.
Descripción del requerimiento	Una vez que el usuario seleccionó que <i>dataset</i> quiere explorar, la aplicación pedirá los datos a Apache Marmotta para que sean cargados en la aplicación.
Requerimiento no funcional	RNF01
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF07
Nombre de requisito	Modo consulta
Tipo	Requisito
Características	La aplicación Web debe mostrar una caja de texto donde el usuario ingresará una consulta.
Descripción del requerimiento	En este modo, el usuario deberá ingresar una consulta que en una caja de texto cargada en la aplicación Web.
Requerimiento no funcional	RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF08
Nombre de requisito	Enviar consulta
Tipo	Requisito
Características	Botón que le permita al usuario enviar la consulta que quiera ser ejecutada.
Descripción del requerimiento	Una vez terminada la tarea de escribir una consulta por parte del usuario, se deberá enviar la consulta escrita a Apache Marmotta.
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF09
Nombre de requisito	Validar consulta
Tipo	Requisito
Características	Validará la consulta enviada a Apache Marmotta.

Descripción del requerimiento	La aplicación mostrará un mensaje de error en caso de que la consulta enviada haya sido incorrecta.
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF10
Nombre de requisito	Recibir resultados
Tipo	Requisito
Características	La aplicación Web recibirá los resultados de la consulta enviada por el usuario.
Descripción del requerimiento	Si la consulta enviada no tuvo algún error, la aplicación Web recibirá los datos provenientes de Apache Marmotta.
Requerimiento no funcional	RNF01
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF11
Nombre de requisito	Visualizar datos
Tipo	Requisito
Características	El usuario podrá visualizar los resultados de la consulta ingresada o <i>dataset</i> selección.
Descripción del requerimiento	Una vez que Apache Marmotta haya terminado de hacer la consulta o de cargar el <i>dataset</i> , la aplicación web dará la opción al usuario de visualizar los datos usando la herramienta <i>Map4RDF</i> .
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF12
Nombre de requisito	Explorar datos
Tipo	Requisito
Características	La aplicación Web cargará la herramienta <i>Map4RDF</i> .
Descripción del requerimiento	Cuando se tenga el <i>dataset</i> seleccionado cargado o los resultados de la consulta, se cargará la herramienta <i>Map4RDF</i> con la que el usuario podrá visualizar e interactuar con los datos.
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF13
Nombre de requisito	Envía consulta
Tipo	Requisito
Características	El compilador envía la consulta al compilador de Apache Marmotta

Descripción del requerimiento	Cuando la consulta
Requerimiento no funcional	RNF01
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

Número de requisito	RF14
Nombre de requisito	Cargar consulta federada
Tipo	Requisito
Características	El compilador envía la consulta al compilador de Apache Marmotta
Descripción del requerimiento	Cuando se tenga el <i>dataset</i> seleccionado cargado o los resultados de la consulta, se cargará la herramienta <i>Map4RDF</i> con la que el usuario podrá visualizar e interactuar con los datos.
Requerimiento no funcional	RNF01
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial

7.2.2. Requerimientos no funcionales

Número de requisito	RNF01
Nombre de requisito	Disponibilidad
Características	El software Apache Marmotta e Internet estarán deberán estar accesibles cuando la aplicación Web lo requiera.
Descripción del requerimiento	Cada vez que la aplicación Web necesite hacer uso de algún recursos de Apache Marmotta o de Internet, deberán estar disponibles para hacer uso de ellos.
Prioridad del requisito	Alta/Esencial

Número de requisito	RNF02
Nombre de requisito	Usabilidad
Características	El usuario contará con alguna leyenda que proporcione información de cómo usar la aplicación Web.
Descripción del requerimiento	Debido a los requerimientos funcionales disponibles en la aplicación Web, se mostrará información en donde sea pertinente que le permita al usuario entender cómo debe de ser usada la aplicación.
Prioridad del requisito	Alta/Esencial

Número de requisito	RNF03
Nombre de requisito	Interfaz de la aplicación
Características	Todas las funcionalidades que ofrece la aplicación Web se podrán acceder desde la interfaz visual.
Descripción del requerimiento	La interfaz visual en la aplicación web permitirá al usuario usar la aplicación en conjunto con Apache Marmotta de manera sencilla.
Prioridad del requisito	Alta/Esencial

7.3. Interfaces de hardware

Para hacer uso de la aplicación Web será necesario tener equipo de cómputo con las siguientes especificaciones

- 4 GB en memoria RAM
- 500 MB de almacenamiento
- Mouse
- Teclado
- Adaptador de Red
- Monitor
- Procesador doble núcleo o superior

7.4. Interfaces de software

Para poder usar en la aplicación Web, se deberá contar con las siguientes características

- Sistema operativo Ubuntu
- Java JDK 6 o superior
- Servidor de aplicaciones Java (Tomcat 7.X o Jetty 6.X)
- Navegador Web (Google Chrome, Firefox, Edge, Safari u Opera)

7.5. Interfaces de comunicación

La comunicación entre la aplicación Web y Apache Marmotta serán mediante métodos HTTP y el estilo de arquitectura de software *REST*. Mientras que para llevar a cabo las consultas, se necesita una conexión a Internet.

7.6. Requerimientos funcionales

7.6.1. Requerimiento funcional 1

- Establecer comunicación: El usuario debe de inicializar la aplicación Web para poder usarla.
 - Se debe de establecer comunicación entre el software Apache Marmotta para que la aplicación funcione.

7.6.2. Requisito funcional 2

- Valida conexión: La aplicación Web debe validar la conexión
 - Para usar la aplicación Web, se debe de conectar la aplicación con Apache Marmotta e Internet con el fin de avanzar o mostrar un mensaje de error de conexión en la aplicación.

7.6.3. Requisito funcional 3

- Selección modo: El usuario deberá escoger el modo de uso de la aplicación: Consultar *dataset* o Realizar consulta.
 - La aplicación Web da la opción de escoger 2 opciones de operación al usuario. En la primera, existirán varios *datasets* precargados en Apache Marmotta mientras que, en la segunda opción, el usuario podrá ingresar una consulta para ser ejecutada en tiempo real.

7.6.4. Requisito funcional 4

- Modo *dataset*: La aplicación Web debe mostrar los *datasets* disponibles en Marmotta.
 - Los *datasets* disponibles en Apache Marmotta serán desplegados en la aplicación Web para que usuario escoja cual quiere explorar.

7.6.5. Requisito funcional 5

- Selección del *dataset*: El usuario podrá escoger los *dataset* disponibles en el sistema para luego ser visualizados.

- El sistema Web en conjunto con Apache Marmotta, deberán mostrar *datasets* disponibles de consultas federadas previas y posteriormente, con la herramienta Map4RDF, visualizar e interactuar con los resultados de la consulta asociada a dicho *dataset*..

7.6.6. Requisito funcional 6

- Cargar datos: Se carga el *dataset* en la aplicación Web.
 - Una vez que el usuario seleccionó que *dataset* quiere explorar, la aplicación pedirá los datos a Apache Marmotta para que sean cargados en la aplicación.

7.6.7. Requisito funcional 7

- Modo consulta: La aplicación Web debe mostrar una caja de texto donde el usuario ingresará una consulta.
 - En este modo, el usuario deberá ingresar una consulta que en una caja de texto cargada en la aplicación Web.

7.6.8. Requisito funcional 8

- Enviar consulta: Botón que le permita al usuario enviar la consulta que quiera ser ejecutada.
 - Una vez terminada la tarea de escribir una consulta por parte del usuario, se deberá enviar la consulta escrita a Apache Marmotta.

7.6.9. Requisito funcional 9

- Validar consulta: Validará la consulta enviada a Apache Marmotta.
 - La aplicación mostrará un mensaje de error en caso de que la consulta enviada haya sido incorrecta.

7.6.10. Requisito funcional 10

- Recibir resultados: La aplicación Web recibirá los resultados de la consulta enviada por el usuario.
 - Si la consulta enviada no tuvo algún error, la aplicación Web recibirá los datos provenientes de Apache Marmotta

7.6.11. Requisito funcional 11

- Visualizar datos: Botón que le permitirá al usuario avanzar a la ventana de visualización.
 - Dependiendo de la selección del *dataset* por parte del usuario, la aplicación web actualizará la ventana donde se podrán visualizar los datos del *dataset* según lo que el usuario haya seleccionado.

7.6.12. Requisito funcional 12

- Explorar datos: La aplicación Web cargará la herramienta *Map4RDF*.
 - Cuando se tenga el *dataset* seleccionado cargado o los resultados de la consulta, se cargará la herramienta *Map4RDF* con la que el usuario podrá visualizar e interactuar con los datos.

7.7. Requisitos no funcionales

7.7.1. Disponibilidad

Cada vez que la aplicación Web necesite hacer uso de algún recursos de Apache Marmotta o de Internet, deberán estar disponibles para hacer uso de ellos.

Apache Marmotta debe de establecer conexión a Internet y a la aplicación el 99% del tiempo siempre y cuando las restricciones se cumplan.

7.7.2. Usabilidad

El usuario contará con alguna leyenda que proporcione información de cómo usar la aplicación Web.

Debido a los requerimientos funcionales disponibles en la aplicación Web, se mostrará información en donde sea pertinente que le permita al usuario entender cómo debe de ser usada la aplicación.

7.7.3. Interfaz de la aplicación

Todas las funcionalidades que ofrece la aplicación Web se podrán acceder desde la interfaz visual.

La interfaz visual en la aplicación web permitirá al usuario usar la aplicación en conjunto con Apache Marmotta de manera sencilla.

8. Diseño del sistema

8.1. Caso de uso

El caso de uso para el sistema (Figura 15)

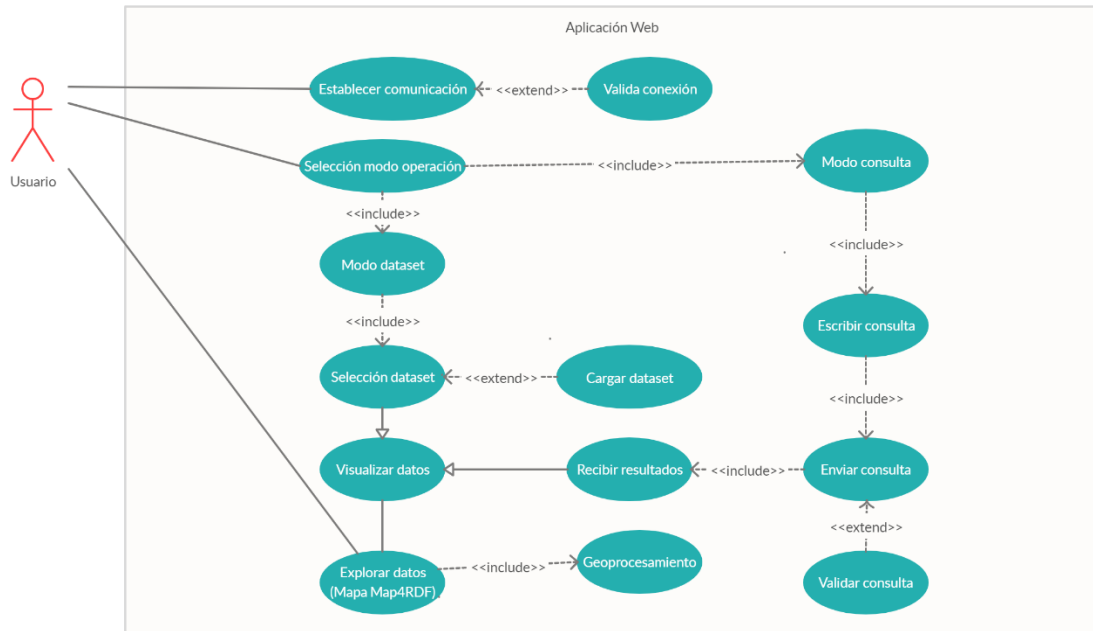


Fig. 15 Caso de uso del diseño del sistema (Aplicación Web)

8.2. Diagramas de estado

8.2.1. Diagrama de estado para la aplicación web

El diagrama de estados de la aplicación Web (Figura 16)

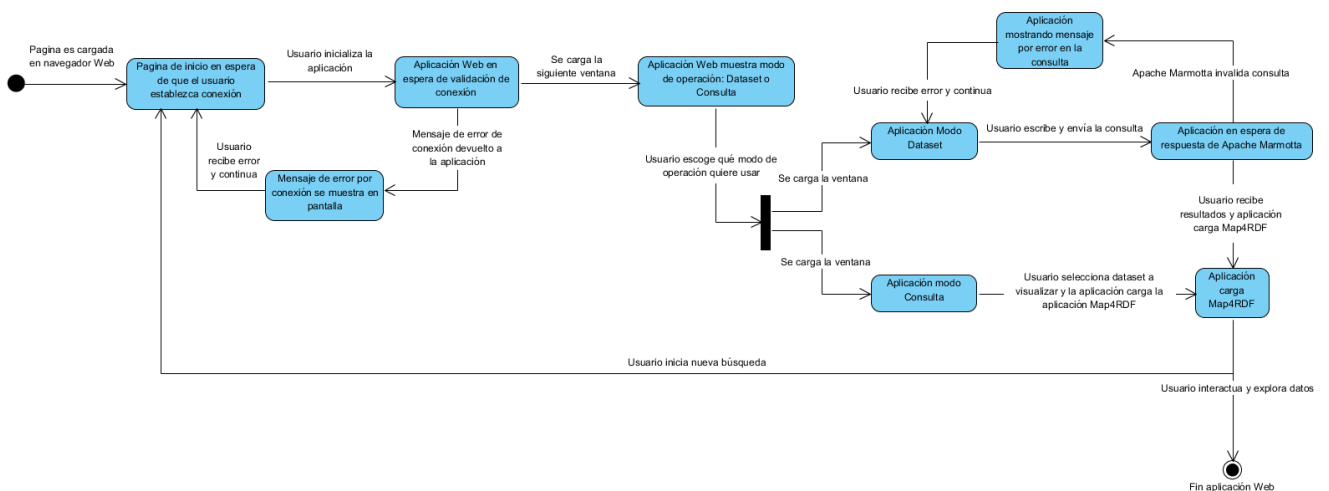


Fig. 16 diagrama de estados - aplicación Web

8.2.2. Diagrama de estado para módulo de consultas en Apache Marmotta

Diagrama de estados del Módulo de consultas (Figura 17)

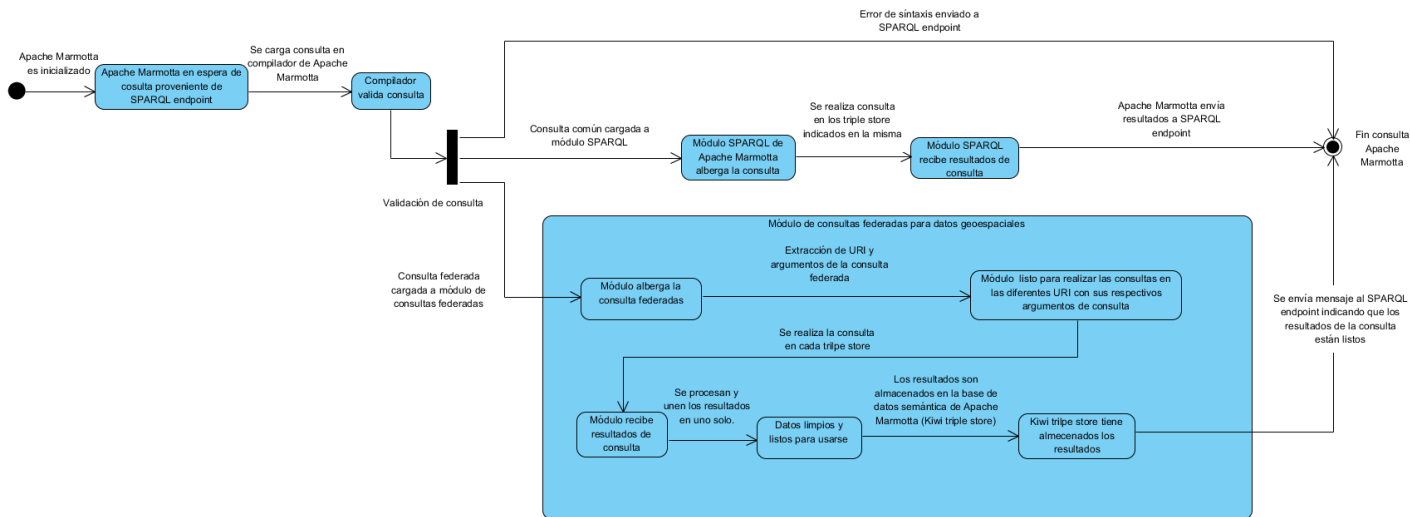


Fig. 17 Diagrama de estados - Módulo de consultas federadas

8.3. Diagramas de secuencia

8.3.1. Diagrama de secuencia para aplicación web.

El diagrama de secuencia para la aplicación Web (Figura 18)

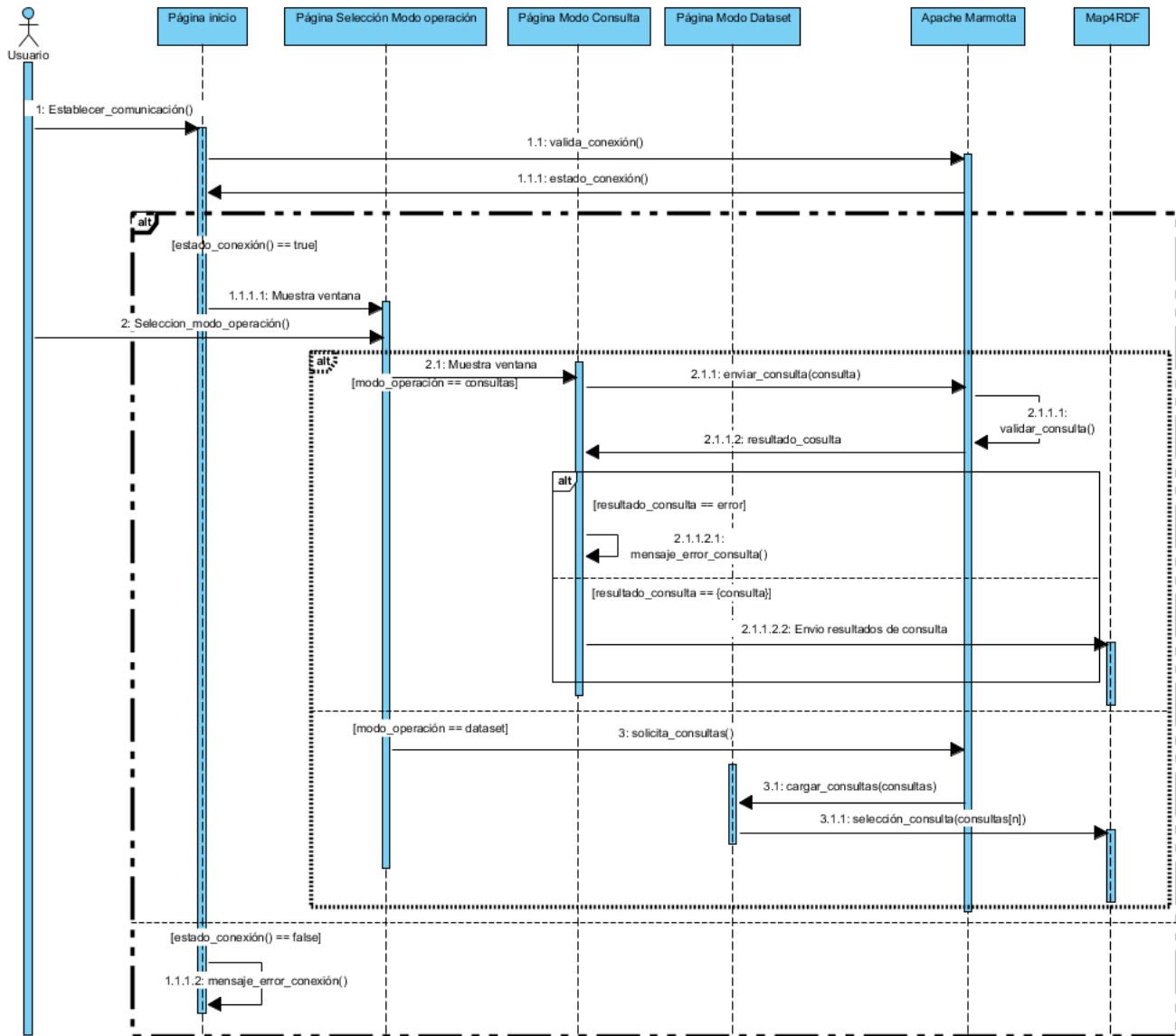


Fig. 18 Diagrama de secuencia - aplicación Web

8.3.2. Diagrama de secuencia para módulo de consultas federadas geoespaciales.
Diagrama de secuencia módulo AM (Figura 19)

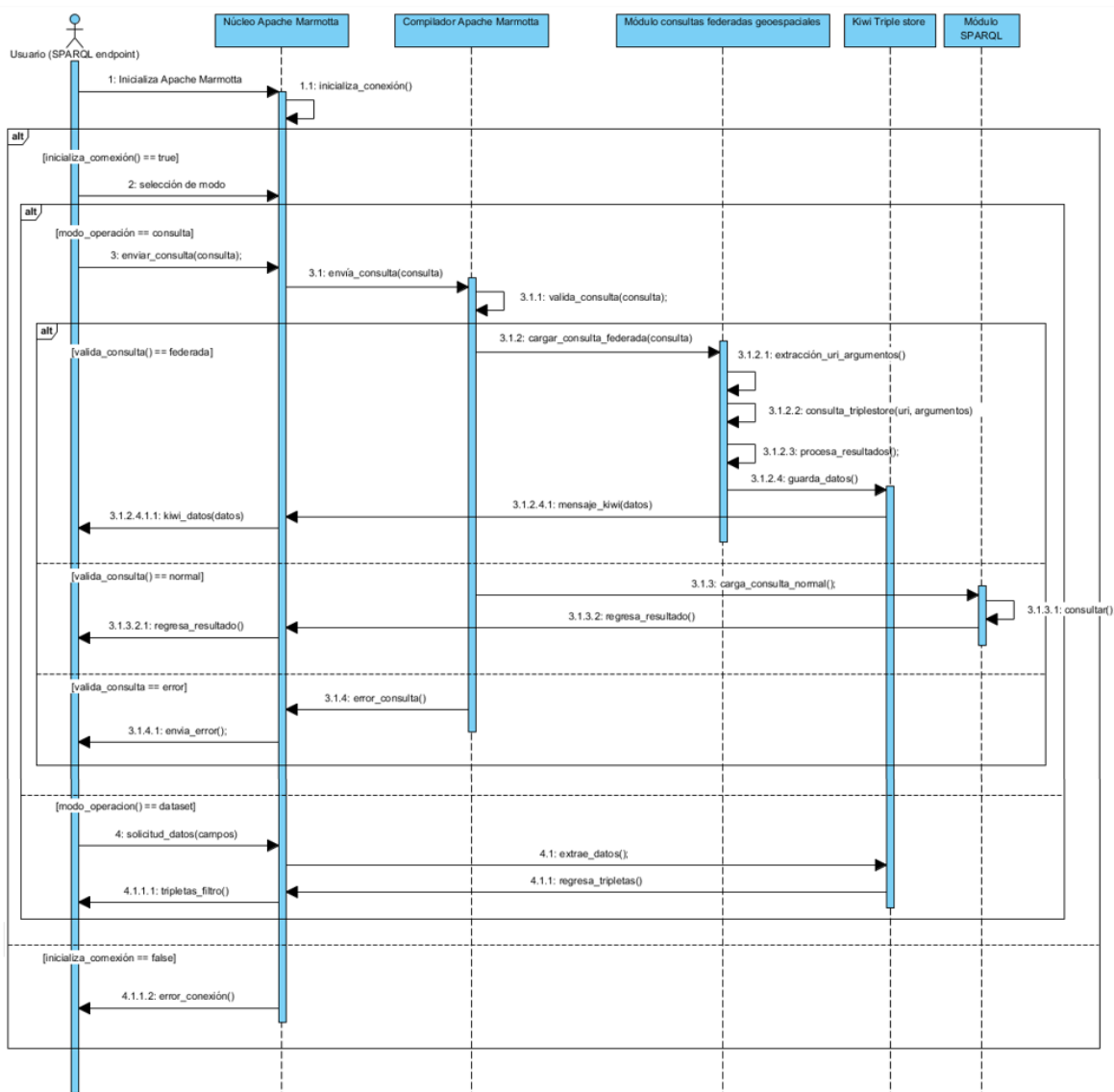
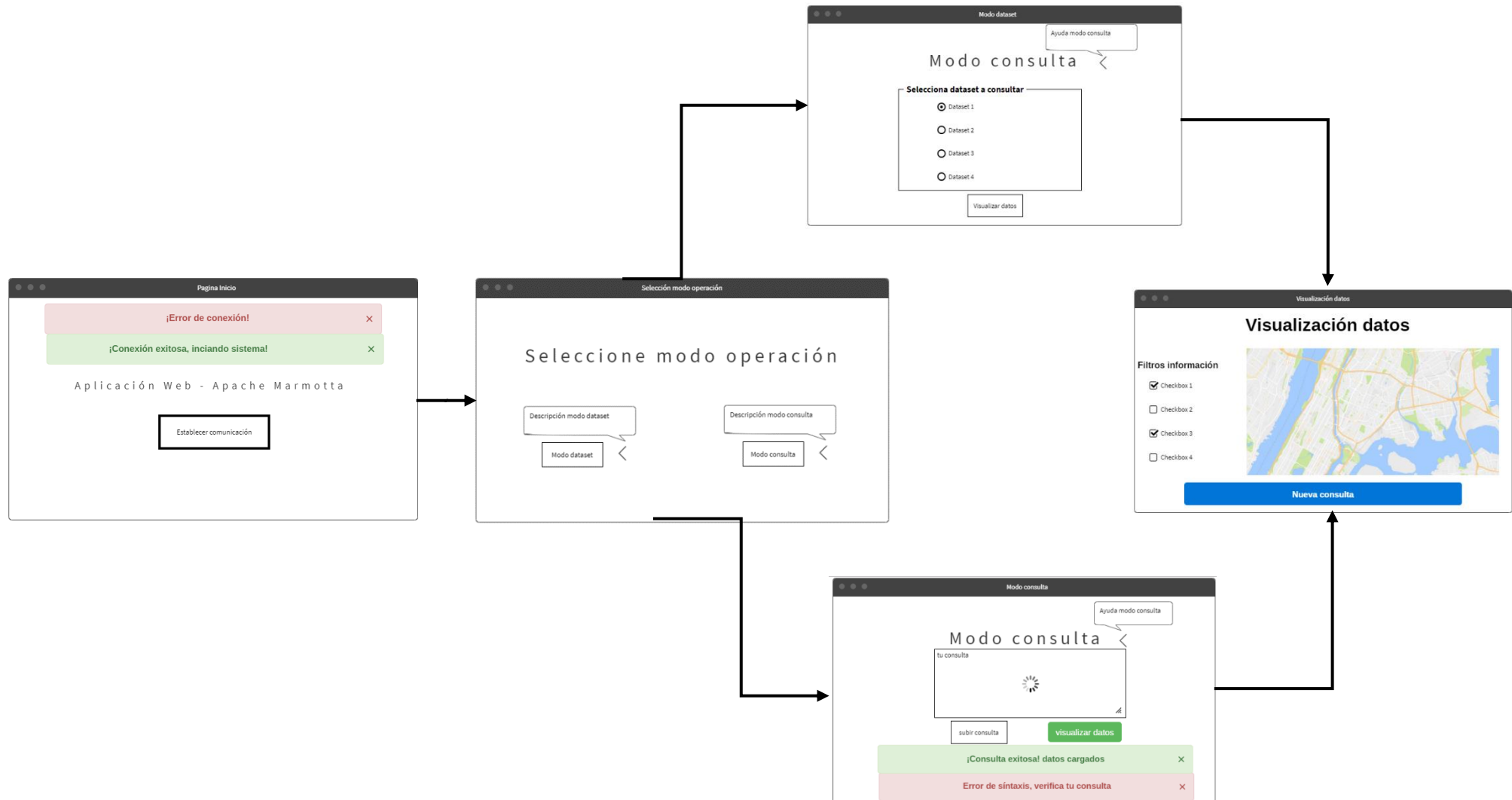


Fig. 19 Diagrama de secuencia para módulo de consultas federadas geoespaciales

8.4. Secuencia de interfaces



Referencias

- [1] T. Berners Lee, J. Hendler y O. Lassila, «The Semantic Web,» *Scientific American*, vol. 284, nº 5, pp. 34-43, 2001.
- [2] C. Bizer, T. Heath y T. Berners-Lee, «Linked Data - The Story So Far,» de *Semantic services, interoperability and web applications: emerging concepts*, USA, Information Science Reference, 2009, pp. 205-227.
- [3] W3C, «SPARQL 1.1 Overview,» 21 Marzo 2013. [En línea]. Available: <https://www.w3.org/TR/sparql11-overview/>. [Último acceso: 18 Marzo 2019].
- [4] OGC, «GeoSPARQL - A Geographic Query Language for RDF Data,» 10 Septiembre 2012. [En línea]. Available: https://portal.opengeospatial.org/files/?artifact_id=47664. [Último acceso: 24 Abril 2019].
- [5] R. Battle y D. Kolas, «Linking Geospatial Data With GeoSPARQL,» de *Semant Web J Interoperability, Usability, Appl. Accessed*, vol. 24, Arlington, 2011.
- [6] L. Lupercio, F. Baculima, M. Espinoza y V. Saquicela, «Explotación de información en el dominio geo-hídrico ecuatoriano utilizando tecnología semántica,» *Maskana*, vol. 6, pp. 69-77, 2015.
- [7] A. Marmotta, «Apache Marmotta Platform: SPARQL,» 30 Abril 2014. [En línea]. Available: <http://marmotta.apache.org/platform/sparql-module.html>. [Último acceso: 24 Abril 2019].
- [8] L. M. Vilches Blázquez y J. Saavedra, «A framework for connecting two interoperability universes: OGC Web Feature Services and Linked Data,» *Transactions in GIS*, vol. 23, nº 1, pp. 22-47, 2018.
- [9] R. Klischewski, «Semantic Web for e-Government,» de *International Conference on Electronic Government*, Heidelberg, 2003.
- [10] Apache, «Apache Marmotta,» 15 Febrero 2019. [En línea]. Available: <http://marmotta.apache.org/>. [Último acceso: 20 Febrero 2019].
- [11] O. E. ingGroup, «MAP4RDF,» 2015. [En línea]. Available: <http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/downloads/172-map4rdf/index.html>. [Último acceso: 24 Abril 2019].
- [12] W3C, «Category: Triple Store,» 5 Noviembre 2013. [En línea]. Available: https://www.w3.org/2001/sw/wiki/Category:Triple_Store. [Último acceso: 20 Febrero 2019].

- [13] O. G. Consortium, «GeoSPARQL - A Geographic Query Language for RDF Data,» 7 Julio 2001. [En línea]. Available: <https://www.opengeospatial.org/standards/geosparql>. [Último acceso: 18 Marzo 2019].
- [14] P. Haase, T. Mathäß y M. Ziller, «An evaluation of approaches to federated query processing over linked data,» de *In Proceedings of the 6th International Conference on Semantic Systems*, Walldorf, Alemania, 2010.
- [15] M. Schmidt, T. Hornung, G. Lausen y C. Pinkel, «SP² Bench: a SPARQL performance benchmark,» de *2009 IEEE 25th International Conference on Data Engineering*, Freiburg, Alemania, 2009.
- [16] C. Bizer y A. Schultz, «The berlin sparql benchmark,» *International Journal on Semantic Web and Information Systems*, vol. 5, nº 2, pp. 1-24, 2009.
- [17] N. Wiegand, R. Grove, J. Wilson y D. Kolas, «Querying Geospatial Data over the Web: a GeoSPARQL Interface,» de *Proceedings of Workshop on Managing and Mining Enriched Geo-Spatial Data*, Virginia, 2014.
- [18] R. Battle y D. Kolas, «Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL,» *Semantic Web*, vol. 3, nº 4, pp. 355-370, 2012.
- [19] C. Buil-Aranda, A. Polleres y J. Umbrich, «Strategies for Executing Federated Queries in SPARQL1.1,» de *International Semantic Web Conference*, Chile, 2014.
- [20] J. Sheridan y J. Tenninson, «Linking UK Government Data,» de *Ldow*, Reino Unido, 2010.
- [21] T. Zhao, C. Zhang, L. Anselin, W. Li y K. Chen, «A parallel approach for improving Geo-SPARQL query performance,» *International Journal of Digital Earth*, vol. 8, nº 5, pp. 383 - 402, 2015.
- [22] M. Morsey, J. Lehmann, S. Auer y A. C. Ngonga Ngomo, «DBpedia SPARQL Benchmark – Performance Assessment with Real Queries on Real Data,» de *International semantic web conference*, Berlin, Heidelberg, 2011.
- [23] N. Charlampos, D. Kallirroi, K. Kostis y K. Manolis, «Sextant: Browsing and Mapping the Ocean of Linked Geospatial Data,» de *Extended Semantic Web Conference*, Grecia, 2013.
- [24] K. Bereta, G. Xiao y M. Koubarakis, «ANSWERING GEOSPARQL QUERIES OVER RELATIONAL DATA,» *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 42, 2017.
- [25] G. A. Atemezeng y F. Amardeilh , «Benchmarking Commercial RDF stores with Publications Office Dataset,» de *European Semantic Web Conference*, 2018.
- [26] G. Garbis, K. Kyzirakos y M. Koubarakis, «Geographica: A Benchmark for Geospatial RDF Stores,» de *International Semantic Web Conference*, Berlin, 2013.

- [27] W. Beek, W. Folmer, L. Rietveld y J. Walker, «Geoyasgui: The GeoSPARQL query editor and result visualizer,» *The international Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 39, 2017.
- [28] E. Ávila Barrientos, «Linked Open Data en la Biblioteca Digital,» de *Biblioteca Digital Académica en Bibliotecología y Estudios de la Información*, México, 2013, pp. 137-152.
- [29] A. Becerril García, R. Lozano Espinosa y J. M. Molina Espinosa, «Enfoque semántico para el descubrimiento de recursos sensible al contexto sobre contenidos académicos estructurados con OAI-PMH,» *Computación y sistemas*, vol. 20, nº 1, pp. 127-142, 2016.
- [30] R. Zárate Escobedo, *Facilitador de contenido móvil para el viajero basado en servicios de localización y web semántica*, México, 2018.
- [31] D. F. Rojas Carrasco y R. Torres Covarrubias, *Recuperación de información geográfica utilizando similitud semántica*, México, 2009.
- [32] Oracle, «Oracle Spatial and Graph,» 14 Marzo 2019. [En línea]. Available: <https://www.oracle.com/technetwork/database/options/spatialandgraph/overview/index.html>. [Último acceso: 1 Mayo 2019].
- [33] SWI-Prolog, «ClioPatria: the SWI-Prolog RDF toolkit.,» 23 Agosto 2017. [En línea]. Available: <https://cliopatria.swi-prolog.org/home>. [Último acceso: 30 Abril 2019].
- [34] Kowari, «Mulgara,» 2018 Agosto 31. [En línea]. Available: <http://code.mulgara.org/projects/mulgara/wiki>. [Último acceso: 30 Abril 2019].
- [35] MarkLogic, «MarkLogic Data Hub Service,» 14 Enero 2019. [En línea]. Available: <https://www.marklogic.com/>. [Último acceso: 30 Abril 2019].
- [36] Blazegraph, «Blazegraph,» 24 Abril 2019. [En línea]. Available: <https://www.blazegraph.com/>. [Último acceso: 1 Mayo 2019].
- [37] W3C, «LinkedData,» 1 Agosto 2016. [En línea]. Available: <https://www.w3.org/wiki/LinkedData>. [Último acceso: 18 Marzo 2019].
- [38] W3C, «Semantic Web,» 9 Marzo 2019. [En línea]. Available: <https://www.w3.org/standards/semanticweb/>. [Último acceso: 18 Marzo 2019].
- [39] W3C, «RDF,» 15 Marzo 2014. [En línea]. Available: <https://www.w3.org/RDF/>. [Último acceso: 18 Marzo 2019].
- [40] W3C, «URI,» 1 Febrero 2005. [En línea]. Available: <https://www.w3.org/wiki/URI>. [Último acceso: 18 Marzo 2019].

- [41] Ontotext, «What is RDF Triple Store?,» 21 Noviembre 2016. [En línea]. Available: <https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/>. [Último acceso: 20 Febrero 2019].
- [42] DBpedia, «Virtuoso SPARQL Query Editor,» 17 Abril 2018. [En línea]. Available: <https://dbpedia.org/sparql?help=intro>. [Último acceso: 20 Febrero 2019].
- [43] D. K. Becker, Information Quality & Service Oriented Architecture, Massachusetts: MIT, 2007.
- [44] T. Berners Lee, R. Fielding, J. Gettys, J. Mogul, H. Frystyk y L. Masinter, Hypertext Transfer Protocol -- HTTP/1.1, W3C/MIT, 1999.
- [45] M. Develpers, «MDN web docs,» 15 07 2019. [En línea]. Available: <https://developer.mozilla.org/es/docs/Web/HTTP/Methods>. [Último acceso: 29 09 2019].
- [46] Codecademy, «Codecademy,» 12 08 2019. [En línea]. Available: <https://www.codecademy.com/articles/what-is-rest>. [Último acceso: 29 09 2019].
- [47] I. E. T. F. (IETF), The JavaScript Object Notation (JSON) Data Interchange Format, T. Bray, 2017.