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)

Noviembre de 2019

ÍNDICE GENERAL

Índice de figuras

Índice de tablas

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.

Capítulo I: Panorama general

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, la 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 la Web pueda ser interpretado por las computadoras a nivel semántico [2]. A partir de este acontecimiento surgen servidores de *triple store* cuya información que almacenan son tripletas 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 el protocolo y lenguaje de consultas para RDF [3], *SPARQL* por sus siglas en inglés, 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*.

Planteamiento del problema

Se ha demostrado que aproximadamente el 80% de los datos tienen relación con una ubicación geográfica [7]. Esta es una de las razones, por la que las herramientas que traten con datos geoespaciales están 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 [8]. 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*.

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. Para contextualizar a Apache Marmotta, una analogía con bases de datos SQL se muestra en la figura 1.

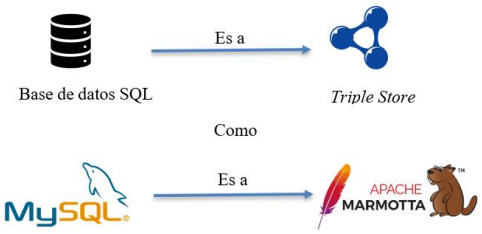

Figura 1 Analogía entre MySQL y Apache Marmotta.

La plataforma Apache Marmotta cuenta con diversas características, las 3 relevantes para este proyecto son: es una LDP, es un *SPARQL endpoint* y también es una base de datos para tripletas RDF, *triple store*. 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.

Figura 2 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 Marmotta[1]. encerrado en un rectángulo rojo, indicando que la plataforma aún no es capaz de hacer consultas federadas.

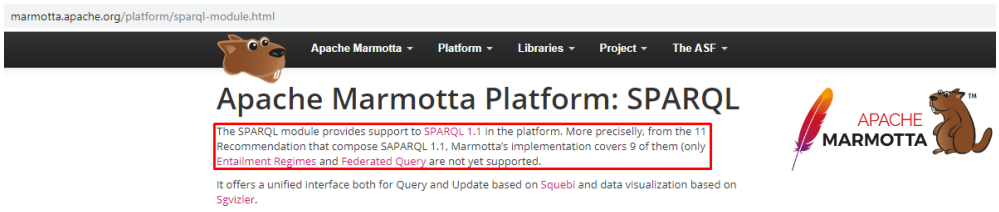

Figura 3 Apache Marmotta no soporta consultas federadas.

[1] http://marmotta.apache.org/platform/sparql-module.html

Sin embargo, hay aspectos a cumplir para que este propósito sea realidad. El primer punto por tratar es que Apache Marmotta [9] 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 mantenible. Esto se irá mejorando con las múltiples pruebas descritas en escenario de pruebas.

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* [10] cuya función es visualizar y explorar *datasets* RDF cuya información sea geométrica o *GeoYASGUI* [11] que es un editor y visualizador de consultas basadas en *GeoSPARQL*. 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 adicional revelará una consulta geoespacial federada frente a una consulta a un único repositorio?

Justificación

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.

Un aspecto por 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.

A pesar de que Apache Marmotta 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 [12]; las consultas federadas será la característica implementada en el presente proyecto terminal con el fin de ofrecer una herramienta *open source* a desarrolladores e investigadores que usen la Web Semántica en sus trabajos e investigaciones sin invertir dinero.

Propuesta de solución

En la actualidad, existen diversos motores *triple store,* aunque gran parte de ellos [13] 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 con una línea sólida una consulta simple a un solo *triple store*, mientras que, en las líneas punteadas representan la capacidad de consultar diversos *triple store* a la vez, funcionalidad que proporcionará el desarrollo de este proyecto.

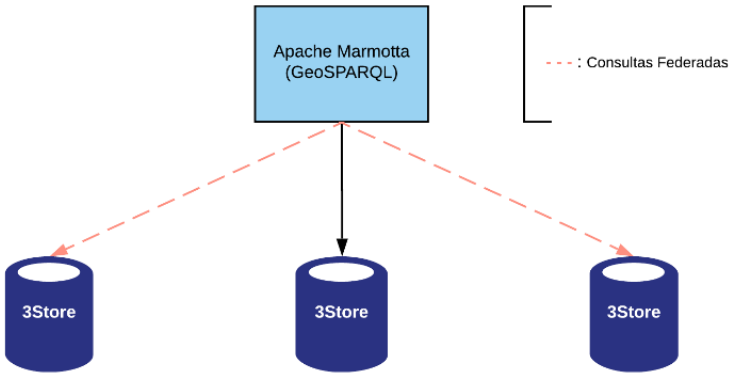

Figura 4 Diagrama de Apache Marmotta haciendo una consulta individual (línea sólida) y consultas federadas (líneas punteadas).

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.

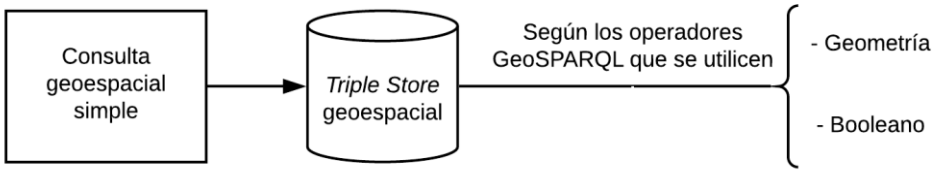

Figura 5 Consulta geoespacial simple.

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.

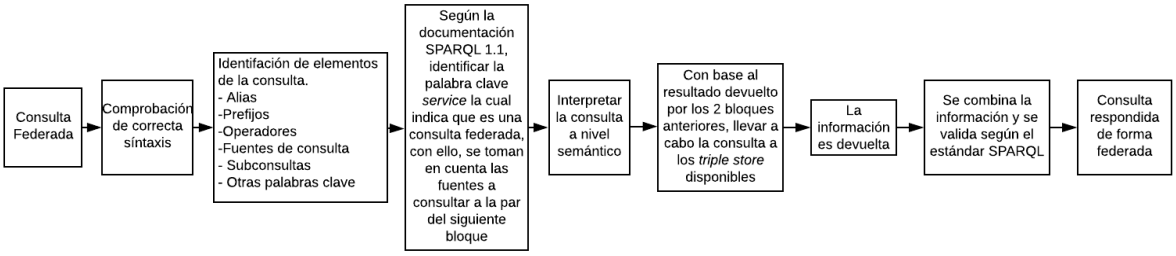

Figura 6 Diagrama a bloques de una consulta geoespacial federada.

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* [14].

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.

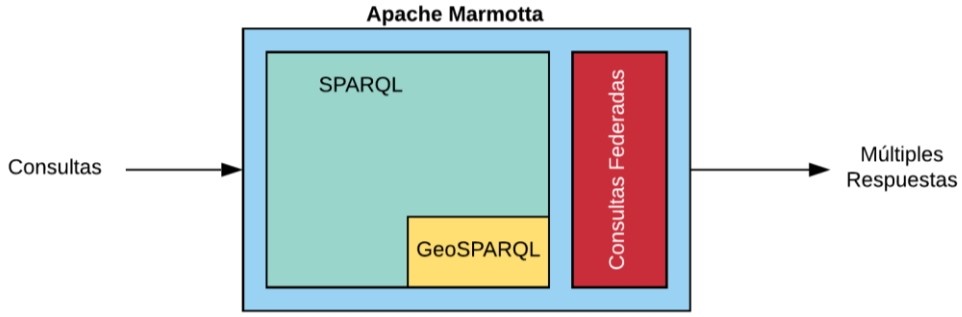

Figura 7 Diagrama a bloques de la implementación del módulo de consultas federadas.

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.

Se analizaron 3 artículos que abordan métodos de evaluación en sistemas de consultas federadas. En ellos se hablan de variables independientes y dependientes. Las variables independientes son aquellas características que debe de ser especificadas con el fin de asegurar que sean replicables en los escenarios de evaluación mientras que las variables dependientes son aquellas que serán medidas en la evaluación. La tabla 1 muestra la relación existente entre ellas

Tabla 1. Relación entre variables dependientes e independientes en el *Benchmarking*.

		Variables dependientes		
Variables independientes		Tiempo de selección del *SPARQL endpoint*	Tiempo de ejecución	Completitud de la respuesta
Consulta	Forma del plan de ejecución	Si	Si	Si
	Número de patrones de tripletas básicas en la consulta	Si	Si	Si
	Instancias y posición en las tripletas	Si	Si	No
Datos	Tamaño del RDF *dataset*	No	Si	No
	Características estructurales del *dataset*	No	Si	No
	Tipo de partición	Si	Si	Si
	Distribución de datos	Si	Si	Si
Plataforma	Memoria RAM	Si	Si	No
	Número de procesadores	Si	Si	No
	Administración de memoria caché	Si	Si	No
SPARQL endpoint	Número de *SPARQL endpoints*	Si	Si	Si
	Tipo de *SPARQL endpoints*	Si	Si	No
	Distribución de las transferencias	Si	Si	Si
	Latencia de red	Si	Si	Si
	Retraso inicial del *SPARQL endpoint*	Si	Si	No

La tabla 1 muestra cuáles son las variables que deben de ser evaluadas en el *benchmarking* (variables dependientes), y su impacto en las características específicas (variables independientes). Los parámetros para tomar en cuenta en la caracterización son los que utilizarán en el *benchmarking*:

- Tiempo de selección del *SPARQL endpoint*
- Tiempo de ejecución.
- Completitud de respuesta.

Estos parámetros están detallados en los siguientes documentos:

- *FedBench: A Benchmark Suite for Federated Semantic Data Query Processing* [15]
- *SP2Bench: A SPARQL Performance Benchmark* [16]
- *The berlin sparql benchmark* [17]

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, como otra alternativa de comprobación de funcionamiento, este trabajo presentará los resultados obtenidos usando uno de esos recursos que permita 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*.

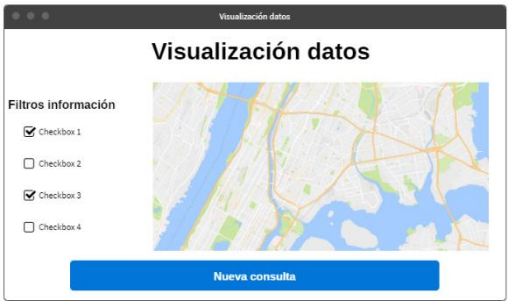

Figura 8 Propuesta de visualización.

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. Los alcances planteados para este trabajo son:

- El *triple store* Apache Marmotta en conjunto con el módulo desarrollado será capaz de hacer consultas federadas geoespaciales con base a los estándares *SPARQL* y *GeoSPARQL*.
- Serealiará un *benchmarking* que se basará en las propuestas descritas en [15], [16] y [17].
- La caracterización está delimitada por los resultados arrojados en el *benchmarking*.
- La comprobación y explotación de datos geoespaciales devueltos por el módulo de consultas federadas en Apache Marmotta se realizará mediante una aplicación Web donde el usuario podrá escoger *datasets* precargados o escribir una consulta federada geoespacial para poder visualizar los datos solicitados.
- 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.
- Una de las 2 herramientas por usar para la exploración de datos geoespaciales desplegados por el módulo de consultas federadas no será desarrollada, serán *Map4RDF o GeoYASGUI* las que se utilizarán.
- Se busca incorporar la funcionalidad a Apache Marmotta mas no competir contra otros *triple stores*.
- El proyecto no realizará una búsqueda semántica inter espacial.

Metodología

Para llevar a cabo la decisión sobre qué metodología se usaría en el proyecto, la decisión fue basada en el artículo de Alfonso Fuggetta, quien propone que el desarrollo de software *open source* debe de llevarse a cabo con metodologías que sean de rápido prototipado cuyo desarrollo sea evolutivo e incremental. Fuggetta propone que las metodologías como espiral y las ágiles son las adecuadas [18].

En la metodología tipo espiral se consideran 4 fases: Determinación de objetivos, análisis de riesgo, desarrollo y prueba y planificación. La ventajas que ofrece esta metodología es que es muy flexible en cuanto a los cambios que requiera el cliente, implementación, reutilización de software e incorporación de objetivos de calidad en el desarrollo del proyecto. Sin embargo, las desventajas que existen en la metodología es que el tiempo de desarrollo es ambiguo ya que al estar haciendo los cambios que el cliente requiera puede nunca terminar y, por ende, si no se realizó un análisis y diseño correcto del sistema, puede afectar negativamente por completo al proyecto.

Las metodologías que Fuggetta también propone son las ágiles. Si bien la más popular es *SCRUM*, se optó por considerar la metodología *Extreme Programmig (XP)* ya que se adapta al desarrollo del proyecto y al contexto de este. La metodología *XP* se adapta para equipos que cuentan con pocas personas y en proyectos donde el desarrollo del sistema sea de forma incremental.

En *XP* se consideran roles dentro del equipo: Clientes, programadores, *testers*, *coach* y mánager. En esta metodología, al igual que espiral, también se manejan 4 prácticas: Planeación, diseño, codificación y pruebas. Una de las ventajas que la metodología en espiral ofrece, *XP* también lo proporciona y es la característica de ser una metodología recursiva. Cada vez que se terminen las 4 fases, se planea la siguiente etapa del proyecto para que la siguiente etapa se concluyan las historias pendientes y así poder avanzar con las siguientes

XP está basada en historias que el cliente propone con ayuda de los *testers*, las cuales son pequeñas tareas que el sistema debe de hacer. Estas historias son desarrolladas por el equipo de programación mediante pruebas unitarias y los *testers* se encargan de validar el correcto desarrollo y funcionamiento de estas. Esta característica de *XP* fuerza al equipo a terminar todas las historias antes de continuar con las siguientes. Cada vez que se terminan las historias correspondientes a la actual fase del proyecto, se procede a integrarlo al proyecto maestro. La práctica de integración evita que los problemas de comunicación e implementación entre interfaces se presenten al implementar cada prueba unitaria aprobada por los *testers* de forma individual en vez de integrar una característica al proyecto por completo.

Ya que *XP* es la metodología que mejor se adapta a las condiciones y contexto del desarrollo del proyecto, se determinó usar esta metodología.

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.

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 lar características geoespaciales de los datos obtenidos del despliegue de consultas federadas mediante una aplicación Web para poder visualizarlos y explorarlos.

Capítulo II: Estado del arte

Los trabajos que se presentan en este capítulo 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.

Trabajos a nivel Internacional

Querying Geospatial Data over the Web: a GeoSPARQL Interface

En este artículo [19] se describe cómo es que Nancy, Ralhp 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).

Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL

Este trabajo de Robert y Dave [20] 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.

Strategies for Executing Federated Queries in SPARQL1.1

En esta propuesta [21] 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.

Linking UK Gorvernment Data

En este trabajo [22] 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.

A parallel approach for improving Geo-SPARQL query performance

Esta investigación [23] 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 aparte 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.

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

El documento [24] 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 se muestra el desarrollo del *benchmarking*, desde la generación del conjunto de datos de DBpedia, hasta los resultados obtenidos. En este trabajo 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*.

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

El documento [6] 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.

Sextant: Browsing and Mapping the Ocean of Linked Geospatial Data

Es este documento [25] presentan una herramienta Web llamada *Sextant* que permite la exploración de datos enlazados geoespaciales para la creación, compartición y edición colaborativa mediante la combinación de datos geoespaciales enlazados y otro tipo de información disponible en archivos de formato OGC.

Answering geospatial queries over relational data

En este artículo [26] 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 es 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.

Benchmarking Commercial RDF stores with Publications Office Dataset

Los autores presentan un *benchmark* [27] 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 de 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.

Geographica: A Benchmark for Geospatial RDF Stores

Se consideró por parte de los autores de este artículo [28], 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.

Geoyasgui: The GeoSPARQL query editor and result visualizer

Los autores abordan una problemática presente en los editores y evaluadores de consultas de *GeoSPARQL* [11], 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.

Trabajos a nivel Nacional

Linked Open Data en la Biblioteca Digital Semántica Académica

En este trabajo [29] 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 con las 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.

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

En este trabajo [30], 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.

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

En esta tesis [31] desarrollada en el CIC del IPN se propone 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 se hace una comparación del 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.

Trabajos desarrollados en UPIITA

Recuperación de información geográfica utilizando similitud semántica

Trabajo hecho por 2 estudiantes [32] 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.

Software similar

Actualmente, no existen más que 4 *triple store* que se asemejan en condiciones a Apache Marmotta. Estos cuatro *triple store* implementan *GeoSPARQL* y están actualmente activos. A continuación, se describirán un poco cada uno de ellos

- Parliament
    - Licencia: *BSD License.*
    - Última actualización: mayo 2019.
    - Lenguajes: Java, C++.
- GraphDB
    - Licencia: Comercial.
    - Última actualización: agosto 2019.
    - Lenguaje: Java.
- Openlink Virtuoso
    - Licencia: Comercial.
    - Última actualización: octubre 2018
    - Lenguaje: C.
- Apache Jena
    - Licencia: Apache 2.
    - Última actualización: mayo 2019.
    - Lenguaje: Java.

La tabla 2, mostrada a continuación, se muestran 4 *triple store* que existen en la actualidad. Dichas plataformas serán las que se usarán en el *benchmarking* contra *Apache Marmotta.*

Tabla 2 Tabla comparativa de software similar a Apache Marmotta.

Nombre	Consultas Federadas	GeoSPARQL	Status	Libre o pago
Parliament	Si	Si, en versión 2.7.4	Activo	Libre
GraphDB	Si	Si	Activo	Ambos
OpenLink Virtuoso	Si	Si	Activo	Ambos
Apache Jena	Si	Si	Activo	Libre
Apache Marmotta	No	Si	Activo	Libre

Tal como se mencionó en la justificación del proyecto, desarrollar herramientas que contribuyan al desarrollo de la Web Semántica es primordial, ya que son pocas las plataformas libres para los usuarios que se apegan a los protocolos *SPARQL* y *GeoSPARQL*. Solamente 2 plataformas son completamente libres; los demás *triple store* en la tabla 2 y los que no fueron incluidos en la tabla, son de pago, no se apegan a los protocolos o no tienen las funcionalidades que las plataformas de la tabla 2 ofrecen.

Capítulo III: Marco teórico

Los conceptos descritos en esta sección son fundamentales en el presente proyecto terminal para mostrar los elementos que lo componen. RDF, URI, *triple store* y *SPARQL endpoint* son conceptos fundamentales que se deben manejar para el desarrollo del módulo; tales conceptos explican los objetos con los que se trabajarán a lo largo de este proyecto terminal.

En lo que respecta a *Linked Data* y Web Semántica, estos 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.

Con respecto a *SPARQL* y *GeoSPARQL*, estos 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 dos estándares.

Web semántica

La Web semántica es el grupo de actividades que el consorcio de la *World Wide Web* con la intención de construir tecnologías que permitan publicar datos para que puedan ser legibles por computadoras a través de conceptos que definan los objetos de la Web, semántica que permita a las máquinas interpretar los conceptos de los objetos, metadatos (RDF) como recurso para describir a los objetos y ontologías para establecer la relación existente entre los objetos de la Web. 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.

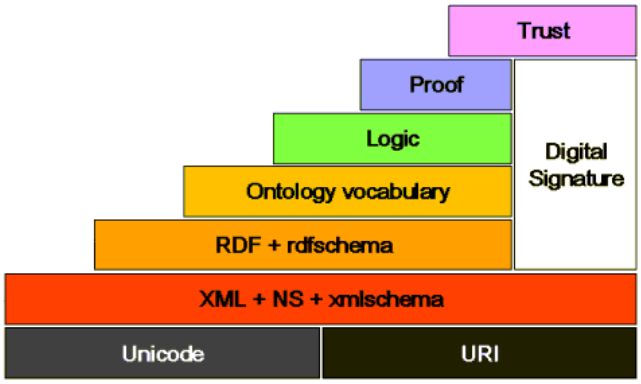

Figura 9 Diagrama propuesto por Tim Berners-Lee mostrando las tecnologías que conforma la Web Semántica.

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 al 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 [33].

Linked Data

El *Linked Data* es un conjunto de buenas prácticas para publicar y conectar datos estructurados en la web. Los principios [34] que tiene asociado son los siguientes:

- URI para identificar entidades en el mundo.
- HTTP que es el mecanismo con el que se recuperan recursos o descripciones de recursos.
- Protocolo SPARQL y archivos RDF para estructurar, consultar y enlazar los objetos presentes en la nube del *Linked Data*.

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.

En la figura 10 se observa la nube de datos *Linked Data,* la cual existe en que existe en *DBpedia*, donde cada burbuja representa un *triple store*. Cada color implica un dominio de datos distinto tales como gobierno, ciencias, multimedia entre otros. El dominio que compete al actual proyecto son los geoespaciales.

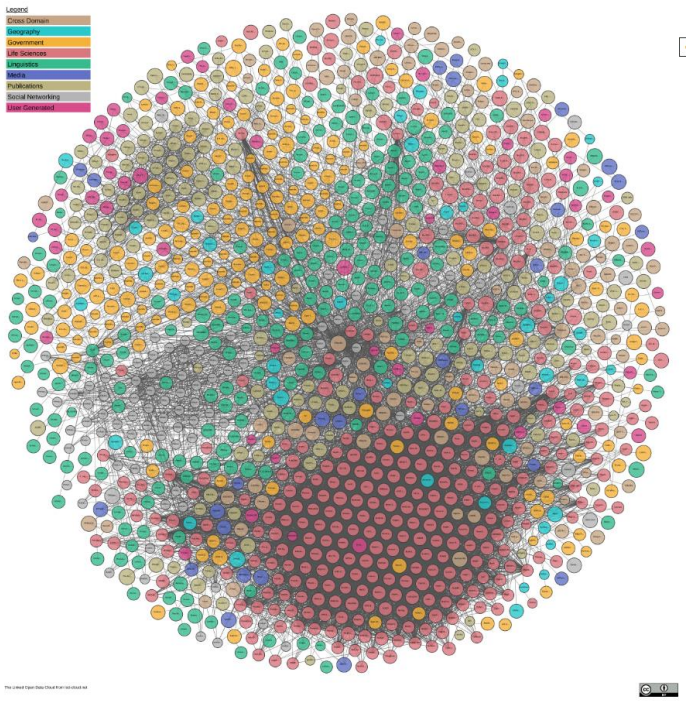

Figura 10 La nube de datos *Linked Data.*

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. La característica de RDF es que 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 y la relación existente entre ellos mediante ontologías a través de relaciones semánticas. La relación semántica que guardan los *RDF* es: sujeto, predicado y objecto. 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 [35]. La figura 11 muestra un ejemplo de cómo diferentes sitios se enlazan entre sí mediante sus respectivas URI. Los vértices son los objetos y sujetos mientras que las aristas son los predicados, cada una asociada a una URI.

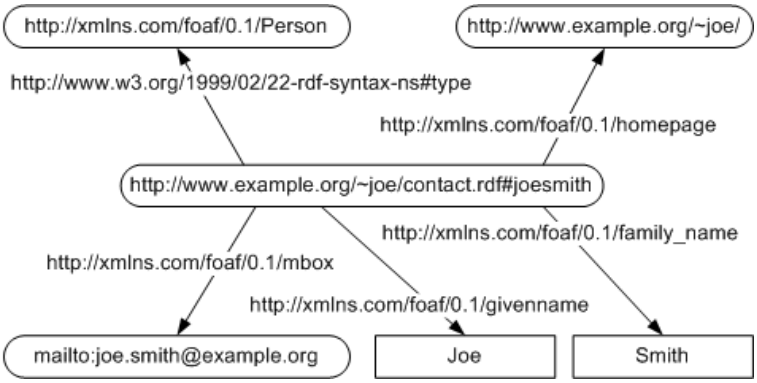

Figura 11 Grafo de RDF.

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 12, 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 [36].

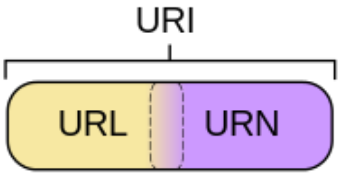

Figura 12 Diagrama de cómo está compuesto un URI.

Un ejemplo que se puede mostrar sobre la composición de una URI es la figura 13 en la que las letras en negritas denotan la ubicación donde el recurso está albergado como URL, y en letra normal, el nombre del recurso como URN.

http://sitiointernet.com/autor/bibliografia.html#posts

Figura 13 Ejemplo URI

RDF *triple store*

Es un tipo de base de datos basada en grafos de tripletas RDF [37], por lo que, 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.

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*) [3]. *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 14 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_Guízar	"Tito Guízar"@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_Valades	"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>

Figura 14 Resultado de la consulta en SPARQL.

Actualmente se encuentra en su segunda versión, *SPARQL* 1.1 [3], y ya describe una extensión para explícitamente delegar subconsultas a diferentes *SPARQL endpoint*. Esta característica es conocida como consulta federada.

SPARQL endpoint

Se le denominar *SPARQL endpoint* al identificador único de recursos, URI por sus siglas en inglés, asociado al servidor HTTP que ofrece y devuelve peticiones HTTP para peticiones provenientes de clientes que usan el protocolo SPARQL [38]. Un ejemplo de *SPARQL endpoint* es DBpedia [39] el cual es uno de los *endpoints* más famoso ya que en ella se albergan aproximadamente 4.58 millones de objetos en su base de datos de conocimiento.

GeoSPARQL

Es un lenguaje de consultas geográficas para datos RDF en la Web Semántica estandarizado por el *Open Geospatial Consortium* (OGC) [4]. *GeoSPARQL* define un vocabulario para la representación de datos en RDF y también define una extensión para el lenguaje de consultas geoespaciales *SPARQL*. Esta extensión de *SPARQL* para geo datos es útil en la solución de problemas de logística, hidrología y turismo [5].

GeoSPARQL busca relaciones topológicas entre objetos que posean datos asociados a una ubiación geográfica. Dicha búsqueda lo realiza mediante tres componentes principales:

- Definición de un vocabulario para representar características, geometrías y sus relaciones (ontologías).
- Conjunto de funciones espaciales para llevar a cabo consultas en *SPARQL.*
- Conjunto de reglas de transformación de consultas.

Un ejemplo de consulta es el siguiente: Determinar los objetos que estén contenidos en la figura A de la figura 15.

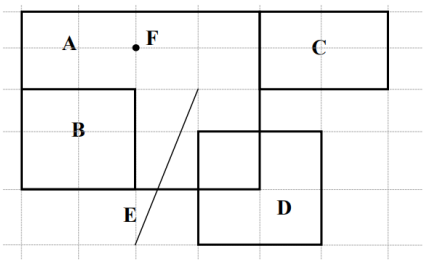

Figura 15 Representación de datos espaciales.

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 16.

?f
my:B
my:F

Figura 16 Resultado de la consulta en GeoSPARQL.

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

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 [40]. 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.

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 [41].

- 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 se quiere llevar a cabo sobre un recurso específico. Los métodos son los siguientes [42].

- GET: Pide una representación de un recurso específico. 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 del método PUT que lo hace sobre todo el recurso.

REST

La transferencia de estado representacional, REST por sus siglas en inglés, es un tipo de arquitectura que define reglas de comunicación entre sistemas computacionales en la Web [43].

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 estableces 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.

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* [44].

JSON está conformado por las siguientes 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]
      }
}
```

Capítulo IV: Análisis del sistema

El análisis de requerimientos llevado a cabo en este proyecto está basado en el estándar IEEE 830-1998 [45]. En este capítulo se presenta la descripción del proyecto especificando qué es lo que debe de hacer.

Descripción general

Se presenta una perspectiva de lo que debe de hacer el producto, las características de los usuarios del proyecto, qué restricciones existen, así como las suposiciones y dependencias que se asumen para que el proyecto funcione.

Perspectiva del producto

El módulo de consultas que se desarrollará permitirá que el software Apache Marmotta tenga la capacidad de realizar consultas federas con el objetivo de devolver una respuesta de datos geoespaciales unificada de los diversos *triple store* que se especifiquen en la consulta ingresada.

La aplicación Web permite llevar a cabo las consultas federadas geoespaciales y usará cualquiera de las 2 herramientas: *Map4RDF* o *GeoYASGUI* para visualizar e interactuar con los resultados en un mapa.

Características de los usuarios

A continuación, se presentan las características que los 2 tipos de usuarios que pueden hacer uso del sistema.

Tabla 3 Característica de usuario normal.

Tipo de usuario	Normal
Formación	Estudiante, profesor y/o investigador
Habilidades	Conocimiento *SPARQL* y *GeoSPARQL*
Actividades	Iniciar sesión, consultar, visualizar datos, filtrar datos.

Tabla 4 Características de usuario administrador.

Tipo de usuario	Administrador
Formación	Estudiante, profesor y/o investigador
Habilidades	Uso de aplicación Web
Actividades	Iniciar sesión, administrar usuarios.

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*
- El lenguaje de programación para el desarrollo del módulo que será implementado en PAache Marmotta es Java.
- 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.
- Las consultas tendrán un LIMIT que permita limitar el número de resultados por recuperar la cual estará asociado a la cantidad de memoria RAM disponible en el sistema. Si no se cuenta con este mecanismo, es probable que el sistema no pueda procesar toda la información que Apache Marmotta pueda devolver al momento de hacer una consulta.

Suposiciones y dependencias

- Las herramientas *Map4RDF* o *GeoYASGUI* 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.

Requisitos específicos

En esta sección se muestran los requerimientos funcionales y no funcionales que contiene el tanto el módulo de consultas como el de la aplicación Web.

Requerimientos funcionales

Los requerimientos funcionales para la aplicación Web son los siguientes:

Tabla 5 Requerimiento funcional establecer comunicación

Número de requisito	RF01
Nombre de requisito	Establecer comunicación
Tipo	Obligatorio
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 e Internet con la aplicación Web ya que, sin ellos la aplicación Web no puede funcionar.
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	Alta/Esencial

Tabla 6 Requerimiento funcional validar conexión.

Número de requisito	RF02
Nombre de requisito	Validar conexión
Tipo	Obligatorio
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	Alta/Esencial

Tabla 7 Requerimiento funcional selección modo operación.

Número de requisito	RF03
Nombre de requisito	Selección modo operación
Tipo	Obligatorio
Características	El usuario deberá escoger el modo de uso de la aplicación: Consultar *dataset* o realizar consulta.
Descripción del requerimiento	La aplicación Web da la opción de escoger 2 opciones de operación. En la primera, existirán varios *datasets* 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.
Requerimiento no funcional	RNF02 RNF03
Prioridad del requisito	Alta/Esencial

Tabla 8 Requerimiento funcional modo *dataset*.

Número de requisito	RF04
Nombre de requisito	Modo *dataset*
Tipo	Obligatorio
Características	La aplicación Web debe mostrar los *datasets* disponibles en Marmotta.
Descripción del requerimiento	Los *datasets* 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	Alta/Esencial

Tabla 9 Requerimiento funcional selección *dataset*.

Número de requisito	RF05
Nombre de requisito	Selección *dataset*
Tipo	Obligatorio
Características	El usuario podrá escoger los *dataset* disponibles en el sistema para luego ser visualizados
Descripción del requerimiento	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*.
Requerimiento no funcional	RNF01 RNF02 RNF03
Prioridad del requisito	Alta/Esencial

Tabla 10 Requerimiento funcional cargar datos.

Número de requisito	RF06
Nombre de requisito	Cargar *dataset*
Tipo	Obligatorio
Características	Se carga el *dataset* en la aplicación Web.
Descripción del requerimiento	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.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Tabla 11 Requerimiento funcional modo consulta.

Número de requisito	RF07
Nombre de requisito	Modo consulta
Tipo	Obligatorio
Características	La aplicación Web debe mostrar una caja de texto donde el usuario ingresará una consulta o ayuda que permita al usuario menos experto construir una consulta federada.
Descripción del requerimiento	En este modo, el usuario deberá ingresar una consulta que en una caja de texto cargada en la aplicación Web o en dado caso que el usuario no sea experto, el sistema ofrecerá ayuda para escribir la consulta federada.
Requerimiento no funcional	RNF02 RNF03
Prioridad del requisito	Alta/Esencial

Tabla 12 Requerimiento funcional enviar consulta.

Número de requisito	RF08
Nombre de requisito	Enviar consulta
Tipo	Obligatorio
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	Alta/Esencial

Tabla 13 Requerimiento funcional validar consulta.

Número de requisito	RF09
Nombre de requisito	Validar consulta
Tipo	Obligatorio

Características	Validará la consulta enviada a Apache Marmotta.
Descripción del requerimiento	El compilador de Apache Marmotta identificará qué tipo de consulta fue la que se ingresó. La respuesta puede ser: federada, normar o error. 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	Alta/Esencial

Tabla 14 Requerimiento funcional recibir resultados.

Número de requisito	RF10
Nombre de requisito	Recibir resultados
Tipo	Obligatorio
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	Alta/Esencial

Tabla 15 Requerimiento funcional visualizar datos.

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

Tabla 16 Requerimiento funcional explorar datos.

Número de requisito	RF12
Nombre de requisito	Explorar datos
Tipo	Obligatorio
Características	La aplicación Web cargará la herramienta *Map4RDF.*

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

Tabla 17 Requerimiento funcional iniciar de sesión.

Número de requisito	RF13
Nombre de requisito	Iniciar sesión
Tipo	Obligatorio
Características	El usuario debe de iniciar sesión para acceder al sistema.
Descripción del requerimiento	Para poder hacer uso del sistema de exploración de datos, el usuario deberá ingresar su usuario y contraseña.
Requerimiento no funcional	RNF03 RNF04
Prioridad del requisito	Alta/Esencial

Tabla 18 Requerimiento funcional iniciar de sesión.

Número de requisito	RF14
Nombre de requisito	Validar nuevo usuario
Tipo	Obligatorio
Características	Se corrobora que el correo y contraseña sean correctos.
Descripción del requerimiento	Para dar acceso al sistema, se verifica en el *backend* de la aplicación Web, que el correo y contraseña sean correctos para brindar el acceso.
Requerimiento no funcional	RNF03 RNF04
Prioridad del requisito	Alta/Esencial

Tabla 19 Requerimiento funcional validar usuario.

Número de requisito	RF15
Nombre de requisito	Registrar usuarios
Tipo	Obligatorio
Características	El administrador de la aplicación Web tendrá que registrar a los usuarios para que puedan acceder a cualquier funcionalidad de la aplicación.
Descripción del requerimiento	Para que los usuarios puedan acceder a la aplicación Web, ellos tendrán que proporcionar correo electrónico y una contraseña para que el administrador de la aplicación los registre en el sistema.
Requerimiento no funcional	RNF03 RNF04
Prioridad del requisito	Alta/Esencial

Tabla 20 Requerimiento funcional validar nuevo usuario.

Número de requisito	RF16
Nombre de requisito	Validar nuevo usuario
Tipo	Obligatorio
Características	El *backend* de la validará si los datos del nuevo usuario son válidos.
Descripción del requerimiento	Cada vez que el administrador de la aplicación Web ingrese datos para registrar a un nuevo usuario, el *backend* de la aplicación corroborará que el correo ingresado no exista.
Requerimiento no funcional	RNF01 RNF02 RNF03 RNF04
Prioridad del requisito	Alta/Esencial

Los requerimientos funcionales para el módulo de consultas federadas son los siguientes:

Tabla 21 Requerimiento funcional cargar consulta federada.

Número de requisito	RF17
Nombre de requisito	Cargar consulta federada
Tipo	Obligatorio
Características	Apache Marmotta le hace llegar la consulta federada al módulo de consultas que se va a desarrollar.
Descripción del requerimiento	Para que se puedan extraer los URI y argumentos de la consulta federada, la consulta proveniente del *SPARQL endpoint* debe de llegar al módulo de consultas federadas.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Tabla 22 Requerimiento funcional extracción URI y argumentos.

Número de requisito	RF18
Nombre de requisito	Extracción URI y argumentos
Tipo	Obligatorio
Características	Se extraen los URI y argumentos de la consulta federada
Descripción del requerimiento	Para poder llevar a cabo la consulta federada, se deben de extraer los URI de los *triple store* y argumentos de la consulta que se van a realizar.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Tabla 23 Requerimiento funcional Consulta *triple store*.

Número de requisito	RF19
Nombre de requisito	Consulta *triple store*
Tipo	Obligatorio
Características	Se consulta el *triple store* asociado a cada URI con sus respectivos argumentos.
Descripción del requerimiento	Para llevar a cabo una consulta federada, se debe de consultar a todos los *triple store* extraídos de la consulta federada para obtener respuestas con base a los argumentos también extraídos de la consulta.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Tabla 24 Requerimiento funcional procesar resultados.

Número de requisito	RF20
Nombre de requisito	Procesar resultados
Tipo	Obligatorio
Características	Los resultados retornados por los diferentes *triple store* serán unificados.
Descripción del requerimiento	El módulo que se desarrollará procesará todas las respuestas de cada *triple store* consultado con el fin de eliminar resultados repetidos y devolver una respuesta unificada.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Tabla 25 Requerimiento funcional guardar datos.

Número de requisito	RF21
Nombre de requisito	Guardar datos
Tipo	Obligatorio
Características	Los resultados de las consultas federadas se guardarán en el *kiwi triple store* para que el usuario pueda usarlos posteriormente.
Descripción del requerimiento	Por cada consulta federada exitosa, Apache Marmotta guardará el resultado de la consulta en el *triple store* de Marmotta, *kiwi triple store*, para que pueda posteriormente hacer uso de los datos de su consulta.
Requerimiento no funcional	RNF01
Prioridad del requisito	Alta/Esencial

Requerimientos no funcionales

Tabla 26 Requerimiento no funcional Disponibilidad.

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 recurso de Apache Marmotta o de Internet, deberán estar disponibles para hacer uso de ellos.
Prioridad del requisito	Alta/Esencial

Tabla 27 Requerimiento no funcional Usabilidad.

Número de requisito	RNF02
Nombre de requisito	Usabilidad
Características	El usuario contará con alguna leyenda que proporcione información de cómo usar la aplicación Web o llevar a cabo una consulta.
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. También ofrecerá ayuda para llevar a cabo una consulta federada en caso de que el usuario haya ingresado una consulta federada de manera incorrecta. De igual forma, la aplicación Web ayudará al usuario menos experto a construir una consulta federada mediante mensajes en la misma página que ofrezca ayuda al usuario, mejor conocido como *tooltips*.
Prioridad del requisito	Alta/Esencial

Tabla 28 Requerimiento no funcional Interfaz de la aplicación.

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

Tabla 29. Requerimiento no funcional Confidencialidad

Número de requisito	RNF04
Nombre de requisito	Confidencialidad
Características	La seguridad de los datos proporcionados por los usuarios estará garantizada en el sistema.
Descripción del requerimiento	La información ofrecida por los usuarios para darse de alta en el sistema estará segura para que nadie más pueda hace uso de ella.
Prioridad del requisito	Alta/Esencial

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.

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).

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.

Capítulo V: Diseño del sistema

En esta sección se especificarán las características tanto de la aplicación Web y del módulo de consultas federadas geoespaciales a desarrollar mediante diagrama de casos de uso, diagrama de clases, diagrama de estados y diagramas de secuencia. Al final de este capítulo se muestran *mockups* de la interfaz de usuario para la aplicación Web.

Caso de uso

Los diagramas de caso de uso sirven para especificar el comportamiento esperado de un sistema sin ahondar cómo se llevará a cabo. El uso de este tipo de diagramas permite representar a nivel visual y textual el diseño del sistema desde la perspectiva del usuario final.

En los diagramas de caso de uso representa la relación existente entre sistemas, usuarios y actores, pero no muestra el orden de cómo es que será ejecutado cada caso de uso.

El siguiente es el único caso de uso, la cual es para la aplicación Web. Es el único ya que es la sección del proyecto que tendrá una interacción directa con los usuarios finales. El caso de uso para la aplicación Web es la mostrada en la figura 17.

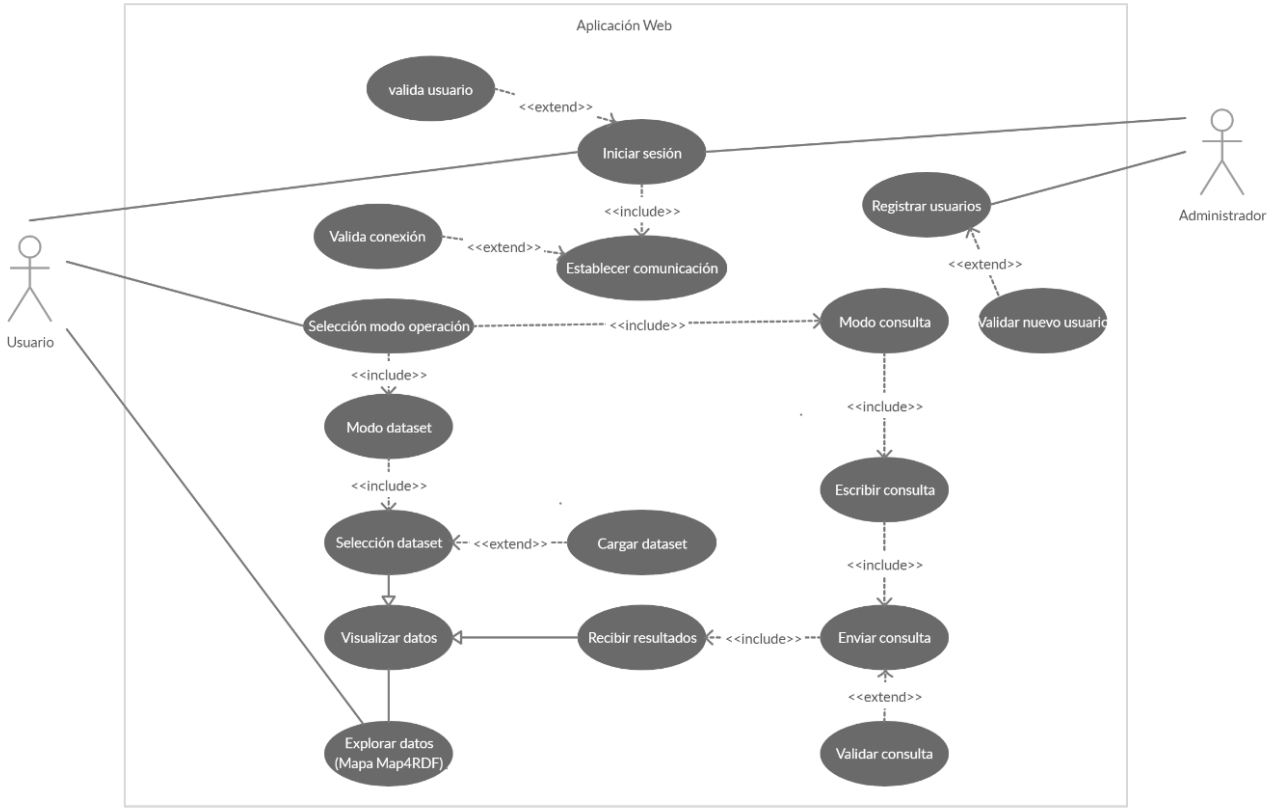

Figura 17 Caso de uso del diseño del sistema.

Diagramas de clase

Los diagramas de clase son diagramas estáticos de estructura de un sistema que permite visualizar y construir sistemas orientados a objetos el cual muestra sus clases, métodos, atributos además de la relación que existen entre los objetos.

A continuación, en la figura 18, se muestra el diagrama de clases para la aplicación Web. En ella se maneja la herencia de una clase, la cual es "Usuario" que generaliza a dos tipos de usuario: normal y administrador. Cada uno de ellos tiene sus métodos propios; el usuario "Estudiante" posee el atributo nivel_usuario que le permite al sistema reconocer si el usuario que esté haciendo uso del sistema es un usuario experto o no. La escala va del 0 al 5 donde el cero indica que el usuario no tiene conocimiento de cómo construir una consulta federada geoespacial y cinco, que indica que el usuario está completamente familiarizado con el desarrollo la consulta. Sin embargo, ambos comparten los atributos de correo y contraseña, y el método iniciar sesión que sirven para acceder a los elementos de la aplicación Web.

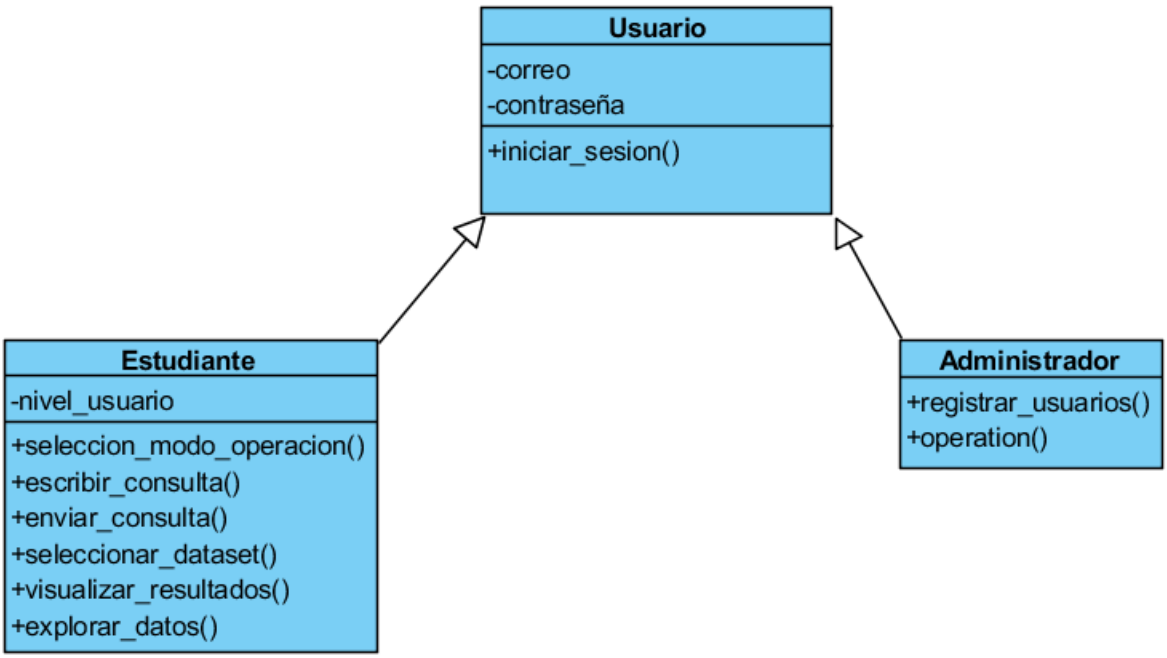

Figura 18 Diagrama de clases.

Cabe resaltar que, al igual que el diagrama de casos de uso, solo existe el diagrama de clase para la aplicación Web ya que Apache Marmotta no necesita tener información almacenada dentro de sus sistema sobre el usuario que realiza alguna operación en él.

Diagramas de estado

Los diagramas de estado son un tipo de diagrama que permite representar los diversos estados de una entidad no solo como una consecuencia de las entradas sino también de sus estados previos. De igual forma, muestra como un sistema responde a diferentes eventos dependiendo del cambio de un estado a otro.

A diferencia de los diagramas pasados, en esta sección se muestran los diagramas de estado que existen en la aplicación Web y en Apache Marmotta con el módulo de consultas federadas geoespaciales que se implementará.

Diagrama de estado para la aplicación web – Usuario

En la figura 19 se muestra el diagrama de estados presente en la aplicación Web. Con este diagrama se puede ver cómo el usuario común puede avanzar en la aplicación considerando el inicio de sesión, el selección de modo de operación y la visualización de datos.

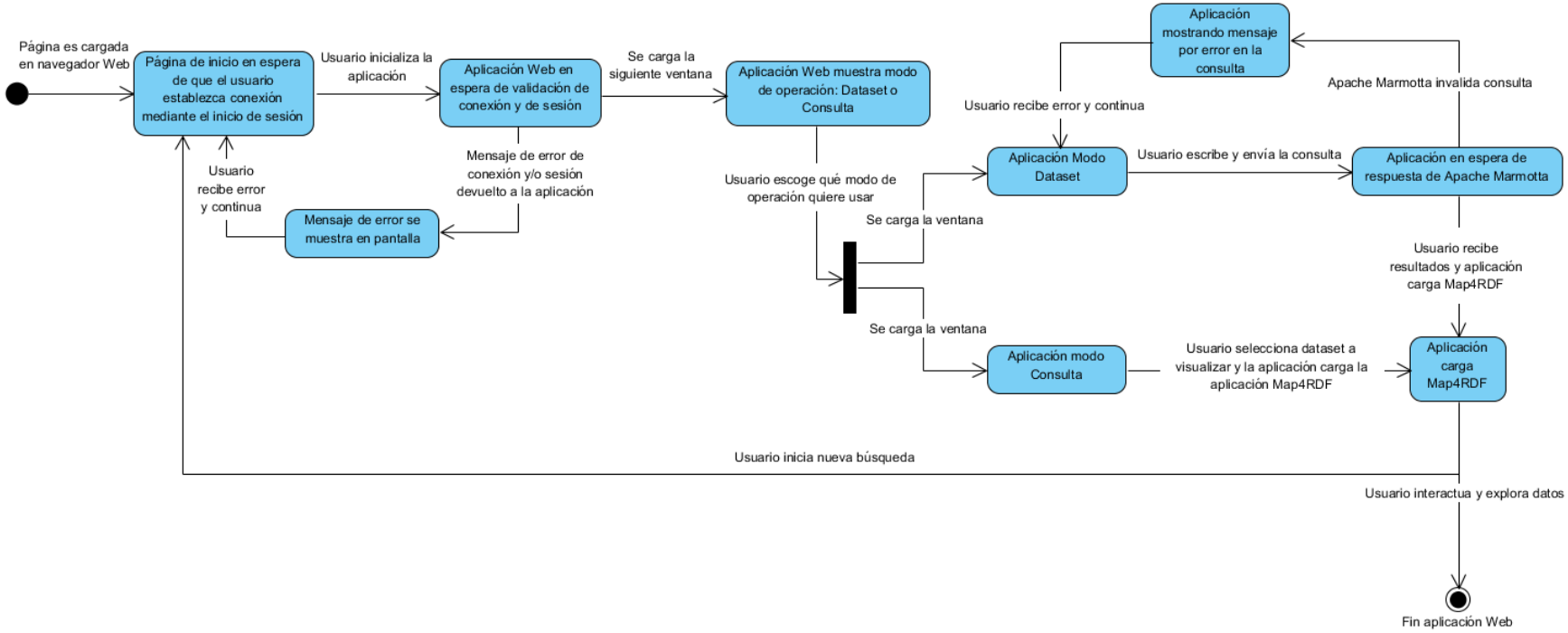

Figura 19 Diagrama de estados - aplicación Web – usuario.

Diagrama de estado para la aplicación web – Administrador

El diagrama de estados para el usuario administrador, la cual se muestra en la figura 20, detalla los estados de la aplicación Web cuando el usuario administrador está usándola.

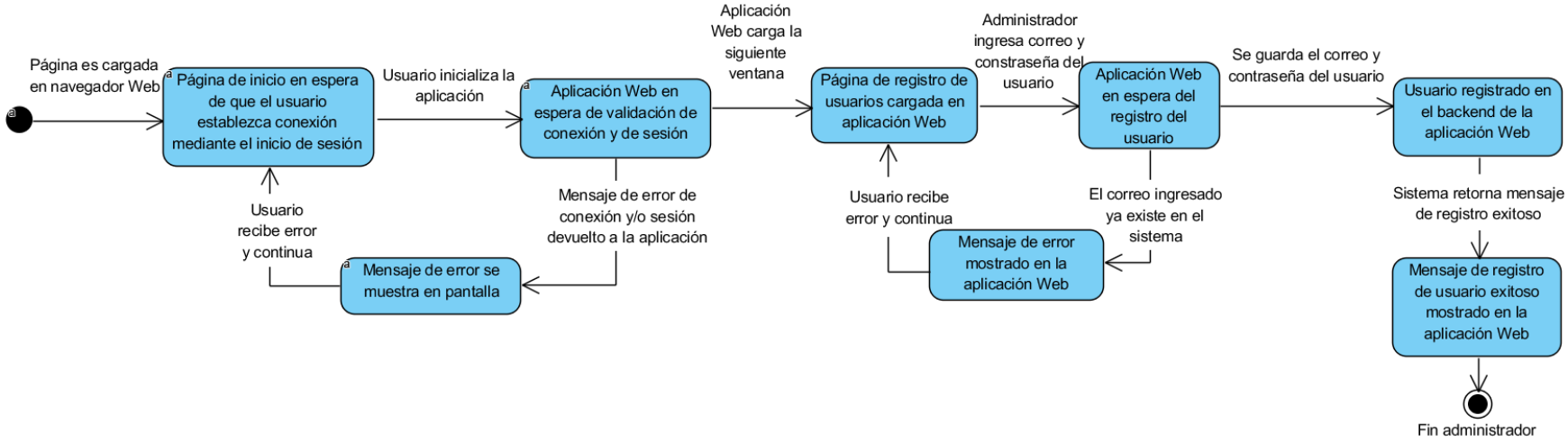

Figura 20 Diagrama de estados - Aplicación Web Administrador.

Diagrama de estado para módulo de consultas en Apache Marmotta

Para el módulo de consultas federadas geoespaciales que se implementará en Apache Marmotta, se diseñó el siguiente diagrama de estados que propone cómo es que el módulo reaccionará a las entradas y a los estados previos. Si bien el bloque azul es el módulo que se desarrollará, lo demás también son elementos que considerar ya que hay que modificar el respectivo elemento de Apache Marmotta que procesa la consulta para que sea capaz de identificar si es una consulta federada o no (Figura 21).

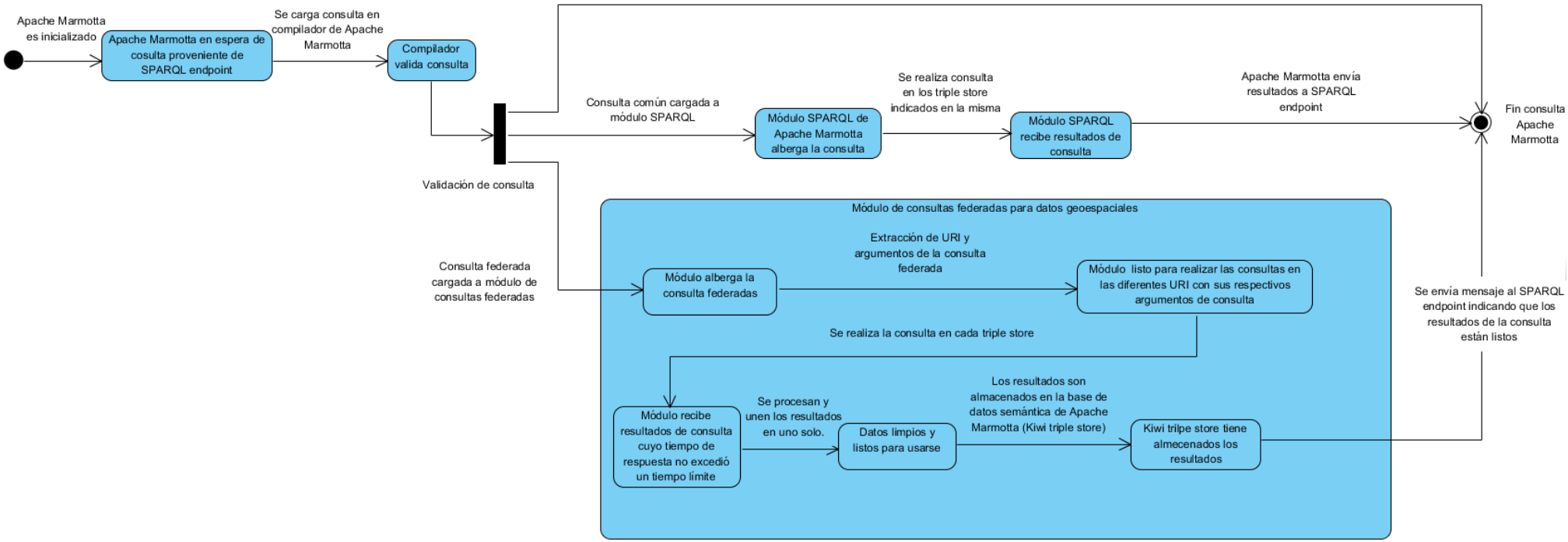

Figura 21 Diagrama de estados - Módulo de consultas federadas.

Diagramas de secuencia

Los diagramas de secuencia son un tipo de diagramas de interacción que especifica como los objetos interactúan entre sí. En estos diagramas se observan las interacciones que existen entre los objetos involucrados en el sistema mediante mensajes, que van de un objeto a otro o a sí mismos, y columnas que representan el tiempo que los objetos están presentes en el sistema. Cabe decir que a pesar de que estos diagramas involucran el tiempo, no implica la especificación de cuánto tiempo debe de existir entre mensajes.

Diagrama de secuencia para aplicación web (Usuario).

En el siguiente diagrama de secuencia, figura 22, muestra la interacción que existe entre el usuario común y la aplicación Web. Con la ayuda del diagrama de casos de uso, se visualiza como es que el usuario puede navegar en la aplicación Web desde el inicio de sesión hasta la visualización de datos.

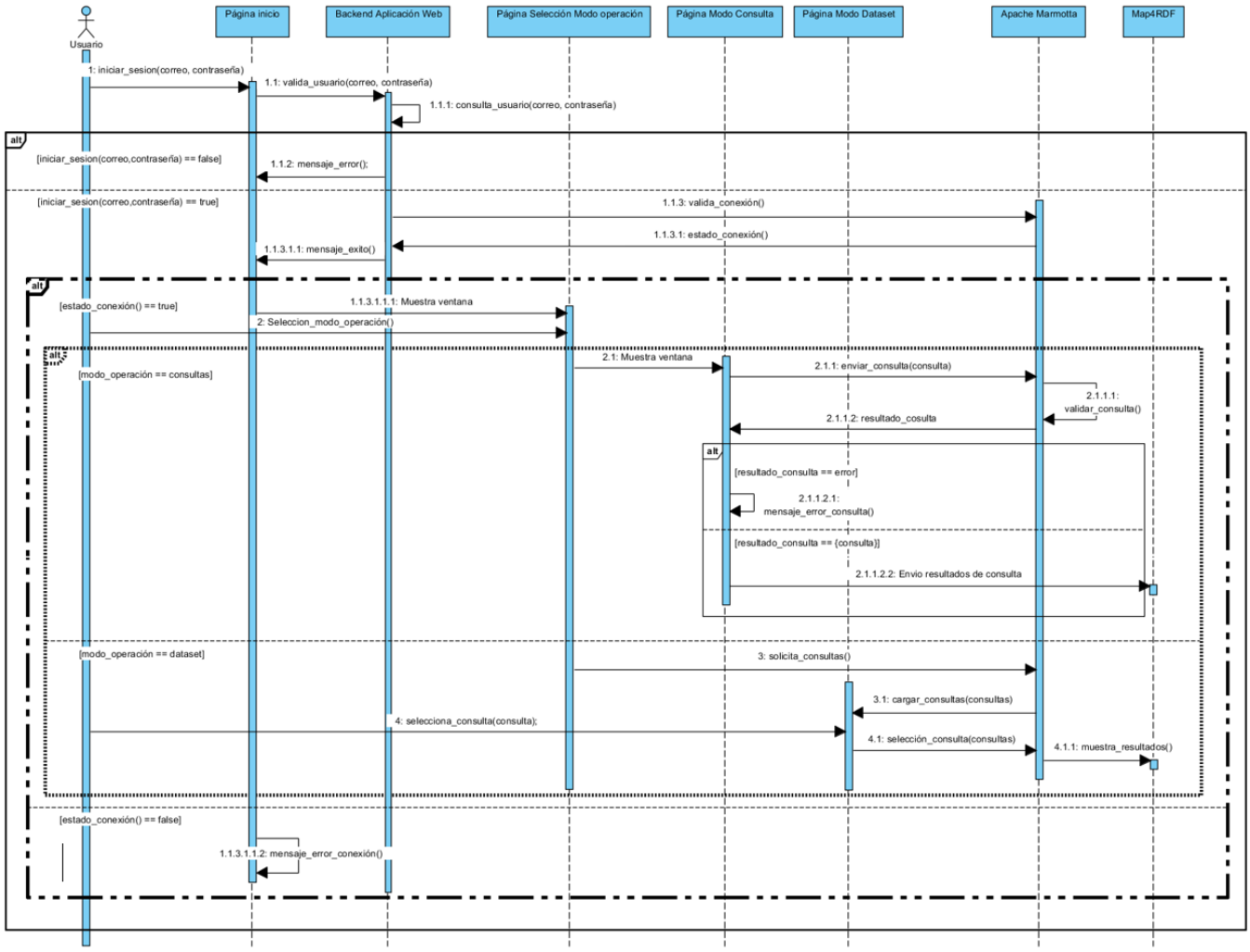

Figura 22 Diagrama de secuencia - aplicación Web – usuario común.

Con el fin de mostrar a mayor detalle el diagrama de secuencia mostrado en la figura 23, en las siguientes figuras se muestra el mismo diagrama de secuencia, pero de manera separada mostrando la referencia a la siguiente diagrama.

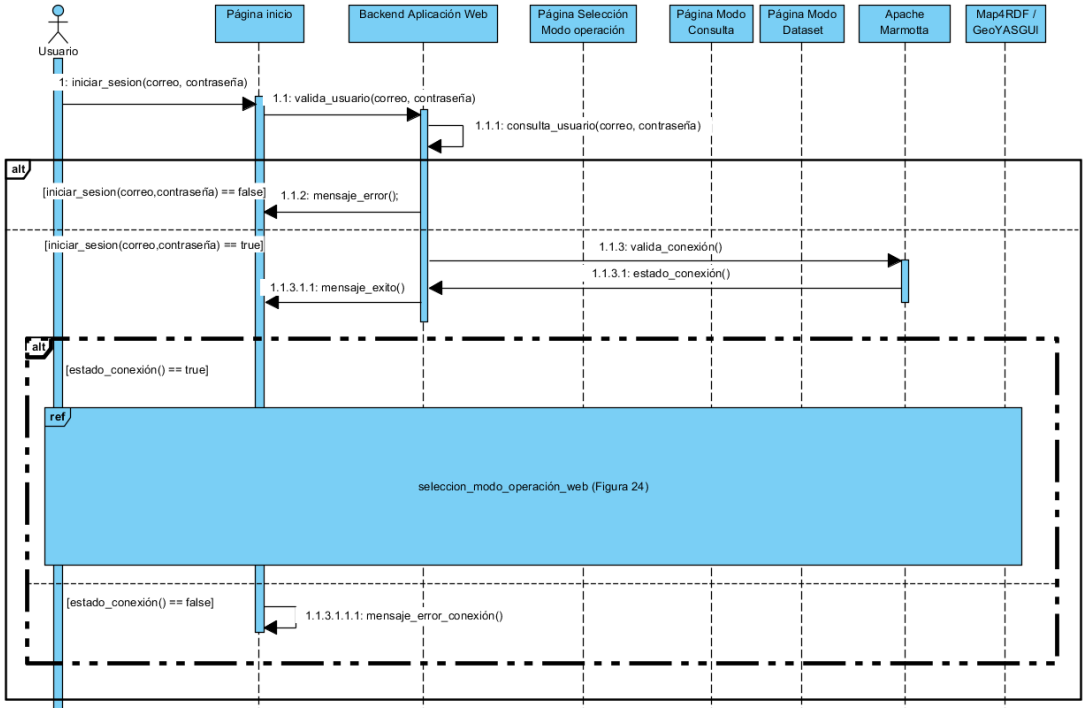

Figura 23 Diagrama de secuencia - Aplicación Web - usuario común.

En la figura 24 se encuentra el diagrama de secuencia que corresponde a la interacción del usuario con la aplicación Web al seleccionar un modo de operación.

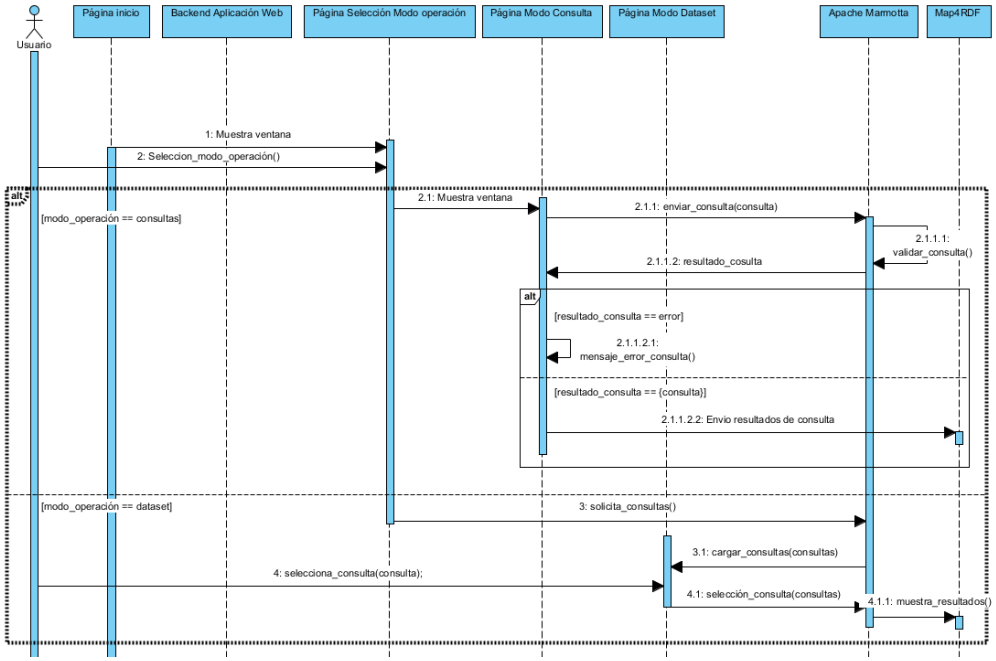

Figura 24 Diagrama de secuencia - Aplicación Web - usuario común, continuación.

Diagrama de secuencia para aplicación Web (Administrador)

Para el administrador de la aplicación Web se tiene el diagrama de secuencia en la figura 25. El diagrama muestra el proceso del administrador para iniciar sesión y para registrar a un nuevo estudiante.

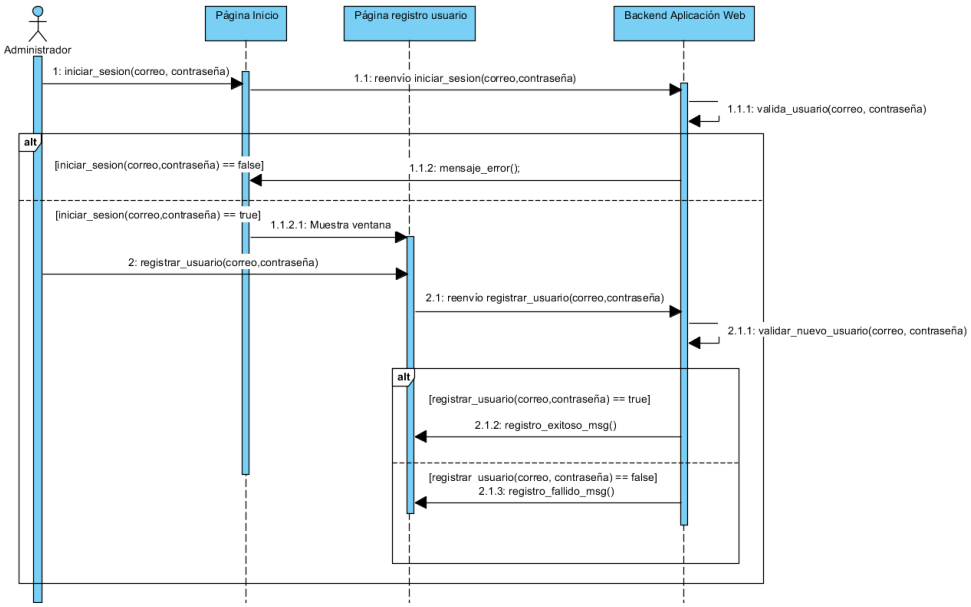

Figura 25 Diagrama secuencia aplicación Web - Administrador.

Diagrama de secuencia para el módulo.

La figura 26 muestra el diagrama de secuencia que el módulo de consultas federadas tendrá para llevar a cabo una consulta. Auxiliándose de su respectivo diagrama de estados, este diagrama detalla cómo es que se llevará a cabo el proceso.

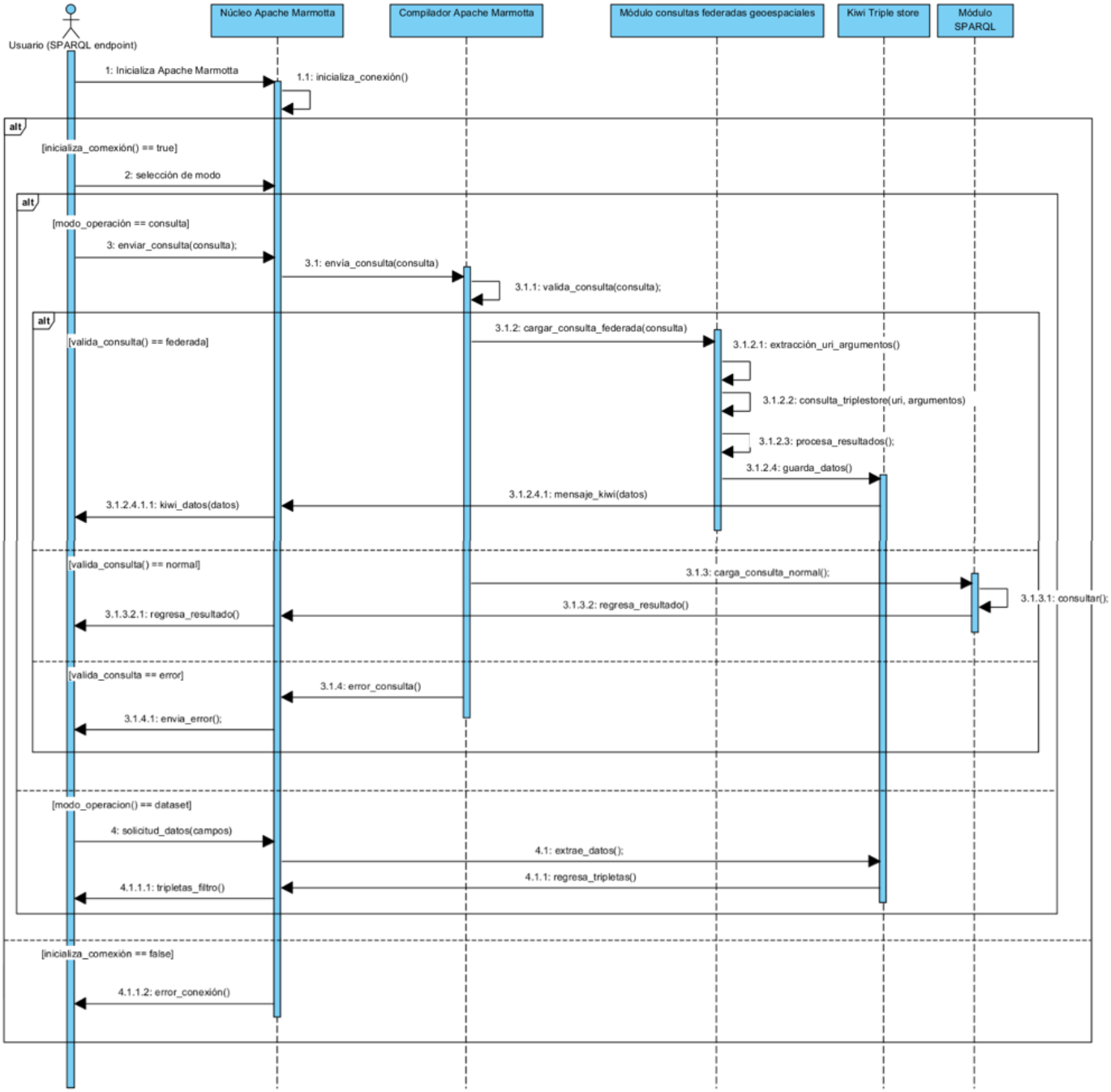

Figura 26 Diagrama de secuencia para módulo de consultas federadas geoespaciales

Con el objetivo de mostrar a detalle la figura 27, se mostrarán a continuación tres figuras que muestran el diagrama de secuencias por partes.

La figura 23 muestra el diagrama de secuencia para establecer conexión con Apache Marmotta.

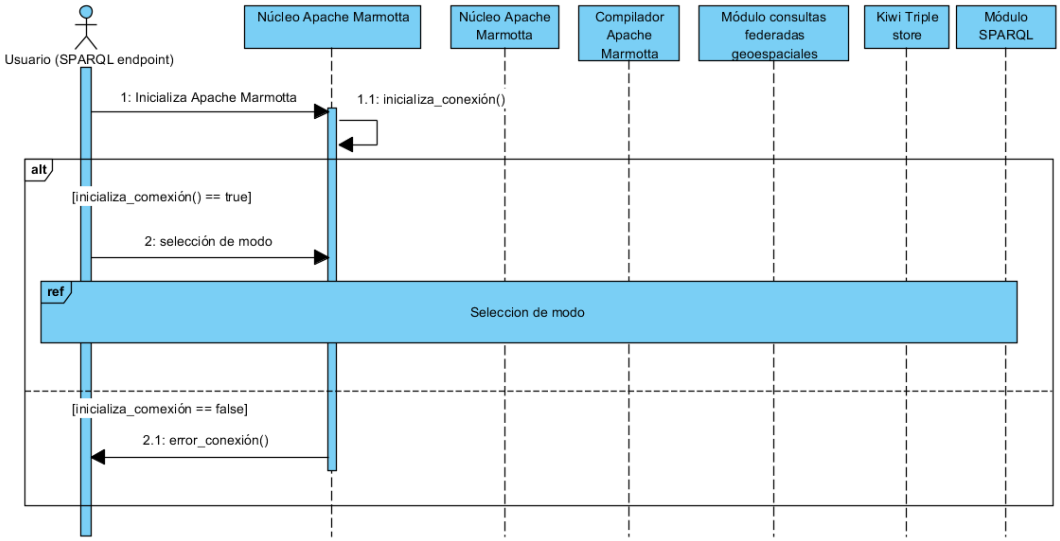

Figura 27 Diagrama secuencia inicio sesión.

La figura 28 muestra el diagrama de secuencia correspondiente a la acción de selección de modo en proveniente de la aplicación Web.

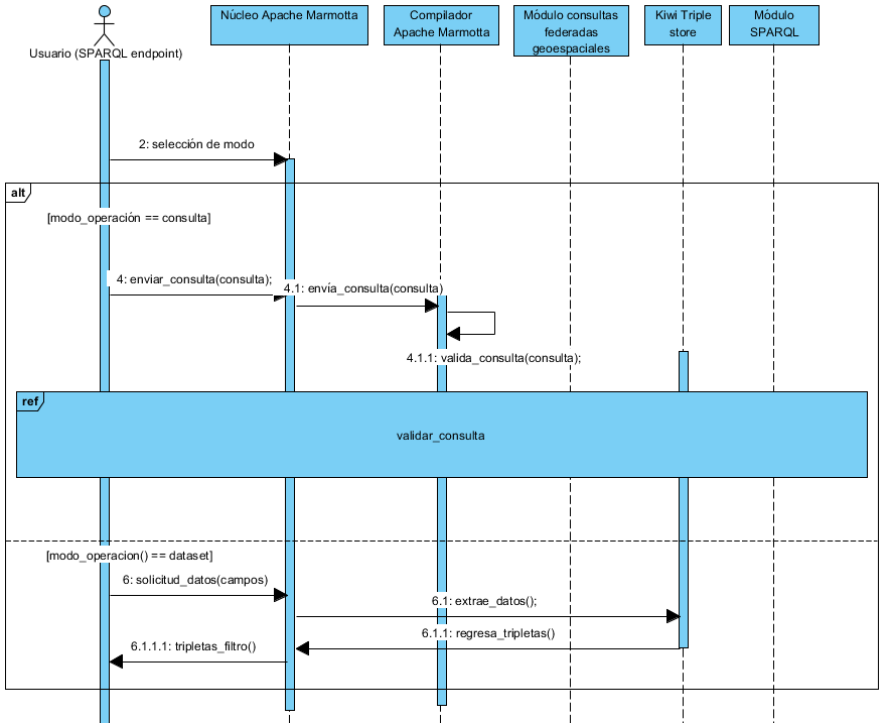

Figura 28 Diagrama de secuencia - selección de modo.

La figura 29 muestra el diagrama de secuencia correspondiente al flujo de la aplicación después de que Apache Marmotta reciba y valide la consulta ingresada por el usuario.

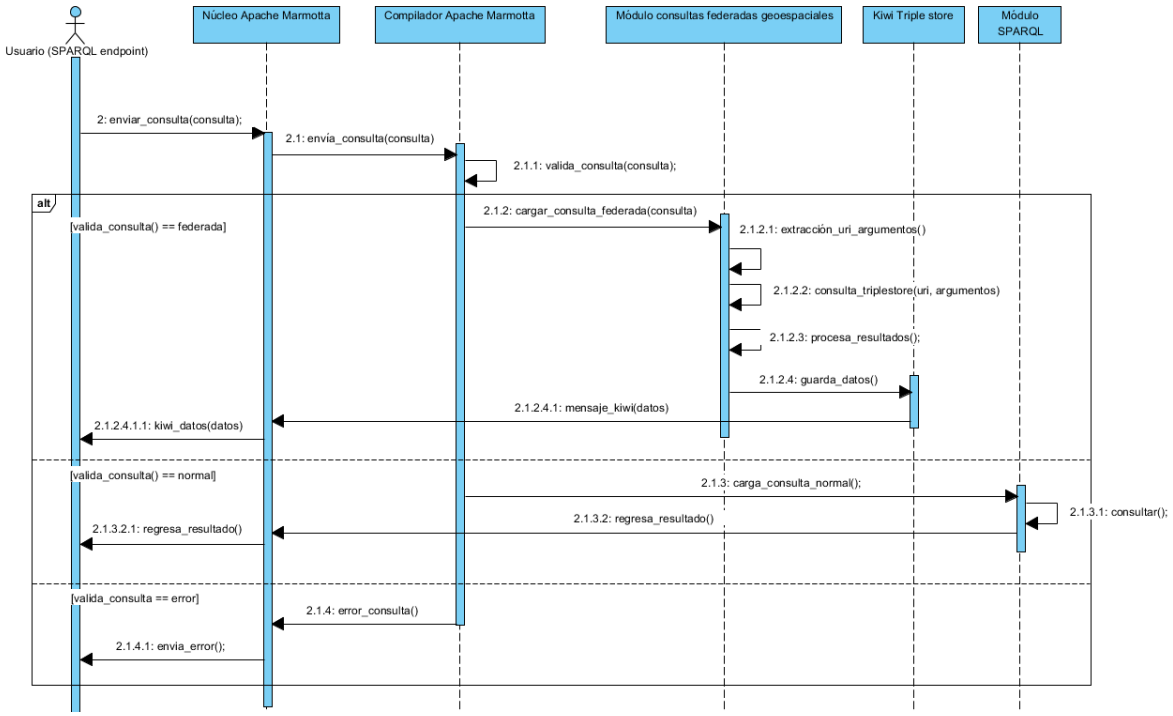

Figura 29 Diagrama de secuencia - validación consulta.

Secuencia de interfaces

A continuación, se muestra la secuencia de interfaces, *mockups*, que tendrá la aplicación Web tanto para el usuario común como para el administrador. Ambos usuarios comparten una pantalla y es la de inicio de sesión, pero para el usuario administrador solo hay una pantalla y es la de registro de usuarios mientras que para el usuario común hay cuatro páginas por la que es usuario puede navegar.

Conclusiones

Uno de los resultados que contemplaban en el desarrollo proyecto terminal I, se cumplió y es la identificación de cuál será la metodología de desarrollo del sistema de software que se utilizará en proyecto terminal II. Esta selección se considera de suma relevancia ya que con la metodología escogida, se podrán realizar todas las tareas de desarrollo e implementación de una manera adecuada ya que la metodología *XP* se adecua correctamente a la situación en la que se encuentra el proyecto, la cual es terminar el proyecto terminal II en tiempo y forma además de que los roles que debe de haber en la metodología *XP* están presentes en el equipo de desarrollo del módulo de consultas federadas geoespaciales para el *triple store* Apache Marmotta.

Tal como se contempló para la actual fase del proyecto, se identificaron cuáles son las variables que estarán involucradas en el *benchmarking* una vez implementado el módulo a desarrollar en Apache Marmotta. De igual manera, se definieron cuáles son los *triple store* que estarán involucrados en el *benchmarking*; la principal característica que comparten estos *triple store* es que son los que más se apegan a las especificaciones que los protocolos *SPARQL* y *GeoSPARQL* estipulan. El *benchmarking* dará paso a la identificación de fortalezas y debilidades de Apache Marmotta frente a sus similares y a su vez, las fortalezas definirán la caracterización del módulo que será desarrollado en proyecto terminal II.

También se estudió a profundidad el funcionamiento de Apache Marmotta. En el estudio se identificó que la arquitectura de software está basada en la arquitectura orientada a servicios, *SOA* por su siglas en inglés, y las tecnologías que hacen que Apache Marmotta funcione; básicamente Maven, Java, HTTP y REST es lo que se necesita para poder desarrollar nuevas funcionalidades en Marmotta. A su vez, conforme se empezó a trabajar con Apache Marmotta, otra tecnología que es partícipe en el desarrollo de Marmotta es GitHub ya que en la plataforma se aloja el código fuente para que desarrolladores, con el fin de mantener la idea de software *open source,* puedan usar, corregir o implementar nuevas características en Apache Marmotta.

Una vez identificados las tecnologías y la metodología que se usará en proyecto terminal II, se realizó un análisis de requerimientos de la aplicación Web y del módulo. En el análisis se describe de forma general lo que el proyecto va a hacer, las restricciones, suposiciones y dependencias que se asumen para que el proyecto funcione, así como los requerimientos funcionales y no funcionales que el proyecto deberá tener. Por último, se establecieron las interfaces de software, hardware y de comunicación. El análisis de requerimientos delimitó qué es lo que hará y lo que no también el proyecto.

Después, el diseño del sistema fue realizado. En el diseño se contemplaron 5 tipos de diagramas. El primero fue caso de uso donde se identificaron los actores y actividades involucrados en la aplicación Web. No se diseñó diagrama de casos de uso para el módulo ya que es un software que funciona como *backend* por lo que la interacción no es directa con algún usuario. El segundo diagrama es diagrama de clases, que al igual que el anterior, solo tiene diagrama para la aplicación Web; en el diagrama de clases se determinaron los dos usuarios que pueden hacer uso del sistema. Los diagramas de estados fueron de suma importancia ya que establece cómo es que la propuesta de solución se realizará tanto para la aplicación Web como para el módulo. Después, los diagramas de secuencia también fueron contemplados ya que en ellos se pudieron determinar cómo es que el usuario interactuará con el sistema, contemplando la aplicación Web como el módulo. Y, por último, la secuencia de interfaces muestra cómo se verá cada ventana por donde los usuarios podrán navegar en la aplicación Web.

Las expectativas puestas para proyecto terminal I se cumplieron ya que las actividades que se estipularon en el cronograma de actividades para ser realizadas en proyecto terminal I fueron realizadas en su totalidad. Además de las actividades, la elaboración del documento final se hizo a la par; cada parcial se estuvo trabajando en un capítulo para poderlo culminarlo a tiempo.

Con base a lo elaborado en proyecto terminal I, solo es cuestión de desarrollar el sistema con la metodología *XP*, usando las tecnologías ya identificadas y basándose completamente en el análisis y diseño del sistema hecho para desarrollar el módulo de consultas geoespaciales en el contexto de la Web de *Linked Data* para el *triple store* Apache Marmotta.

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] 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.

[8] R. Klischewski, «Semantic Web for e-Government,» de *International Conference on Electronic Government*, Heidelberg, 2003.

[9] Apache, «Apache Marmotta,» 15 Febrero 2019. [En línea]. Available: http://marmotta.apache.org/. [Último acceso: 20 Febrero 2019].

[10] 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].

[11] W. Beek, W. Folmer, L. Rietveld y J. Walker, «Geoyasgui: The GeoSPARQL query editor and result visualizer,» *The internationa Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* vol. 42, p. 39, 2017.

[12] 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].

[13] 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].

[14] 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].

[15] M. Shmidt, O. Görlitz, P. Haase, G. Ladwig, A. Shwarte y T. Tran, «FedBench: A Benchmark Suite for Federated Semantic Data Query Processing,» de *International Semantic Web Conference*, Berlín, 2011.

[16] M. Schmidt, T. Hornung, G. Lausen y C. Pinkel, «SP^2 Bench: a SPARQL performance benchmark,» de *2009 IEEE 25th International Conference on Data Engineering*, Freiburg, Alemania, 2009.

[17] 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.

[18] A. Fuggetta, «Open source software—an evaluation,» *Journal of Systems and Software,* vol. 66, nº 1, pp. 77-90, 2003.

[19] 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.

[20] R. Battle y D. Kolas, «Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL,» *Semantic Web,* vol. 3, nº 4, pp. 355-370, 2012.

[21] C. Buil-Aranda, A. Polleres y J. Umbrich, «Strategies for Executing Federated Queries in SPARQL1.1,» de *International Semantic Web Conference*, Chile, 2014.

[22] J. Sheridan y J. Tenninson, «Linking UK Government Data,» de *Ldow*, Reino Unido, 2010.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] G. A. Atemezing y F. Amardeilh , «Benchmarking Commercial RDF stores with Publications Office Dataset,» de *European Semantic Web Conference*, 2018.

[28] G. Garbis, K. Kyzirakos y M. Koubarakis, «Geographica: A Benchmark for Geospatial RDF Stores,» de *International Semantic Web Conference*, Berlin, 2013.

[29] 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.

[30] 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.

[31] R. Zárate Escobedo, Facilitador de contenido móvil para el viajero basado en servicios de localización y web semántica, México, 2018.

[32] D. F. Rojas Carrasco y R. Torres Covarrubias, Recuperación de información geográfica utilizando similitud semántica, México, 2009.

[33] W3C, «Semantic Web,» 9 Marzo 2019. [En línea]. Available: https://www.w3.org/standards/semanticweb/. [Último acceso: 18 Marzo 2019].

[34] W3C, «LinkedData,» 1 Agosto 2016. [En línea]. Available: https://www.w3.org/wiki/LinkedData. [Último acceso: 18 Marzo 2019].

[35] W3C, «RDF,» 15 Marzo 2014. [En línea]. Available: https://www.w3.org/RDF/. [Último acceso: 18 Marzo 2019].

[36] W3C, «URI,» 1 Febrero 2005. [En línea]. Available: https://www.w3.org/wiki/URI. [Último acceso: 18 Marzo 2019].

[37] 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].

[38] DBpedia, «Virtuoso SPARQL Query Editor,» 17 Abril 2018. [En línea]. Available: https://dbpedia.org/sparql?help=intro. [Último acceso: 20 Febrero 2019].

[39] DBpedia, «Learn about DBpedia,» DBpedia, 24 Octubre 2019. [En línea]. Available: https://wiki.dbpedia.org/about. [Último acceso: 10 Noviembre 2019].

[40] D. K. Becker, Information Quality & Service Oriented Architecture, Massachusetts: MIT, 2007.

[41] T. Berners Lee, R. Fielding, J. Gettys, J. Mogul, H. Frystyk y L. Masinter, Hypertext Transfer Protocol -- HTTP/1.1, W3C/MIT, 1999.

[42] 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].

[43] Codecademy, «Codecademy,» 12 08 2019. [En línea]. Available: https://www.codecademy.com/articles/what-is-rest. [Último acceso: 29 09 2019].

[44] I. E. T. F. (IETF), The JavaScript Object Notation (JSON) Data Interchange Format, T. Bray, 2017.

[45] IEEE, Especificación de Requisitos de Software IEEE 830-1998, Standards IEEE, 1998.