UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TITULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Linked Stream Data como aproximación para la representación, almacenamiento y consumo de flujos de datos bajo los principios de Linked Data

TRABAJO DE TITULACIÓN

AUTOR: Agila Espinosa, Dalton Adrian

DIRECTOR: López Vargas, Jorfe Afranio, Ing.

LOJA – ECUADOR

2017

2017

APROBACIÓN DE LA DIRECTORA DEL TRABAJO DE TITULACIÓN

Ingeniero.

Jorge Afranio López Vargas

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación **Linked Stream Data** realizado por **Dalton Adrian Agila Espinosa**, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, Noviembre del 2017

f) _

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

"Yo **Dalton Adrian Agila Espinosa** declaro ser autor del presente trabajo de titulación: **Linked Stream Data - Plataforma para gestión de rutas, buses y paradas de la Universidad Técnica Particular de Loja**, de la Titulación de **Sistemas Informáticos y Computación**, siendo **Jorge Afranio López Vargas** director del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad"

f) _

Autor: Dalton Adrian Agila Espinosa
Cédula: 0705234029

DEDICATORIA

A Dios, verdadera fuente de amor y sabiduría.

A mi madre y padre, cuyo vivir me ha mostrado que en el camino hacia la meta se necesita de la fortaleza para aceptar las derrotas y del coraje para derribar miedos.

A mis hermanos, esposa y a mi hija, que me dan el incondicional abrazo que me motiva y recuerda que detrás de cada detalle existe el suficiente alivio para empezar nuevas búsquedas.

A mis familiares, viejos amigos y a quienes recién se sumaron a mi vida para hacerme compañía con sus sonrisas de ánimo.

AGRADECIMIENTO

El presente trabajo de fin de titulación primeramente me gustaría agradecerte a ti Dios por bendecirme para llegar hasta donde he llegado, porque hiciste realidad este sueño anhelado.

A la UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA por darme la oportunidad de seguir formándome y ser una profesional.

A mi director de tesis, Ing. Jorge López por su esfuerzo y dedicación, quien con sus conocimientos, su experiencia, su paciencia y su motivación ha logrado en mí que pueda terminar mi meta con éxito.

Son muchas las personas que han formado parte de mi vida profesional a las que me encantaría agradecerles su amistad, consejos, apoyo, ánimo y compañía en los momentos más difíciles de mi vida.

Algunas están aquí conmigo y otras en mis recuerdos y en mi corazón, sin importar en donde estén quiero darles las gracias por formar parte de mí, por todo lo que me han brindado y por todas sus bendiciones. Para ellos:

Muchas gracias y que Dios los bendiga.

ÍNDICE DE CONTENIDOS

RESUMEN

Este tema de fin de titulación se enfoca en proporcionar una solución al aplicar principios de datos vinculados sobre los datos obtenidos por medio de sensores, proceso comúnmente conocido como "Linked Stream Data" (LSD), el cual se enfoca en la extracción de la metadata generada por los distintos tipos de sensores y transmitirla por un canal de comunicación.

Sobre este problema se proporciona un mecanismo basado en Servicios REST y una conexión cliente-servidor orientada a websockets, mismos que permiten identificar y acceder al Stream de Datos procedentes de la transmisión de sensores. Este proceso genera un medio por el cual los usuario podrán anclarse, compartir o extraer información de sensores, los cuales posteriormente serán plasmados en una interfaz web para la visualización y análisis de los datos y así puedan ayudar en soluciones de tránsito, evitando congestiones y ayudando para ubicación rápida de distintos medios de transporte como: recolectores de basura o servicios de venta de gas entre otros, detallando los principales requisitos para su creación y las principales razones para tomar decisiones sobre su diseño y arquitectura.

PALABRAS CLAVE: metadata, websockets.

ABSTRACT

This subject to titration focuses on providing a solution to apply principles of linked data on data obtained via sensors, process commonly known as "Linked Stream Data" (LSD), which focuses on extracting the metadata generated by the different types of sensors and transmit it through a communication channel.

A mechanism based on REST services and a client-server connection oriented to websockets are provided on this problem, which allow identifying and accessing the Data Stream coming from the transmission of sensors. This process creates a means by which the user can anchor, share or extract information from sensors which will later be reflected in a web interface for visualization and analysis of data and thus can help transit solutions, avoiding congestion and helping for quick location of different means of transport such as: garbage collectors or gas sales services among others, detailing the main requirements for its creation and the main reasons for making decisions about its design and architecture.

KEYWORDS: metadata, websockets.

CAPÍTULO 1. ESTADO DEL ARTE

1.1. Introducción

Tal y como (W3C, 2007) menciona, la Web ha cambiado profundamente la forma en la que nos comunicamos, hacemos negocios y realizamos nuestro trabajo. La comunicación prácticamente con todo el mundo en cualquier momento y a bajo coste es posible hoy en día. Podemos realizar transacciones económicas a través de Internet. Tenemos acceso a millones de recursos, independientemente de nuestra situación geográfica e idioma. Todos estos factores han contribuido al éxito de la Web. Sin embargo, al mismo tiempo, estos factores que han propiciado el éxito de la Web, también han originado sus principales problemas: sobrecarga de información y heterogeneidad de fuentes de información con el consiguiente problema de interoperabilidad.

Basado en los cambios mencionados y considerando la forma en que la información compartida puede ser interpretada por los ordenadores por cómo se vinculan y se exploran, el presente trabajo de fin de titulación propone el desarrollo e implementación de una plataforma web y móvil, que tienen como objetivo proveer una herramienta a los usuarios que cuenten con un dispositivo móvil que integre sensores tales como: GPS, giroscopio y de batería, permitiendo que cada uno de estos compartan sus datos y metadatos.

Se plantea esta solución para beneficiarnos de los distintos tipos de sensores integrados en los dispositivos móviles, generando una red de sensores que permita brindar un análisis de los datos enfocado a posiciones geo-referencias y dar soluciones como: mejores rutas para evitar tráfico, sitios de interés o de reunión concurrentes, ubicación exacta para servicios que se realizan a través de un medio de transporte como: recolección de basura, servicio para venta de gas, medios de transporte que basados en estos sensores ayuden a que los usuarios de la plataforma se beneficien de la data proporcionada.

Considerando la gran acogida de Linked Data que consiste en enlazar distintos tipos de información, se ha tomado los principios que la acogen para enlazar la información de los sensores con dataset ya existentes y así los usuarios puedan aprovecharse de la información aspirando a que en un futuro puedan desarrollar aplicaciones que ayuden con el análisis de los datos que la red de sensores proporciona.

Conforme se ha realizado la investigación de herramientas se ha establecido lo necesario para el desarrollo e implementación de una plataforma que nos ayude con la

recolección y visualización de la data y metadata de sensores integrados a los dispositivos móviles, mediante herramientas open source para la visualización de datos, contando con diseños más interactivos para el usuario y así permita mejorar la representación de información.

1.2. Justificación

El problema se basa en que no se cuenta con un medio de extracción de datos de sensores para aprovechar la data y metadata generada por estos, con esto no se puede realizar un análisis de esta información para proporcionar solución a distintos problemas habituales como el tráfico, rutas que se debe tomar para llegar a un destino específico, lugares de importancia y posicionamiento exacto de distintos tipos de servicio de medios de transportes (vehículos particulares, buses particulares, buses urbanos, buses interprovinciales, bicicletas, vehículos de entrega de gas entre otros).

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar e implementar bajo los principios de Linked Data, un modelo para el trabajo con información producida con Stream de datos.

1.3.2. Objetivos Específicos

- Proveer información geo referenciada a toda la comunidad tal como: rutas generadas por medio de la captura de puntos geo referenciados a través de los distintos usuarios.
- Desarrollar un conjunto de visualizaciones que muestren de forma atractiva los datos.
- Desarrollar mecanismos que permitan publicar los flujos de información (rutas).
- Desarrollar mecanismos que permitan observar los datos geo referenciados en tiempo real.

1.4. Antecedentes

En los últimos años la tecnología ha venido dando grandes resultados en distintos ámbitos sociales y culturales, enfocándonos en el ámbito social han empezado a ser parte de nuestra vida cotidiana, plataformas basadas en redes sociales como Facebook, Twitter, LinkedIn, entre otros.

En vista del gran impacto que han generado a nivel mundial se propone aplicar la estrategia que usan estas plataformas, haciendo uso de los usuarios finales para la obtención de información, misma que pueda ser compartida, enriqueciéndose de los datos proporcionados y generando información geo-referenciada para que sea analizada y pueda presentada lo más entendible para la toma de decisiones.

CAPÍTULO 2. MARCO TEÓRICO

2.1. Web Semántica

Así como (W3C, 2007) denota a la Web Semántica como una Web extendida, dotada de mayor significado en la que cualquier usuario en Internet podrá encontrar respuestas a sus preguntas de forma más rápida y sencilla gracias a una información mejor definida. Al dotar a la Web de más significado y, por lo tanto, de más semántica, se pueden obtener soluciones a problemas habituales en la búsqueda de información gracias a la utilización de una infraestructura común, mediante la cual, es posible compartir, procesar y transferir información de forma sencilla. Esta Web extendida y basada en el significado, se apoya en lenguajes universales que resuelven los problemas ocasionados por una Web carente de semántica en la que, en ocasiones, el acceso a la información se convierte en una tarea difícil y frustrante.

Según (Berners-Lee & Tim, 2010), la arquitectura de la Web Semántica se podría representar de la siguiente forma:

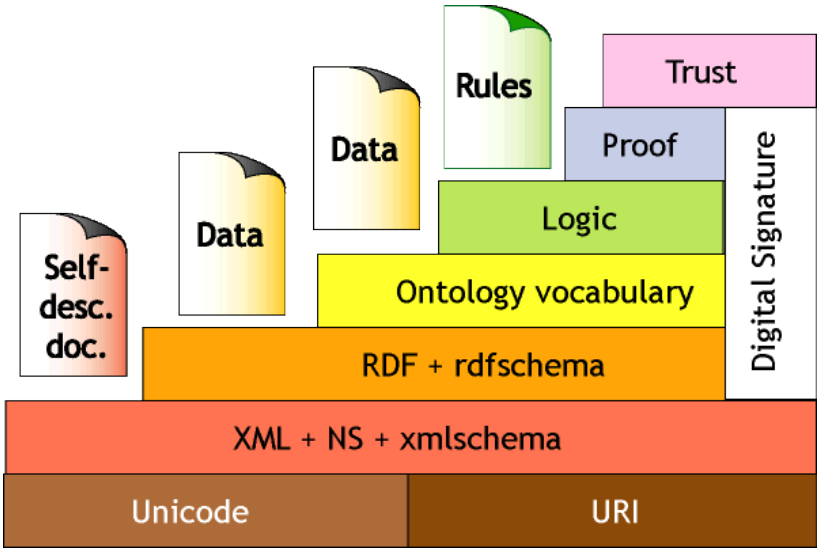

Figura 1. Arquitectura Tecnológica de la Web Semántica
Fuente: Recuperado de http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide11-0.html
Elaborado por: (Tim Berners-Lee, James Hendler, 2001)

La mayoría de los sitios web están construidos en lenguaje HTML con marcas o etiquetas que se muestran cuando se visualiza el código fuente, pero que permanecen ocultas en la visualización normal de los navegadores y que contienen información sobre el contenido de la página, enlaces hacia otras páginas, formatos de letra, color, párrafos, imágenes, vídeos, etc. Los orígenes de la Web se basaron en el carácter abierto y universal de la base de la Web: el lenguaje HTML, y el empleo de archivos ASCII y los gráficos GIF y/o JPG. Esto permite a los buscadores clasificar los documentos HTML de la red y ponerlos en una página web a modo de índice

o catálogo, que se puede mostrar por medio del navegador. Gracias a que el lenguaje HTML se ajusta a unas normas estandarizadas, todos los ordenadores pueden reproducir correctamente esos documentos. Sin embargo, el lenguaje HTML se quedaba corto pues, orientado a la presentación de datos, la información que ofrece es muy limitada, no permite describir datos y no es extensible, esto es, únicamente ofrece un pequeño número de etiquetas. El sistema evolucionó y se realizaron algunas mejoras para hacer este lenguaje algo más dinámico con la introducción de otros elementos como DHTML, Javascript, hojas de estilo e, incluso, se añadieron a la Web otros lenguajes que permitieran ofrecer una información más estructurada, como el lenguaje XML, pero hacen falta otros lenguajes que permitan una descripción más detallada del documento y de su contenido, y que faciliten la comunicación entre los ordenadores. Y también hace falta una nueva generación de buscadores más inteligentes que puedan leer y evaluar rápidamente los documentos de Internet.

Así pues, el desarrollo de la Web semántica requiere la utilización de otros lenguajes como el lenguaje estructurado XML (Extensible Markup Language) y el lenguaje RDF (Resource Description Framework) que puedan dotar a cada página, a cada archivo y a cada recursos o contenido de la red, de una lógica y un significado, y que permitan a los ordenadores conocer el significado de la información que manejan con el fin de que esta información pueda no sólo ser presentada en pantalla, sino también que pueda ser integrada y reutilizada. XML ha logrado convertirse hoy en un lenguaje estándar. Se trata de un subconjunto del complejo y sofisticado lenguaje SGML que aporta datos estructurados a la Web y que se ha convertido en la infraestructura preferida para el intercambio de datos. Además, las páginas XML pueden ubicar metadatos, esquemas XML y esquemas RDF, que aportan un mecanismo para que los programas puedan interpretar y comprender documentos con un vocabulario descriptivo.

Es necesario, pues, crear una ontología o biblioteca de vocabularios descriptivos/semánticos, definidos en formato RDF y ubicados en la Web para determinar el significado contextual de una palabra por medio de la consulta a la ontología apropiada. De esta forma, agentes inteligentes y programas autónomos podrían rastrear la Web de forma automática y localizar, exclusivamente, las páginas que se refieran a la palabra buscada con el significado y concepto precisos con el que interpretemos ese término. Por lo tanto, para potenciar el uso de ontologías en la Web, se necesitan aplicaciones específicas de búsqueda de ontologías, que indiquen a los

usuarios las ontologías existentes y sus características para utilizarlas en su sistema. En la Figura 2 podemos ver el flujo que aplica la web semántica en su estructura.

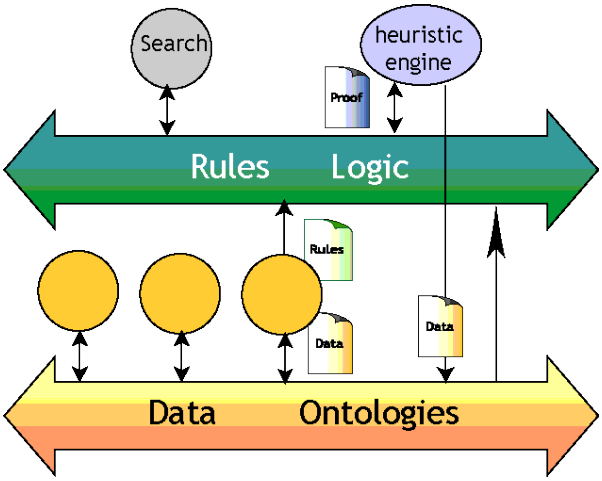

Figura 2. Bus de Web Semántica
Fuente: Recuperado de http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide14-0.html
Elaborado por: (Tim Berners-Lee, James Hendler, 2001)

Gracias a la semántica en la Web, el software es capaz de procesar su contenido, razonar con este, combinarlo y realizar deducciones lógicas para resolver problemas cotidianos automáticamente.

Supongamos que la Web tiene la capacidad de construir una base de conocimiento sobre las preferencias de los usuarios y que, a través de una combinación entre su capacidad de conocimiento y la información disponible en Internet, sea capaz de atender de forma exacta las demandas de información por parte de los usuarios en relación, por ejemplo, a reserva de hoteles, vuelos, médicos, libros, etc.

Si esto ocurriese así en la vida real, el usuario, en su intento, por ejemplo, por encontrar todos los vuelos a Quito para mañana por la mañana, obtendría unos resultados exactos sobre su búsqueda. Sin embargo la realidad es otra. Resultados inexactos que se obtendrían con el uso de cualquier buscador actual, el cual ofrecería información variada sobre Quito pero que no tiene nada que ver con lo que realmente el usuario buscaba. El paso siguiente por parte del usuario es realizar una búsqueda manual entre esas opciones que aparecen, con la consiguiente dificultad y pérdida de tiempo. Con la incorporación de semántica a la Web los resultados de la búsqueda serían exactos. Estos resultados ofrecen al usuario la información exacta que estaba buscando. La ubicación geográfica desde la que el usuario envía su pregunta es detectada de forma automática sin necesidad de especificar el punto de partida, elementos de la oración como "mañana" adquirirían significado, convirtiéndose en un

día concreto calculado en función de un "hoy". Algo semejante ocurriría con el segundo "mañana", que sería interpretado como un momento determinado del día. Todo ello a través de una Web en la que los datos pasan a ser información llena de significado. El resultado final sería la obtención de forma rápida y sencilla de todos los vuelos a Quito para mañana por la mañana.

La forma en la que se procesará esta información no sólo será en términos de entrada y salida de parámetros sino en términos de su **SEMÁNTICA**. La Web Semántica como infraestructura basada en metadatos aporta un camino para razonar en la Web, extendiendo así sus capacidades.

No se trata de una inteligencia artificial mágica que permita a las máquinas entender las palabras de los usuarios, es sólo la habilidad de una máquina para resolver problemas bien definidos, a través de operaciones bien definidas que se llevarán a cabo sobre datos existentes bien definidos.

Para obtener esa adecuada definición de los datos, la Web Semántica utiliza esencialmente RDF, SPARQL, y OWL, mecanismos que ayudan a convertir la Web en una infraestructura global en la que es posible compartir, y reutilizar datos y documentos entre diferentes tipos de usuarios. Basado en la información previamente analizada, la cual se considera esencial para llegar a entender los objetivos del tema de fin de titulación, se inicia con la investigación de uno de los términos más particulares que tienen relación con la web semántica, tal como es Linked Data.

2.2. Linked Data

Según (Berners-Lee & Tim, 2010), Linked Data trata sobre el uso de la Web para conectar datos previamente vinculados o mediante la Web permitiendo reducir las barreras para enlazar datos actualmente enlazados utilizando otros métodos.

La web semántica usa los datos enlazados para vincular los distintos datos que están distribuidos en la Web, referenciándose de la misma forma que lo hacen los enlaces de las páginas web. La web semántica no conlleva únicamente de la publicación de datos en la Web, sino que éstos se pueden vincular a otros, de forma que las personas y las máquinas puedan explorar la web de los datos, pudiendo llegar a información relacionada que se hace referencia desde otros datos iniciales.

Como lo hace notar (Consortium, 2017), Linked Data permite construir una gran base de datos interconectados y distribuidos en la Web. Los datos se vinculan y se exploran de una forma similar a la utilizada para vincular los documentos HTML.

Los Datos Enlazados, como parte de la Web Semántica, se basa en la aplicación de ciertos principios básicos y necesarios, que impulsan el crecimiento de la Web, tanto a nivel de los documentos HTML (vista clásica de la Web), como a nivel de los datos expresados en RDF (vista de la Web Semántica). Se debe respetar los siguientes pasos para conseguir tener los datos interconectados. Gracias a esta interconexión, se permite reutilizar la información de cualquier manera esperada o inesperada, lo que ofrece un valor añadido a la Web.

En el año 2006, Tim Berners-Lee escribió una nota de diseño (Berners-Lee, 2006), proponiendo soluciones a los problemas que impedían el enlace de los datos, mediante la aplicación de los principios de Linked Data que son los siguientes:

- Usar URIs para identificar las cosas
- Usar URIs HTTP
- Ofrecer información sobre los recursos usando RDF
- Incluir enlaces a otros URIs

Lo anterior implica usar el modelo de datos RDF para publicar datos estructurados en la web y usar enlaces RDF para enlazar los datos de diferentes fuentes. Estos principios permitieron impulsar el desarrollo de varios proyectos de Linked Data (Heath, Tom; Bizer, 2011).

Por otra parte, Tim Berners-Lee presentó en TED 2009 (Technology Entertainment and Design) una conferencia, en la que redefinió los principios de Linked Open Data presentándolos como tres reglas que se resumen en: a) asignar a todas las cosas conceptuales nombres que comienzan con http; b) obtener información importante de retorno a partir de los nombres; y c) la información obtenida debe contener relaciones. Se hace énfasis en esta conferencia en publicar los datos lo más pronto posible (Berners-Lee, 2009).

A partir de 2009 se ha producido un avance más rápido en el desarrollo de tecnologías de Linked Data. Se han liberado y usado algunas especificaciones de la W3C tales como: SPARQL, GRDDL, RDFa, VoID, se ha formado la comunidad del proyecto Linking Open Data y cada vez se observa un creciente uso en gobierno electrónico con numerosos catálogos de datos publicados y aplicaciones con funcionalidades específicas orientadas a dominios que combinan datos de varias fuentes de Linked Data (Berners-Lee, 2009), (Sheridan, 2010), (Cyrille, Axel; Ngomo, 2010).

2.3. Beneficios de Linked Data

Para realizar un correcto análisis del tema de fin de titulación, se ha investigado los beneficios que proporciona Linked Data, tal y como (Nacional, 2014) lo lista a continuación.

2.3.1. **Igualdad**

Acceso libre e igualitario a los datos, lo que permite la interpretación y reinterpretación de la información generada por las instituciones y organismos públicos, desde el espacio ciudadano.

2.3.2. **Empoderamiento de la ciudadanía**

Da poder a la ciudadanía en la solución de los problemas que los afectan, impulsando los procesos de identidad, conocimiento, comprensión y colaboración ciudadana.

2.3.3. **Transparencia**

La oferta de información en “crudo”, clara y actualizada en formatos abiertos, estándar y reutilizables, promueve una mayor transparencia de la gestión en organismos e instituciones públicas.

2.3.4. **Colaboración: Innovación, soluciones y co-creación**

La reutilización de los datos sin restricciones de copyright permite iniciativas de participación creativa, donde los ciudadanos comparten lo que saben y generan soluciones innovadores en las áreas donde tienen conocimientos, creando nuevos servicios o mejorando los que ya existen. Abre la co-creación para entender necesidades de usuarios y el uso de la opinión pública para ordenar ciertos datos.

2.3.5. **Participación**

Posibilita la intervención efectiva de la ciudadanía, proceso en el cual las personas se involucran en la toma de decisiones que los afectan o en iniciativas que mejoran su entorno de convivencia.

2.4. Tecnologías relacionadas con Linked Data

Según lo describe (Data, 2012), varias tecnologías están ya disponibles para publicación y consumo de Linked Data que pueden ser usadas también para integración y mejor gestión de datos corporativos. En este apartado se clasifican atendiendo a su finalidad: técnicas y herramientas que sirvan para la publicación, o técnicas y herramientas destinadas a facilitar el consumo de datos enlazados.

2.4.1. **Identificador Uniforme de Recursos (URI)**

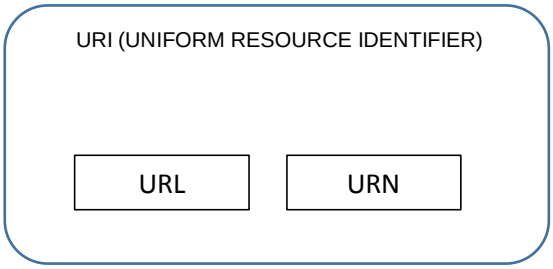

Figura 3. Estructura de una URI
Fuente y elaboración por: Autor

URI: Identificador Uniforme de Recursos (Identifica)

URL: Localizador Uniforme de Recursos (Localiza)

URN: Nombre Uniforme de Recursos

Un URI definido según el estándar RFC (3986), tal como (Lee, Roy, & Larry, 2005) nos indica que permite identificar un recurso ya sea por un localizador (ubicación), un nombre o ambos. Un recurso hace referencia a: documentos, imágenes, archivos, páginas web entre otros.

Se puede considerar las siguientes diferencias:

- Toda URL es un URI.
- Toda URN es un URI.
- Un URL es un URI, pero un URI no es siempre una URL.
- Las URL, siempre contienen un mecanismo de acceso: http, https, ftp.

Inmersos en el contexto de Linked Data, como lo enfatiza en el 2° principio, se hace el uso de URIs HTTP, con el fin de garantizar que un recurso se pueda buscar y acceder mediante la web.

Una URI HTTP permite:

- Referenciar de manera única a un recurso.
- Obtener información de un recurso representada en estándares RDF.
- Identificar entidades del mundo real y sus relaciones.
- (Heatj & Bizer, 2011) argumenta que proporcionan una forma sencilla de crear nombres únicos a nivel mundial de una manera descentralizada.

2.4.2. **Marco de Descripción de Recursos (RDF)**

Tal como lo plantea (Graham & Carroll, 2004) en la W3C (World Wide Web Consortium) RDF es un modelo de datos para representar metadatos, permite describir la semántica de información de una manera accesible para las máquinas.

RDF se basa en los estándares de URIs y Unicode además de que se puede presentar en XML.

Las declaraciones o modelo de datos RDF son elaborados mediante el uso de tripletas, que son los recursos: sujeto y objeto, vinculados por el predicado.

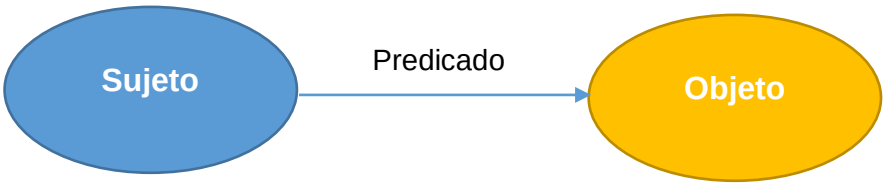

Figura 4. Estructura Tripleta RDF
Fuente y elaboración por: Autor

- **Sujeto:** Recurso, objeto o cosa de la que se hace una afirmación. Ejemplo: Universidad, carrera, docentes, etc.
- **Predicado:** Describen las relaciones entre los recursos, corresponden a las etiquetas de los arcos del grafo, por ejemplo “es parte de”.
- **Objeto:** Valor del predicado para el sujeto:

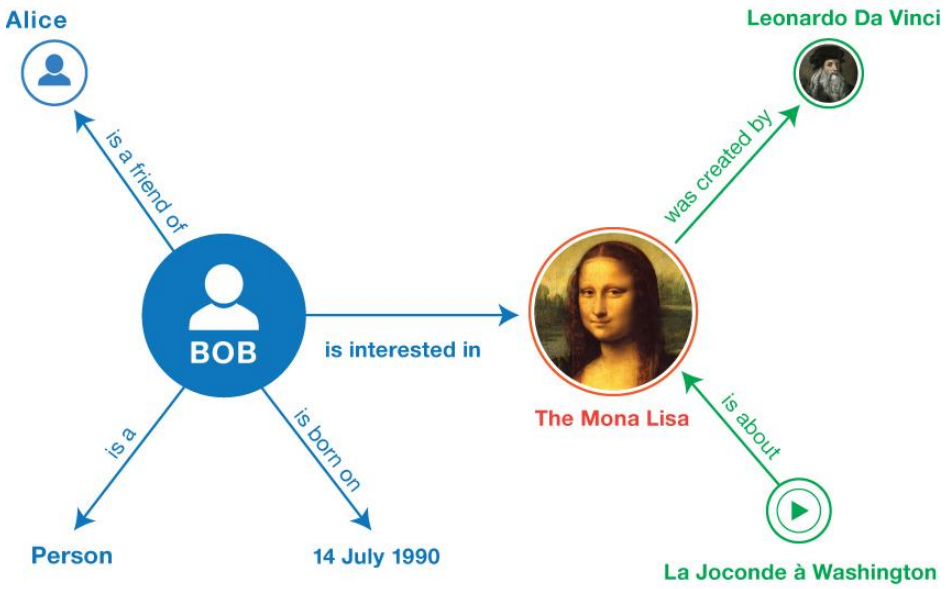

Figura 5. Grafo informal de tripletas
Fuente: Recuperado de https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/
Elaborado por: (Manola Frank, Miller Eric, 2014)

Veamos un ejemplo concreto, extraído de la especificación (Manola Frank, Miller Eric, 2014) donde se muestran una serie de declaraciones o sentencias: hay una persona identificada por http://www.w3.org/People/EM/contact#me, cuyo nombre es Eric Miller,

cuya dirección de correo electrónico es em@w3.org, y cuyo título es "Dr." que podría representarse como el grafo RDF de la siguiente figura:

Figura 6. Un grafo RDF describiendo a Eric Miller
Fuente: Recuperado de https://www.w3.org/TR/2004/REC-rdf-primer-20040210/
Elaborado por: (Frank Manola, 2004)

Esta figura ilustra que RDF usa *URIs* para identificar:

- Individuos, por ejemplo, **Eric Miller**, identificado por **http://www.w3.org/People/EM/contact#me**
- Clases de cosas, por ejemplo, **Person**, identificado por **http://www.w3.org/2000/10/swap/pim/contact#Person**
- Propiedades de estas cosas, por ejemplo, **mailbox**, identificado por **http://www.w3.org/2000/10/swap/pim/contact#mailbox**
- Valores de estas propiedades, por ejemplo, **mailto:em@w3.org** como el valor de la propiedad **mailbox** (RDF también usa cadenas de caracteres tales como "Eric Miller", y valores de otros tipos de datos como enteros y datos, o valores de propiedades)

RDF también provee una sintaxis basada en XML (**llamada RDF/XML**) para guardar e intercambiar estos grafos. Este ejemplo es una pequeña muestra de RDF en RDF/XML correspondiente al grafo de la ilustración anterior.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

```
   xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

Este RDF/XML también contiene URIs, y propiedades como **mailbox** y **fullName** (en forma abreviada), y sus respectivos valores **em@w3.org**, y **Eric Miller.**

Como HTML, este RDF/XML es procesable por máquina y, usando URIs se pueden enlazar las piezas de información a lo largo de la Web. Sin embargo, al contrario que el hipertexto convencional, los URIs de RDF pueden hacer referencia a cualquier cosa identificable, incluyendo cosas que pueden no ser directamente recuperables en la Web (tales como la persona **Eric Miller**). El resultado es que, en adición para describir tales cosas como páginas *web*, RDF puede también describir coches, negocios, gente, noticias, eventos etc. E incluso, las propiedades RDF que tienen los URIs, para precisamente identificar las relaciones que existen entre los términos enlazados. En RDF tanto los sujetos, como las propiedades y los objetos, son recursos.

2.4.3. **RDF – SCHEMA**

Tal como lo plantea (Brickley, Guha, & McBride, 2014) en la W3C (World Wide Web Consortium) RDFS permite modelar metadatos con una representación explícita de su semántica y permite especificar restricciones de tipos de datos para los sujetos y objetos de las tripletas de RDF. Un esquema define el significado, características y relaciones de un conjunto de propiedades. RDFS se construye por sobre RDF y permite:

- El modelado de objetos con una semántica explícita.
- Especificar restricciones de tipos de datos.
- Define el significado, características y relaciones entre las propiedades.
- Define Dominio y rango de propiedades: en qué clases está cada propiedad y qué valores puede tomar.
- Jerarquía de clases.

En la Tabla 4 y Tabla 5 se presentan algunas de las clases y propiedades de RDF-SCHEMA.

Tabla 1. Clases RDF y RDF SCHEMA

Clases	Descripción
rdf:Property	Clase de las propiedades RDF.
rdf:XMLLiteral	Clase de valores literales XML
rdfs:Class	Clase de las clases.
rdfs:Resource	Clase de Recursos.
rdfs:Literal	Clase de valores literales, por ejemplo strings e integer

Elaborado por: El Autor

Tabla 2.Propiedades RDF y RDF SCHEMA

Clases	Descripción
rdf:type	Se utiliza para indicar que un recurso es una instancia de una clase.
rdfs: domain	Se utiliza para indicar que cualquier recurso que tiene una propiedad dada es una instancia de una o más clases.
rdfs: range	Se utiliza para indicar que los valores de una propiedad son instancias de una o más clases.
rdfs:subClassOf	Se utiliza para indicar que todas las instancias de una clase son instancias de otro
rdfs: subPropertyOf	Indica que el sujeto es una subpropiedad de una propiedad
rdfs:comment	Se utiliza para proporcionar una descripción legible de un recurso
rdfs:label	Se puede utilizar para proporcionar una versión legible del nombre de un recurso.
rdfs:seeAlso	Se utiliza para indicar un recurso que puede proporcionar información adicional sobre el recurso sujeto

Fuente y elaboración por: Autor

2.4.4. **Protocolo simple y lenguaje de consulta RDF (SPARQL)**

Así como (Seaborne, 2008) determina que SPARQL, es el lenguaje recomendado por la W3C, para consulta de datos RDF, la expresividad de SPARQL permite realizar consultas a múltiples fuentes de datos o grafos, que deben estar en formato RDF, su sintaxis es similar a SQL aunque orientado a tripletas y grafos RDF. Los resultados de las consultas SPARQL pueden ser conjuntos de tripletas RDF, grafos RDF, URIs de entidades o simplemente valores.

SPARQL permite acceder a una gran cantidad de información disponible en la Web, ya sea en DBpedia, el New York Times, o el gobierno de El Reino Unido. Es importante recordar que en el caso de DBpedia, la información es tomada de Wikipedia y extraída lo mejor posible; sin embargo, siempre es posible que exista algún error en los datos, algo que es más difícil que ocurra en los datos de un periódico o un gobierno.

2.4.4.1. ***Sintaxis SPARQL***

SPARQL es bastante similar a SQL. Por ejemplo:

```
SELECT ?persona ?nombre WHERE{ ?persona . ?persona ?nombre . }
```

Como se puede ver, las variables comienzan con "?" y después del WHERE comienzan una lista de patrones de grafos (una lista de sujeto, predicado y objeto que finaliza con un punto) que los posibles resultados deben cumplir. Así, la consulta anterior busca cualquier entidad que sea de tipo "Persona" y que además posea un nombre.

Lenguaje de Consulta para RDF utilizado para realizar consultas de diversas fuentes de datos.

PREFIX: Equivalente al namespace en XML, en lugar de repetir las uris varias veces se utilizan un prefix.

SELECT: Clausula requerida en cada consulta, es similar al uso especificado en SQL, define cada una de las variables a retornar, van precedidos por el símbolo (?).

WHERE: Patrón de consulta, con una o más tripletas encerradas entre llaves {}.

Tabla 3. Sintaxis y clausulas SPARQL

Sintaxis y Cláusulas	Descripción
DESCRIBE	Devuelve un grafo RDF, con la descripción del recurso encontrado. Opcional

FILTER	Permite establecer restricciones adicionales al patrón de búsqueda.
ORDER BY	Misma funcionalidad que la definida en SQL.
LIMIT n	Restringe el número de resultados devueltos. Opcional.
OFFSET m	Misma funcionalidad que la definida en SQL. Opcional.

Fuente y elaboración por: Autor

2.4.5. **Publicación**

La publicación en la Web semántica se puede realizar usando una variedad de herramientas y patrones de publicación siguiendo los principios de Linked Data tal como lo indica (Data, 2012), lo cual permite la interoperabilidad de los datos y reutilización en la Web. Sin embargo esto no implica el abandono de los sistemas existentes, sino más bien la adición de capas extras para conectar los datos en la Web. Estas capas pueden estar formadas `por extractores de entidades para armar archivos RDF, almacenes de datos RDF, interfaces a Linked Data como Pubby, vistas RDF sobre Bases de Datos relacionales, APIs, uso de Sistemas de Administración de Contenidos (CMS) con RDFa o con salidas RDF, entre otros (Hendler, 2011). Adicionalmente la publicación de un conjunto de datos implica: escoger el conjunto de datos, aplicar una licencia abierta, publicar los datos, difundir la publicación, y evaluar su utilidad.

2.4.6. **Herramientas para publicación**

Para publicar Linked Data se han desarrollado varias herramientas entre las cuales tenemos:

- D2R server es una herramienta para publicar bases de datos relacionales como Linked Data transformando el contenido de una base de datos relacional a RDF (Cyganiak, Richard; Bizer, 2011).
- Triplyfy es un plugin para aplicaciones web que revela las estructuras semánticas codificadas en bases de datos relacionales haciendo el contenido de bases de datos disponibles como RDF, JSON o Linked Data (Jaenicke, 2009).
- La Plataforma Talis proporciona hospedaje para contenido y datos RDF tanto para publicar datos como para desarrollo. Además ofrece interfaces de consulta y manipulación de datos con Kasabi, una plataforma estructurada de SaaS (Software as a Service) (Heatj & Bizer, 2011).

- Pubby es una interfaz para Linked Data que permite interactuar con puntos SPARQL (SPARQL endpoints) mediante consultas SPARQL (Cyganiak, Richard; Bizer, 2011).
- Paget es un marco de trabajo (framework) para construir aplicaciones con Linked Data (Hendler, 2011).
- Linked Media Framework: Es un servidor con actualizaciones y búsquedas semánticas (Hendler, 2011).
- Open Link Virtuoso Universal Server: Es un sistema de administración de bases de datos orientado a objetos que permite almacenar datos RDF y presentarlos con la funcionalidad de un punto SPARQL. Los datos en RDF pueden ser almacenados directamente en Virtuoso o creados a partir de bases de datos relacionales no RDF (Erling, 2010).
- Virtuoso Sponger: Permite transformar datos no RDF en RDF. Acepta como entradas páginas web HTML, páginas web HTML con micro formatos y servicios web como aquellos de Google, Del.icio.us, Flickr etc. Y crea RDF como salidas (Hendler, 2011).
- Sesame: Es un sistema formado por un repositorio, una máquina de búsqueda y un módulo de administración para añadir y borrar datos RDF e información de esquema. Sesame está siendo usado en algunas grandes compañías y agencias de gobierno para integración de datos (Broekstra, Jeen; Kampman, 2004).
- Jena TDB: Es un repositorio de datos RDF que permite añadir datos usando SPARQL/Update y permite consultas de datos usando el lenguaje SPARQL (Hendler, 2011).
- 3Store: Es un repositorio de ternas RDF basado en MySQL. El software no presenta interfaces directamente al usuario pero puede ser consultado por varios servicios incluyendo el navegador Direct RDF (Aduna, 2009).

2.4.7. **Formatos de Publicación**

Para la publicación de datos enlazados en RDF existen varios tipos de formatos. Por consiguiente se hace una descripción breve de los más comunes, cada una con un ejemplo enmarcado en el contexto de la presente investigación.

2.4.7.1. ***RDF/XML***

Tal como lo plantea (Dave Beckett, 2014) se considera un formato estándar y normativo difícil para leer y escribir. En la siguiente imagen se representa un recurso mediante la notación RDF/XML.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.w3.org/2001/sw/RDFCore/ntriples/">
    <dc:creator>Art Barstow</dc:creator>
    <dc:creator>Dave Beckett</dc:creator>
    <dc:publisher rdf:resource="http://www.w3.org/"/>
  </rdf:Description>
</rdf:RDF>
```

Figura 7. Fuente: Sintaxis RDF/XML
Fuente: Recuperado de https://www.w3.org/2001/sw/RDFCore/ntriples/
Elaborado por: (Dave Beckett, 2014)

2.4.7.2. ***Notación 3 (N3)***

N3 no es un estándar de la W3C, a lo cual (Berners-Lee & Connolly, 2011) cataloga que es ampliamente usado, en su mayoría en los foros de discusión sobre la Web Semántica; además de ser la notación más importante de RDF para entenderlo, ya que captura de manera clara el grafo abstracto.

2.4.7.3. ***N-Triples***

En su último artículo sobre N-Triples (Beckett & Barstow, 2001) plantea que se utiliza para publicar grandes conjuntos de datos RDF como DBpedia. En la siguiente imagen se representa un recurso mediante la notación N-Triples.

```
<http://www.w3.org/2001/sw/RDFCore/ntriples/> <http://purl.org/dc/elements/1.1/creator> "Dave Beckett" .
<http://www.w3.org/2001/sw/RDFCore/ntriples/> <http://purl.org/dc/elements/1.1/creator> "Art Barstow" .
<http://www.w3.org/2001/sw/RDFCore/ntriples/> <http://purl.org/dc/elements/1.1/publisher> <http://www.w3.org/> .
```

Figura 8. Sintaxis N-Triples
Fuente: Recuperado de https://www.w3.org/2001/sw/RDFCore/ntriples/
Elaborado por: (Beckett & Barstow, 2001)

2.5. Trabajos relacionados con Linked Data

2.5.1. **Ambar Linked Data Portal**

Tal como da a conocer (Foundation, 2011), Ambar es un Portal de Datos creado con el propósito de desarrollar una nueva infraestructura de datos abierta en el contexto de Gestión Inteligente de Territorios. Su objetivo es ser una puerta de entrada a los datos publicados por diferentes actores: gobiernos, organizaciones, universidades y centros de investigación. El portal se puso en marcha en el contexto de la iniciativa Smart Land.

2.5.2. **IPHealth**

Tal como lo expresa (Cientifica, 2013) la medicina personalizada (PM, Personalized Medicine) busca la identificación de terapias personalizadas que hagan seguro y efectivo el tratamiento individualizado de pacientes específicos. Una de las grandes

dificultades para llevar a cabo esta práctica clínica de forma efectiva es que en la actualidad no existen sistemas flexibles de información capaces de proporcionar conocimiento preciso, actualizado e interrelacionado basado en el acceso estratificado a múltiples orígenes de datos de tipo heterogéneo. Toda esta información, generada en estudios experimentales, ensayos clínicos y en la práctica clínica diaria, así como recientemente a través de sensores biomédicos y grandes conjuntos de datos de libre acceso y entrelazado (Open y Linked Data) debería convertirse en una fuente extraordinaria de conocimiento para el avance de la PM. Sin embargo, la PM se enfrenta en la actualidad a grandes retos. Es necesario integrar información heterogénea dispersa en múltiples orígenes, de diferentes géneros, dominios, estructuras y escalas, donde además juega un papel muy importante el componente textual. Para afrontar estos retos, en este proyecto se propone la aplicación de forma coordinada de técnicas de integración de información para conseguir abarcar fuentes de tipo heterogéneo y de minería de textos y datos para facilitar la extracción del conocimiento asociado.

A continuación se exponen algunos proyectos que son considerados como una iniciativa dentro del campo de la educación, así mismo se señala algunos de los eventos más importantes, en donde podemos encontrar una gran cantidad de información respecto a Web Semántica y Linked Data.

2.5.3. **Linked Open Data University Munster (LODUM)**

Basado en lo que (Muster's, 2013) manifiesta, el Proyecto de la Universidad de Munster (Alemania), fue creado con el objetivo de disponer de datos de investigación como datos vinculados abiertos para mejorar la transparencia, visibilidad y accesibilidad a los datos.

2.5.4. **Educational Curriculum for the usage of Linked Data – EUCLID**

En cuanto a la descripción de (Devsaran, 2016), EUCLID es un Proyecto europeo para facilitar la formación de profesionales de datos, que tienen como objetivo utilizar los Datos Enlazados. Este ofrece un currículo implementado como una combinación de materiales y actividades (series eBook, seminarios, capacitación cara a cara), validados por la comunidad de usuarios a través de una retroalimentación continua de aprendizaje.

2.5.5. **Open University Data**

Según advierte (University, 2017), es una plataforma para exponer los datos disponibles en varios repositorios institucionales de la Universidad y ponerla a disposición abierta para ser reutilizado. Los conjuntos de Datos se refieren a las

publicaciones, cursos, materias producidas por Open University, estos datos están disponibles en RDF y SPARQL

2.5.6. **Linked University**

Tal y como (Mathieu d'Aquin, Kessler, & Kauppinen, 2014) especifica que es una alianza de universidades europeas que participan en la exposición de sus datos públicos como datos vinculados. La idea es que los datos de diferentes instituciones y organizaciones pueden contribuir a un espacio común de datos en la Web: la Web de Datos apoya al desarrollo de vocabularios, así como a la creación y recomendación de herramientas reutilizables y además proporciona guías y experiencias para la publicación de datos.

2.6. Linked Open Data

Para entender lo que es Datos abiertos enlazados hay que comenzar conociendo el concepto Open Data (datos abiertos).

2.6.1. **Open Data**

Tal y como (Nacional, 2014) expresa, Open data es considerado un movimiento digital al que están adhiriendo paulatinamente gobiernos e instituciones de todo el mundo para poner los datos que administran a libre disposición de las personas e instituciones sin restricciones de copyright, patentes u otros; en formatos que permitan su reutilización para cualquier fin, por ejemplo, el desarrollo de nuevos análisis o de aplicaciones online.

De la mano del término Open Data está el de Linked Data, pero es conveniente aclarar que ambos términos no son sinónimos: el concepto de Linked Data hace referencia a que los datos están enlazados mediante tecnologías de Web Semántica, en particular RDF, que es el estándar que utiliza para describir los recursos web, especificar metadatos y representar información. El RDF tiene como modelo de datos el grafo.

El término Linked Data es considerado un conjunto de buenas prácticas para la publicación de datos, no implica que los datos deban ser gratuitos, de utilización libre o abierta como postula Open Data. Por ejemplo, una empresa puede tener dos áreas (stock y ventas) y ambas publicar sus datos como Linked Data, de manera que sus sistemas inter-operen basados en el estándar RDF, todo dentro de su red privada.

Sólo cuando ambos conceptos se unen surge el término Linked Open Data o Datos abiertos enlazados. Linked Open Data indica que son datos abiertos en RDF (que es el estándar W3C para Web Semántica). Esto significa que el usuario puede enlazar

datos provenientes de diversas fuentes, instituciones u organizaciones, explorar y combinar estos datos de manera libre y sin restricciones de copyright para nuevos desarrollos web.

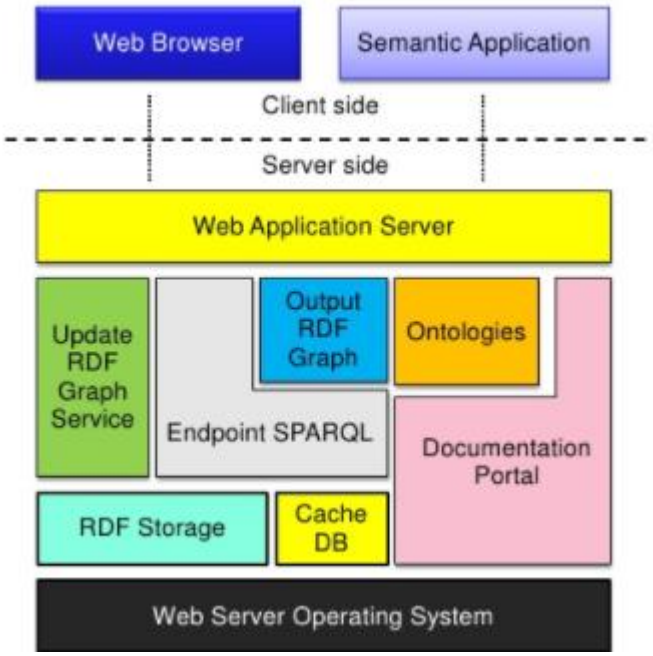

Figura 9. Arquitectura de Linked Open Data
Fuente: Recuperado de https://www.slideshare.net/jelabra/towards-an-architecture-and-adoption-process-for-linked-data-technologies-in-open-government-contexts-a-case-study-for-the-librery-of-congress-of-chile
Elaborado por: (Gayo, 2011)

2.7. Linked Stream Data (LSD)

Desde el punto de vista de (Danh et al., 2012), LSD es considerado un modelo de datos RDF extendido para representar datos de flujo generados a partir de sensores y aplicaciones de redes sociales.

Así como (Sequeda & Corcho, 2009) expresa, el costo de despliegue de redes de sensores ha estado cayendo en los últimos años, mientras que su capacidad ha ido aumentado constantemente. Como resultado, se están desplegando más redes de sensores en muchos entornos diferentes (carreteras, bosques, tierras agrícolas, personas, hogares entre otros.) y la información procedente de estas redes de sensores se utiliza con mayor frecuencia para mejorar la evaluación de la situación y la toma de decisiones.

La cantidad de información generada por las redes de sensores desplegadas es extremadamente grande. Por ejemplo, los sensores de temperatura pueden emitir sus lecturas cada 30 min, mientras que los sensores de frecuencia cardíaca pueden enviar

sus datos a un repositorio cada minuto. Disponer de estos datos no sólo internamente a las aplicaciones legadas, sino también disponibles en la web proporcionará una nueva fuente de conocimiento para los científicos, tomadores de decisiones y otros tipos de usuarios. Podemos entonces hablar de la red mundial de sensores.

Sin embargo, la disponibilidad de estos datos en la web plantea nuevos desafíos relacionados con la forma en que estos datos pueden ser descubiertos, identificados y explotados en una serie de aplicaciones. En otras palabras, debe haber una manera de identificarlo, describirlo consistentemente y acceder fácilmente.

Creemos que existe una buena oportunidad para aplicar a los datos de los sensores los mismos principios que se han utilizado para la publicación de otros tipos de datos (más estáticos) en la Web (Semántica), en el contexto de la iniciativa Linked Data.

En la aplicación de estos principios, hay varias características que claramente diferencian el tipo de datos que normalmente se publican en el mundo Linked Data de datos que se originan en redes de sensores. En primer lugar, los datos basados en sensores están normalmente relacionados con mediciones físicas y observaciones, por lo tanto predominantemente numéricos. En segundo lugar, las consideraciones de tiempo y espacio tienen que ser tratadas: los sensores y las redes de sensores están situados en posiciones geográficas específicas y estas posiciones suelen ser importantes para la toma de decisiones y para el procesamiento ulterior y las mediciones proporcionadas son comúnmente marcadas con marcas de tiempo (lo que permite fragmentos de datos basados en sensores para ser identificados por ventanas de tiempo). En tercer lugar, la exactitud de los datos y la incertidumbre es otro factor a tener en cuenta. Aunque estas son características de la mayoría de los tipos de datos basados en sensores, esto no es exclusivo de este tipo de datos y también podría ser el caso de otros tipos de datos.

2.8. Propósito de Linked Stream Data

En la sección anterior, hemos presentado varios requisitos para identificar sensores y datos de flujo procedentes de sensores en la web. Por nuestra motivación, creemos que Stream Data puede convertirse en parte de la Web de datos mediante la definición de un mecanismo basado en URI, siguiendo los principios de datos enlazados, para identificar sensores y datos de flujo, opcionalmente dado su contexto espacial y temporal, que se puede proporcionar en muchas formas diferentes. En esta sección, presentamos una propuesta de URIs human-friendly para identificar sensores y datos de flujo.

2.9. Casos de uso orientados a Linked Stream Data

En esta sección presentamos una serie de casos de uso en los que se utiliza información procedente de sensores y que puede beneficiarse de la disponibilidad de esta información como datos enlazados.

2.9.1. **Referencia lineal de carretera**

Es un punto de referencia bien establecido para los sistemas de gestión de flujo de datos. Este punto de referencia específica un sistema de peaje variable determinando los factores cambiantes de la congestión del automóvil en una carretera. Cada coche en la carretera está equipado con un sensor que emite la ubicación exacta del vehículo y la velocidad cada 30 segundos. Los datos emitidos por los sensores se envían como flujos a un sistema central donde se generan estadísticas sobre las condiciones de tráfico en las carreteras. Este sistema de peaje está diseñado para desalentar a los conductores a utilizar carreteras ya congestionadas porque tienen un peaje mayor. En consecuencia, alentaría a los conductores a utilizar carreteras menos congestionadas porque habrían disminuido los peajes. Aunque este caso de uso se creó para probar y comparar diferentes características de sistemas de gestión de flujo de datos existentes, el dominio en el que se aplica es uno que puede beneficiarse claramente de su disponibilidad como datos enlazados.

2.9.2. **Sensores del corazón**

Los pacientes con problemas cardíacos pueden tener sensores que controlan su ritmo cardíaco y su ubicación actual. Los datos emitidos por el sensor cardíaco pueden enviarse como datos de flujo al hospital del paciente donde se monitoriza. El hospital puede detectar si el paciente sufre de cualquier anormalidad del corazón en tiempo real. Además, si un paciente está teniendo un ataque al corazón, el hospital puede enviar inmediatamente una ambulancia a la ubicación exacta del paciente. El uso de protocolos Web comunes para publicar y acceder a estos datos, a la vez que preserva la seguridad y la privacidad, puede facilitar el desarrollo de este tipo de aplicaciones de gestión de emergencias.

2.9.3. **Sensores ambientales**

Los investigadores ambientales necesitan rastrear un gran número de aspectos de regiones específicas. Por ejemplo, una aplicación específica puede monitorizar la temperatura y la humedad de una región. Normalmente se utilizan sensores estáticos en este tipo de aplicaciones, ya que están supervisando áreas fijas, por lo tanto el espacio no es un problema en estos tipos de sensores. Sin embargo, la exactitud de las mediciones realizadas puede ser relevante.

2.10. Extracción de información de casos de uso orientados a LSD

En esta sección presentamos algunos de los requisitos que se pueden extraer de los casos de uso anteriores y que consideramos que necesitan ser satisfechos por nuestra propuesta de datos de flujo enlazado.

2.10.1. **Identificación**

Los recursos en la Web se identifican comúnmente por medio de URIs. Como se menciona en los principios de Datos Enlazados, los URIs actúan como nombres únicos (identificadores) para dichos recursos en la web, pero también pueden extenderse a objetos en el mundo real. Como consecuencia, podemos proponer el uso de URIs para identificar sensores que se despliegan en el mundo real. Además, los URI también se pueden utilizar para identificar datos que son emitidos por sensores.

Los principios de Datos Vinculados también estipulan que estos deben ser HTTP URIs y que una vez de-referenciados, se debe proporcionar información útil. Por lo tanto, al recuperar los datos una vez que se des-refiere un URI, el URI actúa como una interfaz de consulta o un servicio RESTful.

2.10.2. **Consulta**

Los flujos de datos proporcionados por un sensor o grupo de sensores específicos pueden ser identificados por un momento específico en el tiempo o por una ventana de tiempo. Por ejemplo, considere un sensor que emite la frecuencia cardíaca de una persona. Uno podría identificar la frecuencia cardíaca exacta de una persona en un momento específico. Además, se podría también identificar la serie de frecuencias cardiacas emitidas por ese sensor en una ventana de tiempo específica.

2.10.3. **Dimensión Espacial**

Los flujos de datos también se pueden identificar teniendo en cuenta su contexto espacial. Por ejemplo, considere un sensor móvil que emite la frecuencia cardíaca de una persona. Uno podría identificar la frecuencia cardíaca exacta de una persona en un lugar específico. En el caso de la referencia de carretera lineal, podemos estar interesados solamente en datos procedentes de vehículos en un segmento específico. Al igual que el caso de dimensión de tiempo, el espacio de identificación en los sensores se puede hacer por una ubicación específica (o coordenada en este caso), o por un área de delimitación (similar a las ventanas de tiempo, pero para el espacio). Un área delimitada, dado un punto central, puede ser una radio, un cuadrado o un polígono.

- Los datos de flujo deben ser identificables en un lugar específico.
- Los datos de flujo deben ser identificables en un área delimitada.

- Las áreas delimitadoras pueden ser definidas por una radio, un cuadrado o un polígono.

2.10.4. **Combinación de tiempo y espacio**

Las nociones de tiempo y espacio son particularmente interesantes en el caso de sensores móviles. En el caso del sensor de ritmo cardiaco, se podría identificar la frecuencia cardíaca exacta de una persona en un momento y lugar determinados. En este caso, consideramos que los requisitos de tiempo y espacio deben ser satisfechos al identificar los datos de flujo de los sensores móviles.

- Los datos de flujo deben ser identificables en un momento específico y en una ubicación específica.
- Los datos de flujo deben ser identificables en una ventana de tiempo y en una ubicación específica.
- Los datos de flujo deben ser identificables en un momento específico en el tiempo y en un área delimitada.
- Los datos de flujo deben ser identificables en una ventana de tiempo y en un área delimitada.

2.11. Vocabularios Abiertos Vinculados (LOV)

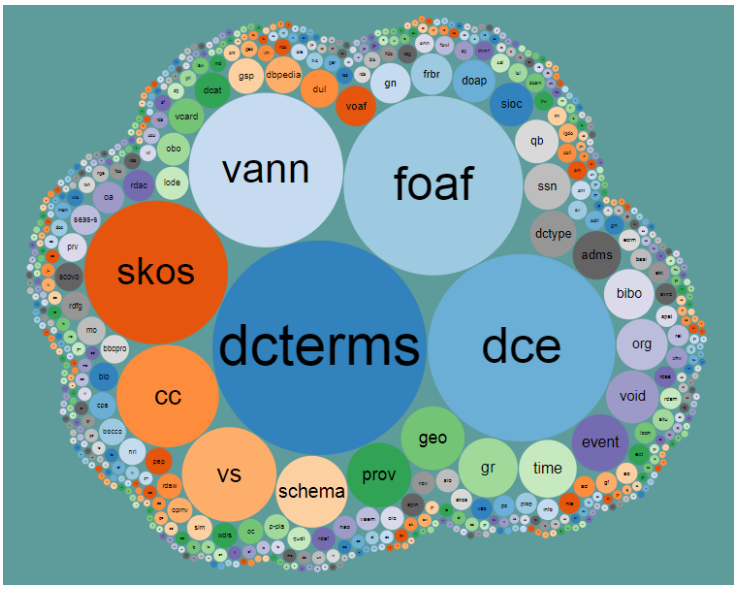

Figura 10. Vocabularios en LOV
Fuente: Recuperado de http://lov.okfn.org/dataset/lov/
Elaborado por: (Consortium, 2011)

Un vocabulario según (Foundation, Knowledge, 2014) reúne las definiciones de un conjunto de clases y propiedades (en conjunto llamadas simplemente términos del vocabulario), útil para describir tipos específicos de cosas o cosas en un dominio o

industria determinados, o cosas en general, pero para un uso específico. Los términos de los vocabularios también proporcionan los enlaces en los datos enlazados, en el caso anterior entre una Persona y una Ciudad. Las definiciones de los términos proporcionados por los vocabularios traen semántica clara a descripciones y enlaces, gracias al lenguaje formal que usan (algún dialecto de RDF como RDFS u OWL). En resumen, los vocabularios proporcionan la cola semántica permitiendo que Datos se conviertan en datos significativos.

Actualmente se puede encontrar una enorme cantidad de vocabularios que se han publicado abiertamente, algunos de estos vocabularios creados para su aplicación en un campo específico por ejemplo: medicina, gobierno, organizaciones, bibliotecas, educación entre otros. A continuación se específica los vocabularios aplicables dentro del contexto de la actual investigación:

2.11.1. **Amigo de un amigo (FOAF)**

Tal y como (Brickley & Libby Miller, 2014) describe a FOAF como proyecto que proporciona los términos que permiten enlazar personas e información entre ellas. El vocabulario permite describir personas y cosas que hacen y crean, también describe la información personal que pueda ser procesada, compartida y reutilizada.

En la Tabla 6 se representa un ejemplo utilizando algunos de los términos del vocabulario FOAF.

```
<foaf:Person rdf:about="#danbri" xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <foaf:name>Dan Brickley</foaf:name>
  <foaf:homepage rdf:resource="http://danbri.org/" />
  <foaf:openid rdf:resource="http://danbri.org/" />
  <foaf:img rdf:resource="/images/me.jpg" />
</foaf:Person>
```

Figura 11. Uso de términos del vocabulario FOAF
Fuente: Recuperado de http://xmlns.com/foaf/spec/
Elaborado por: (Brickley & Libby Miller, 2014)

2.11.2. **SCHEMA**

Tal como (Group, 2015) ha considerado, proporciona una colección de vocabularios compartidos, los mismos proporcionan las clases y propiedades para: eventos, salud, medicina, organización, personas, lugares, y productos.

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "LocalBusiness",
  "address": {
    "@type": "PostalAddress",
    "addressLocality": "Mexico Beach",
    "addressRegion": "FL",
    "streetAddress": "3102 Highway 98"
  },
  "description": "A superb collection of fine gifts and clothing to accent your stay in Mexico Beach.",
  "name": "Beachwalk Beachwear & Giftware",
  "telephone": "850-648-4200"
}
</script>
```

Figura 12. Uso de términos del vocabulario SCHEMA en JSON-LD
Fuente: Recuperado de https://schema.org/Place
Elaborado por: (Group, 2015)

2.11.3. **Tarjetas personales electrónicas (VCARD)**

Como plantea (Iannella & James McKinney, 2014), VCARD es un vocabulario que proporciona un formato RDF estándar para el intercambio de información personal, correspondiente a información de contacto de personas y organizaciones.

```
<vcard:Individual rdf:about="http://example.com/me/corky">
  <vcard:fn>Corky Crystal</vcard:fn>
  <vcard:nickname>Corks</vcard:nickname>
  <vcard:hasEmail rdf:resource="mailto:corky@example.com"/>
</vcard:Individual>
```

Figura 13. Uso de términos del vocabulario GEO
Fuente: Recuperado de https://www.w3.org/2003/01/geo/
Elaborado por: (Dan Brickley, 2003)

2.11.4. **GEO**

Como menciona (Dan Brickley, 2003), GEO es un vocabulario RDF básico que proporciona a la comunidad de la Web Semántica un espacio de nombres para representar lat (itude), long (itude) y otra información sobre cosas espacialmente localizadas, usando WGS84 como datos de referencia.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#">
  <geo:Point>
    <geo:lat>55.701</geo:lat>
    <geo:long>12.552</geo:long>
  </geo:Point>
</rdf:RDF>
```

Figura 14. Uso de términos del vocabulario GEO
Fuente: Recuperado de https://www.w3.org/2003/01/geo/
Elaborado por: (Dan Brickley, 2003)

2.12. Notificaciones en Tiempo Real

Tal y como describe (Rafael Veiga Cid, 2015), cuando queremos desarrollar aplicaciones que reciban notificaciones en tiempo real, podremos hacerlo utilizando diferentes técnicas. La mayor parte de ellas consisten en mantener una conexión con el servidor abierta el mayor tiempo posible, de tal manera que se puedan recibir en el lado del cliente las notificaciones del servidor.

Ejemplos de aplicaciones web con información en tiempo real:

- http://demo.kaazing.com/forex/
- http://rumpetroll.com/
- http://www.websocket.org/demos.html

En el desarrollo web se ha creado un nuevo término denominado Comet, que se utiliza para describir un modelo de aplicación web en el que una petición HTTP que se mantiene abierta permite enviar datos a un cliente mediante tecnología Push, sin que el cliente lo solicite explícitamente. En realidad Comet son múltiples técnicas que consiguen esta interacción.

Aquí vemos un gráfico en el que se muestran las diferentes técnicas.

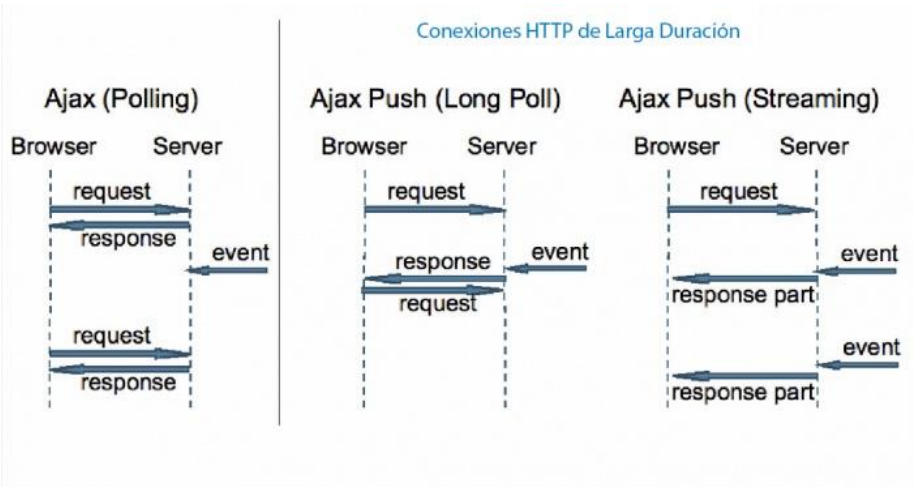

Figura 15. Técnicas de notificación en tiempo real
Fuente: Recuperado de https://manuais.iessanclemente.net/index.php/Node.js_y_Websockets#Introducci.C3.B3n_a_las_notificaciones_en_tiempo_real
Elaborado por: (Rafael Veiga Cid, 2015)

2.12.1. **Polling**

La técnica de **Polling** consiste en realizar peticiones ajax desde el cliente al servidor web cada X tiempo, a la espera de obtener nuevos datos desde el servidor.

PROS:

- Facilidad de implementación.

CONTRAS:

- Hay que hacer una conexión cada vez que queremos obtener datos.
- Muchas veces no se obtiene ningún resultado en la petición (si no hay datos nuevos en el servidor).
- Sobrecarga del servidor es proporcional al número de clientes conectados
- Para simular tiempo real habría que aumentar la frecuencia de las peticiones, lo que implicaría más sobrecarga del servidor.

2.12.2. **Long Polling**

La técnica de **Long Polling** o Polling extendido consiste en realizar peticiones Ajax desde el cliente al servidor web cada vez que queramos obtener datos actualizados. La diferencia con respecto al "Polling" es que en este caso la conexión se mantiene abierta hasta el tiempo máximo permitido de conexión, y si durante ese tiempo recibimos una notificación con datos de respuesta del servidor, los mostraremos en el lado del cliente, cerraremos esa conexión y abriremos una nueva a la espera de recibir nuevos datos.

Pros:

- Facilidad de implementación.
- Sobrecarga de trabajo menor en el servidor, ya que las conexiones permanecen abiertas durante más tiempo.
- Puede resolver problemas de escalabilidad asociados a la técnica de polling.
- Recomendable cuando nuestra aplicación necesita actualizaciones cada 30 segundos o más.

Contras:

- Hay que hacer una conexión cada vez que queremos obtener datos.
- Al tener las conexiones abiertas durante mucho más tiempo, puede ocurrir que no se estén enviando datos, pero se está consumiendo una conexión en el servidor, lo que podría provocar denegaciones de servicio si tenemos muchos clientes conectados.

- Cuando está la conexión abierta, sólo podremos recibir datos desde el servidor en esa conexión. Si queremos enviar datos, tendríamos que abrir una nueva conexión. No es por lo tanto bidireccional.

2.12.3. **Http Streaming**

La técnica de **Http Streaming** es similar a la técnica de "Long Polling" excepto en que la conexión nunca se cierra, incluso después de que el servidor haya enviado respuestas al cliente.

El cliente enviará una petición Ajax al servidor y recibirá las respuestas a medida que se vayan originando en el servidor, re-utilizando la misma conexión para siempre. Esta técnica reduce significativamente la latencia en la red, ya que no es necesario abrir y cerrar conexiones con el servidor. Por ejemplo Gmail utiliza o utilizaba esta técnica para actualizar su interface de correo en tiempo real.

Pros:

- Complejidad de implementación.
- Mejora de la sensación de tiempo real.
- Recomendable cuando nuestra aplicación necesita actualizaciones muy frecuentes en intervalos cortos de tiempo.

Contras:

- Los puristas consideran que esta técnica hace abuso del protocolo HTTP.
- Si el servidor envía datos de forma muy continuada, puede afectar al rendimiento de la red y de las aplicaciones Ajax. Puede ocurrir que nuestra aplicación no sea capaz de renderizar la página tan rápido como recibe datos.
- Si tenemos muchos clientes conectados, el servidor puede verse afectado al enviar tantos datos simultáneos en tan poco tiempo.
- Sigue siendo uni-direccional. El cliente sólo puede recibir respuestas del servidor, no puede hacer nuevas peticiones en esa conexión HTTP-streaming.
- Si nuestra aplicación envía datos cada 5 minutos, se recomienda usar long polling, ya que quizás el precio de re-abrir conexiones sea menor que el de mantener una conexión abierta durante largo rato sin recibir datos.
- La especificación HTTP 1.1. sugiere un límite de 2 conexiones simultáneas desde un cliente a un servidor. Esto quiere decir que si nuestra aplicación utiliza long polling o http streaming al mismo servidor, cualquier otra pestaña

abierta en el navegador con el mismo servidor daría como resultado bloqueo en la conexión por parte de nuestro navegador.

2.13. WebSockets

WebSocket es una tecnología que proporciona un canal de comunicación **bidireccional** y **full-duplex** sobre un único socket TCP.

Antes de la llegada de los **websockets en HTML5** las comunicaciones entre clientes web y servidores recaían en el protocolo HTTP.

A partir de ahora, la transmisión de datos puede fluir libremente sobre una **conexión websocket** que es **persistente** (siempre conectada), **full duplex** (bi-direccional de forma simultánea) y **extremadamente rápida**.

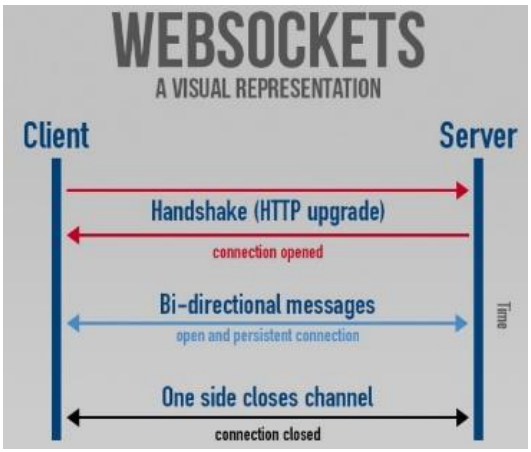

Figura 16. Representación visual de Websockets
Fuente: Recuperado de https://manuais.iessanclemente.net/index.php/Node.js_y_Websockets#Introducci.C3.B3n_a_las_notificaciones_en_tiempo_real
Elaborado por: (Rafael Veiga Cid, 2015)

Está diseñada para ser implementada en navegadores y servidores web, pero puede utilizarse en cualquier tipo de aplicación cliente/servidor.

Como las conexiones TCP ordinarias sobre puertos diferentes al 80 son habitualmente bloqueadas por los administradores de redes, el uso de esta tecnología proporcionaría una solución a este tipo de limitaciones proveyendo una funcionalidad similar a la apertura de varias conexiones en distintos puertos, pero multiplexando diferentes servicios WebSocket sobre un único puerto TCP (a costa de una pequeña sobrecarga del protocolo).

En el lado del cliente, WebSocket está ya implementado en Mozilla Firefox 8, Google Chrome 4 y Safari 5, así como la versión móvil de Safari en el iOS 4.2.1.

En el lado del servidor existen diferentes implementaciones, pero nosotros nos vamos a centrar en su implementación con servidor Node.js utilizando un módulo adicional llamado **socket.io.**

2.14. Tecnologías para el desarrollo de aplicaciones web

Es necesario hacer mención a distintas tecnologías seleccionadas debido a que son consideradas son el pilar básico sobre el cual se desarrollan las aplicaciones web, estas permitirán realizar un desarrollo fácil, intuitivo y rápido ya que utiliza metodologías básicas para la construcción de ambientes web.

2.14.1. **Protocolo HTTP**

Tal y como (Kioskea, 2014) establece, el protocolo denominado Hypertext Transfer Protocol, es el método más usual con el cual se intercambia información en internet, trasfiriendo las paginas o servicios Web que provienen de un servidor y se trasfiere hacia un cliente. Este protocolo trabaja a nivel de aplicación, para sistemas de información multimedia, lo que hace es trasladar ficheros de tipo HTML entre dispositivos, HTML es un lenguaje que trabaja en el lado del cliente caracterizado por emplear etiquetas para identificar a los diferentes elementos que lo conforman.

En la figura 10 se muestra el papel que desempeña este protocolo.

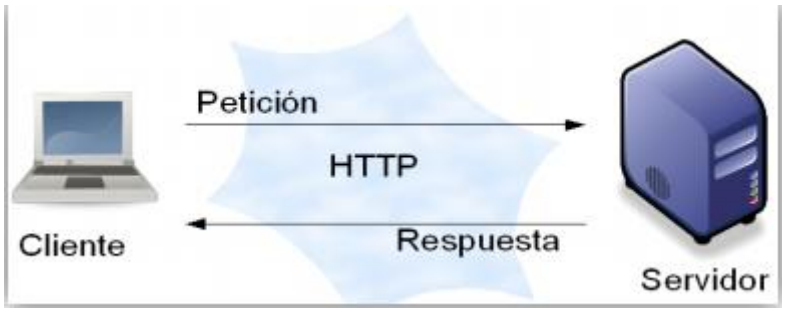

Figura 17. Función del protocolo HTTP
Fuente: http://dspace.ucuenca.edu.ec/bitstream/123456789/4303/1/tesis.pdf
Elaborado por: (Aguilar & Davila, 2013)

Entre las propiedades del protocolo HTTP se puede destacar las siguientes:

- Su esquema de direccionamiento es comprensible, utiliza el URL para localizar los sitios Web sobre los que hay que aplicar algún método. La forma general de un URL es servicio://host/fichero.ext, por ejemplo http://www.facebook.com/index.html
- Implementa la arquitectura cliente-servidor, HTTP se basa en el paradigma solicitud-respuesta, cuya comunicación se asienta sobre los protocolos TCP/IP. Por defecto, el número de puesto empleado por HTTP es el 80.

- Es un protocolo que trabaja sin conexión y sin estado, luego de que el servidor ha respondido una petición del cliente, la conexión se elimina entre ambos. Además no se guarda memoria del contexto de la conexión para siguientes conexiones.

Una vez definido claramente este protocolo, se requiere emplearlo de alguna manera en la plataforma. Para hacerlo se han establecido varios lenguajes de programación que ayudan a gestionar el manejo de HTTP, este al ser implementado en una estructura cliente-servidor, es evidente que se requieren dos tipos de lenguajes. Los conocidos Lenguajes del lado del cliente y Lenguajes del lado del Servidor.

2.14.2. **Lenguajes del lado del cliente**

Tal como menciona (Alvarez, 2006) para que un usuario, pueda ser un administrador y que a través de la aplicación pueda visualizar la información acerca del dispositivo que le corresponde, es necesario que dicha aplicación brinde este recurso, y lo hace mediante un lenguaje de programación que trabaja en el lado del cliente, es decir que se ejecuta en el dispositivo desde el cual el cliente accede a la aplicación web.

La gran ventaja de este tipo de lenguajes es que se evita la recarga de trabajo en la parte del servidor de la aplicación generando así, una mayor agilidad en el desarrollo de un proceso. Algunos ejemplos de este tipo de lenguajes son:

- HTML5
- CSS 3
- JavaScript
- VBScript
- Flash
- Flex, entre otros.

Es muy común trabajar con más de uno de estos lenguajes, dependiendo de las necesidades de la aplicación, en nuestro caso empleamos JavaScript de manera indirecta, debido a que utilizamos un framework conocido como EXTJS, de lo cual se hablara más adelante. A continuación se iniciara describiendo brevemente el lenguaje JavaScript, ya que se emplea en el presente trabajo aunque de una manera indirecta.

2.14.3. **JavaScript**

Desde el punto de vista de (Guillermo, Riera, Alfredo, & Garzón, 2013), existen procesos dentro de la aplicación del distributivo, como por ejemplo validaciones al momento de ingresar o actualizar datos, el lenguaje JavaScript se emplea para realizar estas validaciones de información y se podría describir como el complemento de

HTML, debido a que interactúa directamente con este, tomando la responsabilidad de los procesos que se desarrollaran en la aplicación a través del navegador.

La función principal de este lenguaje es presentar una página web dinámica, en la cual, los cambios o modificaciones realizadas en la aplicación web, se vean reflejadas de inmediato en el navegador.

Tal y como (Hazaël-Massieux, 2016) destaca que JavaScript, tiene muchas posibilidades, es orientado a objetos, maneja funciones y estructuras de datos complejas; existen dos características básicas de los lenguajes orientados a objetos que JavaScript no implementa: herencia y polimorfismo; aunque permite la creación y manipulación de objetos sencillos, además de la definición de métodos y propiedades para dichos objetos. La mayoría de los navegadores, en sus últimas versiones interpretan el código JavaScript integrado dentro de las páginas Web.

Un código escrito en JavaScript no siempre va a funcionar de la misma manera en dos navegadores distintos, esto se debe a que cada navegador posee un intérprete diferente; lo cual suele ser un inconveniente bastante molesto, una solución sería comunicarle al usuario que actualice su navegador o inclusive que lo cambie; otra opción sería desarrollar un archivo JavaScript dedicado a cada navegador, es una solución laboriosa pero útil.

2.14.4. **Lenguajes del lado del servidor**

Es muy común que el Administrador de la aplicación o un usuario en particular, requiera por ejemplo un reporte de una información en particular, la programación del "lado del servidor", se encarga de este proceso y se trata de una tecnología que consiste en el procesamiento de una petición que el usuario realizo anteriormente a través del navegador.

Esta petición se interpreta mediante un script que se encuentra en el servidor de la aplicación, con el objetivo de generar paginas HTML dinámicamente con la respuesta a la petición realizada.

Hoy en día existen lenguajes del lado del servidor, como Java, Python, Ruby, PHP, Asp entre otros; como se ha citado, Java y Python están liderando el mercado actual, debido a su portabilidad y constantes mejoras; Se profundizara en el lenguaje Java, ya que es la herramienta que se emplea en el presente trabajo.

2.14.5. **Java**

(Navarra, 2014) describe que este es un lenguaje de programación bastante robusto, basado en C y C++, orientado a objetos, basado en clases, concurrente y de

propósito general; se trata de un lenguaje compilado, es decir que requiere ser traducido a partir de su código fuente por medio de un compilador, en un archivo ejecutable para una plataforma determinada; una vez compilado, se puede ejecutar varias veces sin la necesidad de compilarlo en cada momento, en este sentido, Java es un lenguaje multiplataforma, pudiendo de esta manera ejecutarse en cualquier sistema operativo, con la particularidad de que requiere de una máquina virtual, denominada JVM.

La máquina virtual de Java, se trata de un artefacto, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial, el cual es generado por el compilador del lenguaje Java.

Algunas de las características de este lenguaje se citan a continuación:

- Orientado a Objetos
- Distribuido
- Robusto
- Seguro
- Indiferente a la arquitectura
- Portable
- Interpretado
- Dinámico
- Produce Applets
- Alto rendimiento

La novedad que aporta Java dentro de las nuevas generaciones de navegadores, es la capacidad de desplazar el control de la interactividad de los servidores hacia las computadoras de los usuarios.

Así como los lenguajes de programación, existen también técnicas que se aplican con el objeto de agilizar los procesos que se desarrollan en el servidor; tal es el caso de la técnica Ajax, la cual cumple un papel fundamental en el presente tema de fin de titulación.

2.14.6. **Ajax**

Así como lo manifiesta (Guillermo et al., 2013) todo el tiempo, en una aplicación web, se envían y reciben datos entre los clientes y el servidor, información o datos que necesitan ser actualizados automáticamente en la parte del cliente.

Hasta hace no mucho, la única forma empleada para realizar esta función era la de recargar absolutamente toda la página web, con los datos que se solicitaron desde el cliente, esto, con todas las dificultades que están implicadas, principalmente el tiempo que se toma para realizar la petición al servidor y consecuentemente la respuesta del mismo.

Ajax permite que se actualicen únicamente los datos requeridos por el cliente, sin la necesidad de recargar toda la página web, se trata de una técnica de desarrollo web para crear aplicaciones interactivas, conocidas como RIA's (Rish Internet Application's), las cuales han abarcado gran parte del mercado actual.

El objetivo de Ajax, es mantener una comunicación asíncrona entre la aplicación presentada al cliente y el servidor, esto significa que en cualquier momento se puede realizar peticiones al servidor, sin que este se encuentre en sincronía con el cliente, de esta manera se pueden comunicar indefinidamente estos dos elementos.

El cliente al momento de requerir datos adicionales, se solicitan al servidor, el acceso a los datos se realiza a través del objeto denominado XMLHttpRequest, lo hace en segundo plano, impidiendo el cambio de comportamiento de la aplicación Web o su visualización, por otra parte, el servidor, responde a este objeto de requerimiento mediante el estándar XMLHttpResponse, y lo realiza en conjunto con los datos que se solicitaron anteriormente.

2.14.7. **Framework**

Como lo hace notar (Gutiérrez, 2014), un framework brinda una estructura conceptual y tecnológica que ayuda a la parte grafica de un Sistema Informático, lo hace comúnmente con artefacto y módulos de software concretos, que se implementaran en una aplicación web, esto, con motivos de agilidad en la aplicación y sobre todo funcionalidad.

Actualmente existen numerosos instrumentos que nos facilitan este trabajo y vienen escritos para los distintos lenguajes de programación (del lado del cliente).

2.14.8. **Modelo, Vista, Controlador (MVC)**

Se trata de un patrón de arquitectura de las aplicaciones de software, su principal característica es que separa completamente la lógica de negocio de la interfaz de usuario y a su vez de la lógica de control empleada en el desarrollo. Este patrón de arquitectura, fue descrito inicialmente en el año 1979 para la compañía Smaltalk y se ha venido empleando desde entonces.

Muy comúnmente en Aplicaciones Web se ve reflejado en patrón de diseño MVC, se lo emplea debido a los constantes cambios que requiere la Aplicación a lo largo del tiempo, es importante mantener los bloques de trabajo bien definidos e independientes, de tal manera que los cambios realizados en un bloque, se verán reflejados en otro bloque sin necesidad de grandes cambios a nivel de código fuente. Desarrollando las partes del patrón MVC, tenemos que:

El modelo.- Es el responsable de acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.

Define las reglas de negocio (la funcionalidad del sistema). Un ejemplo de reglas puede ser: “Si la mercancía pedida no está en el almacén, consultar el tiempo de entrega estándar del proveedor”.

Lleva un registro de las vistas y controladores del sistema.

Si estamos ante un modelo activo, notificara a las vistas los cambios que en los datos pueda producir un agente externo (por ejemplo, un método que actualiza los datos de un docente, un temporizador que mantiene un seguimiento de la sesión del usuario entre otros).

El controlador.- Se encarga de recibir los eventos de entrada (un click, un cambio en un campo de texto entre otros).

La vista.- Responsable de recibir datos del modelo mostrándolos al usuario. Tiene un registro de su controlador asociado (normalmente porque además lo instancia). Puede dar el servicio de “Actualizacion()”, para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa que los cambios en los datos producidos por otros agentes).

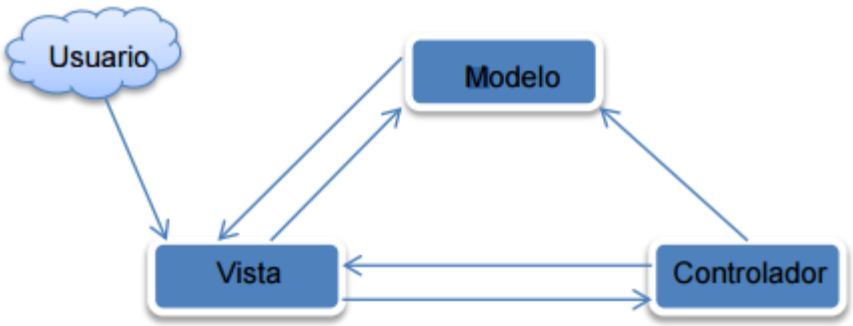

Figura 18. Función del protocolo HTTP
Fuente: http://dspace.ucuenca.edu.ec/bitstream/123456789/4303/1/tesis.pdf
Elaborado por: (Aguilar & Davila, 2013)

2.15. Herramientas para el desarrollo de aplicaciones web

Las herramientas que son detalladas a continuación han permitido un desarrollo ordenado, rápido y eficiente durante el desarrollo de la aplicación móvil, ya que cuentan con una interfaz muy amigable y robusta en cuando al desarrollo de distintos lenguajes de programación.

2.15.1. **NetBeans (V. 8.2)**

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE2 es un producto libre y gratuito sin restricciones de uso. NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring.

2.15.2. **Sublime Text 3**

Sublime Text es un editor de texto y editor de código fuente está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por esto aún conserva un modo de edición tipo vi llamado **Vintage mode**.

Se puede descargar y evaluar de forma gratuita. Sin embargo no es software libre o de código abierto y se debe obtener una licencia para su uso continuado, aunque la versión de evaluación es plenamente funcional.

2.16. Tecnologías para el desarrollo de aplicaciones móviles

A continuación se detalla las tecnologías que ayudaron con el desarrollo de la aplicación móvil, la cual forma parte esencial de la plataforma en cuanto a la extracción y transmisión de datos desde los sensores.

2.16.1. **Android Studio (V. 2.3.3)**

Android Studio posee distintos componentes que ayudan a la tarea de la construcción de aplicaciones; sistema de construcción basado en Gradle, la construcción de variantes y múltiples archivo APK, como también plantillas de código que ayudan a la creación de aplicaciones. Un completo editor de diseño con soporte para la edición de arrastrar y soltar el tema elementos. Facilidad de uso y compatibilidad de versiones, Código encoge con ProGuard y consumo de recursos cada vez menor con Gradle. Por último, soporte integrado para Google Cloud Platform, lo que hace más fácil de integrar Google mensajería en la nube y la App Engine.

En cuanto al desarrollo del flujo de trabajo, Android Studio posee un conjunto de herramientas encargadas, Adicionando a eso el posible acceso desde la línea de comandos las herramientas SDK. Lo importante de todo esto es que, Android Studio ofrece comodidad para los desarrolladores, ya que desde él es posible invocar, durante el desarrollo de aplicaciones, las herramientas necesarias como una forma más ágil de trabajo.

2.16.1.1. ***Etapas de desarrollo***

Entre las fases de desarrollo que abarcan la realización de aplicaciones en Android Studio encontramos cuatro etapas. La primera es la **configuración de entorno**; durante esta fase se instala y configura el entorno de desarrollo. Además se realiza la conexión a los elementos en donde se pueden realizar la instalación de las App, y se crean dispositivos virtuales Android (AVDS). La segunda fase abarca la **Configuración del Proyecto y Desarrollo**; durante esta se realiza la configuración del proyecto y el desarrollo del mismo. Hablamos de la creación de módulos que contengan recursos para la aplicación y archivos de código fuente. La tercera fase comprende las **pruebas, depuración y construcción de la aplicación**; A esta altura se construye el proyecto en un paquete (s) depurable .apk que se puede instalar y ejecutar en el emulador o en un dispositivo con Android. Se utiliza un sistema de construcción basado en Gradle. Con este se proporciona flexibilidad, variantes de construcción a la medida y la resolución de dependencias. En el caso de usar otro IDE se puede desarrollar el proyecto usando Gradle, y a su vez, instalarlo en un dispositivo que use ADB. Posteriormente se hace la depuración de la aplicación a través de los

mensajes de supervisión de dispositivos, más un dispositivo de registro de Android (Logcat) junto con la idea de IntelliJ. Además, Se puede usar un depurador JDWP compatible, adicionando las herramientas de depuración y de registro que se proporcionan con el SDK de Android. Ya al finalizar, se usan las herramientas de prueba SDK de Android para las pruebas a la aplicación.

CAPÍTULO 3. PROPUESTA DE LA SOLUCIÓN

En este capítulo se detalla, la solución implementada para el tema de fin de titulación en el cual se muestra las distintas aplicaciones que se han integrado para la construcción y puesta en marcha de la plataforma denominada “Geoamigo”. Para contar con una mayor acogida por parte de los distintos usuarios se ha adquirido un hosting y dominio aptos para su publicación en Internet. La plataforma cuenta con 2 módulos principales como son: 1) Aplicación móvil para la recolección de datos, y 2) Aplicación web para la visualización y publicación de datos.

3.1. Tecnologías implementadas en la plataforma

Se realiza una investigación de distintos contextos en los cuales puede llegar a ser útil la plataforma a desarrollar, para así empezar a realizar una lluvia de ideas con las cuales se puede tener una base con la cual iniciar.

Debido a que la plataforma que se desea desarrollar no ha sido modelada previamente, se ha realizado varios modelos iniciales tales como: aplicaciones móviles hibridas, canal de comunicación no real-time y uso de datasets estáticos; estos modelos se han ido puliendo con la investigación realizada, seleccionando los más óptimos y genéricos que se puedan integrar con distintas tecnologías actualmente existentes orientadas a Linked Stream Data.

Las tecnologías integradas en la plataforma han sido seleccionadas con el principal objetivo de evitar adquirir licencias que conlleven un costo económico, minimizando los obstáculos para proporcionar una solución mucho más fácil de integrar, las mismas se listan a continuación:

- MySQL 5.6
- Servidor GlassFish 4.1
- NodeJS 7.4
- Servidor Apache 2.4
- Virtuoso Universal Server
- Pubby

3.2. Consideraciones Técnicas

El funcionamiento de un sistema informático en la web tiene que ser igual e independiente del sistema operativo o versión del mismo instalado en el cliente. Esto significa que la aplicación debe ser escrita y consecuentemente ejecutada una sola vez, reflejando el mismo comportamiento siempre. Para lograr este objetivo se utilizan los protocolos, estándares y lenguajes de programación dedicados propiamente a las aplicaciones web.

A nivel técnico, existen varias tecnologías que se emplean al momento de desarrollar e implementar una aplicación web; en el presente documento se hablara de estas tecnologías divididas en dos grandes grupos: Tecnologías del lado del cliente y Tecnologías del lado del servidor, no sin antes mencionar las bases sobre las cuales se encuentran desarrolladas las mismas.

La base fundamental que se requiere conocer en un Sistema Informático, para la comunicación entre cliente y servidor, se da mediante los estándares establecidos por la W3C y la IETF (The Internet Engineering Task Force).

3.3. Arquitectura

La solución está orientada a que los usuarios puedan compartir información y a su vez reutilizarla para el propósito que requieran, este proceso se da por medio de cada uno de los colaboradores que la conforman.

Basándonos en el gran auge que están dando los dispositivos móviles al permitir obtener metadata mediante el uso de sensores integrados en estos tales como: GPS, giroscopio, huella dactilar entre otros, se ha optado por el desarrollo de una aplicación móvil nativa basada en Android debido a su robusto sistema operativo que nos permite extraer eficientemente la metadata de los sensores que posea el dispositivo.

Una vez instalada la aplicación en el dispositivo esta empezara a extraer la metadata del sensor del GPS (latitud, longitud, altitud), giroscopio (rumbo o dirección) y de la batería (conectado a una fuente de energía), la cual será transmitida por una canal de comunicación bidireccional, utilizando una conexión cliente-servidor. Ademas se proporciona características adicionales como: crear un recorrido, guardar puntos de interés y compartir esta información con otros usuarios. Una vez la información empiece a guardarse en la base de datos del sistema, se implementa una aplicación web para poder visualizar los datos de distintas maneras (en mapas geo-referenciados, reportes), permitiendo analizarlos para realizar mejoras que no solo ayudaran al usuario final sino que también permitirá proporcionar nuevas características útiles y a medida para los distintos tipos de uso que se dan en los medios de transporte.

Para tener una vista más gráfica, se ha realizado un modelado grafico de la arquitectura de la plataforma, mostrando cada uno de los módulos que la conforman, la cual podremos ver en la Figura 20.

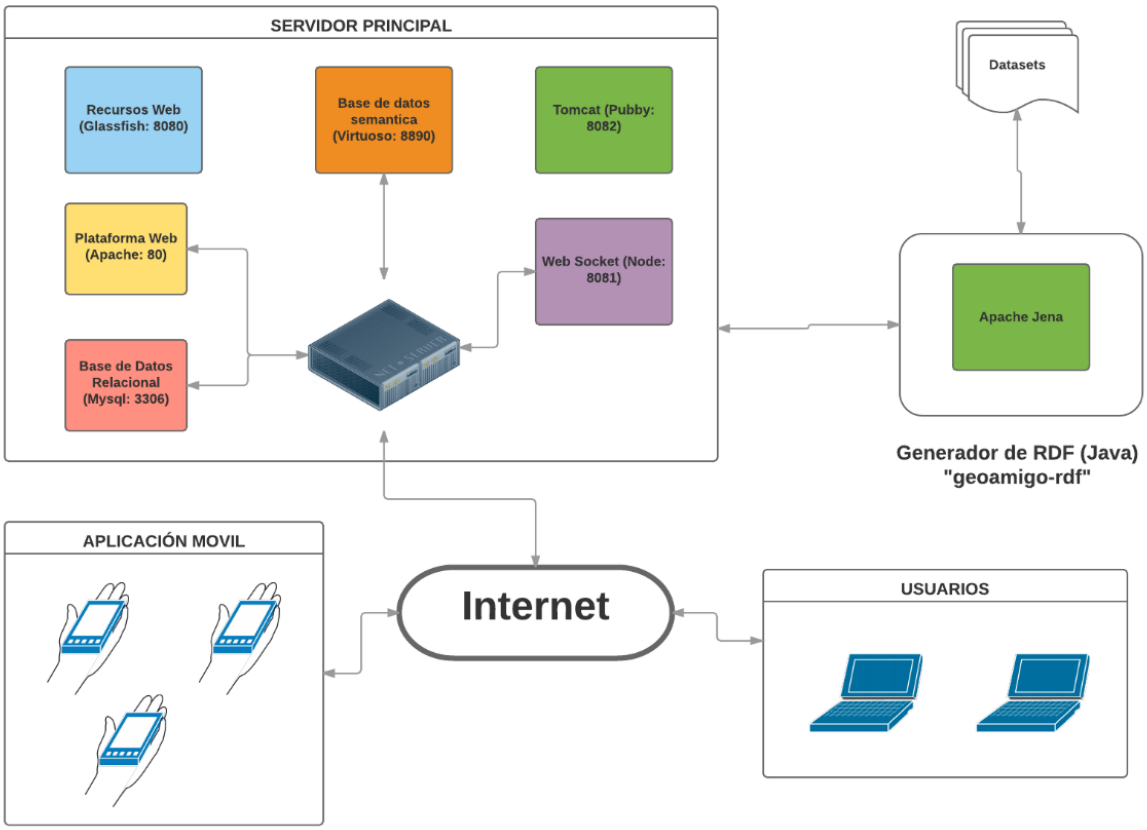

Figura 19. Arquitectura de la plataforma Geoamigo
Fuente y elaboración por: Autor

3.4. Servidor de Base de Datos (MySQL)

Este servicio lo utilizamos para el almacenamiento constante de la información, el servidor se levanta sobre el puerto 3306 (por defecto), pero puede ocuparse cualquier otro mientras este se encuentre libre.

Iniciando con la descripción, se adjunta la imagen de la base de datos relacional utilizada para el almacenamiento de la información. Mediante la cual se plasma el contexto sobre el cual trabaja la plataforma para la recolección de la información, además de algunas funcionalidades extras orientas al usuario tales como: poder comentar el recorrido o ruta realizada por otro miembro de la comunidad siempre y cuando esta sea catalogada como publica y especificar tipo de recorridos o rutas.

En la Figura 21, se ha plasmado el modelado lógico de la base de datos con sus tablas y atributos, que permitirá proporcionar un mejor entendimiento de su estructura.

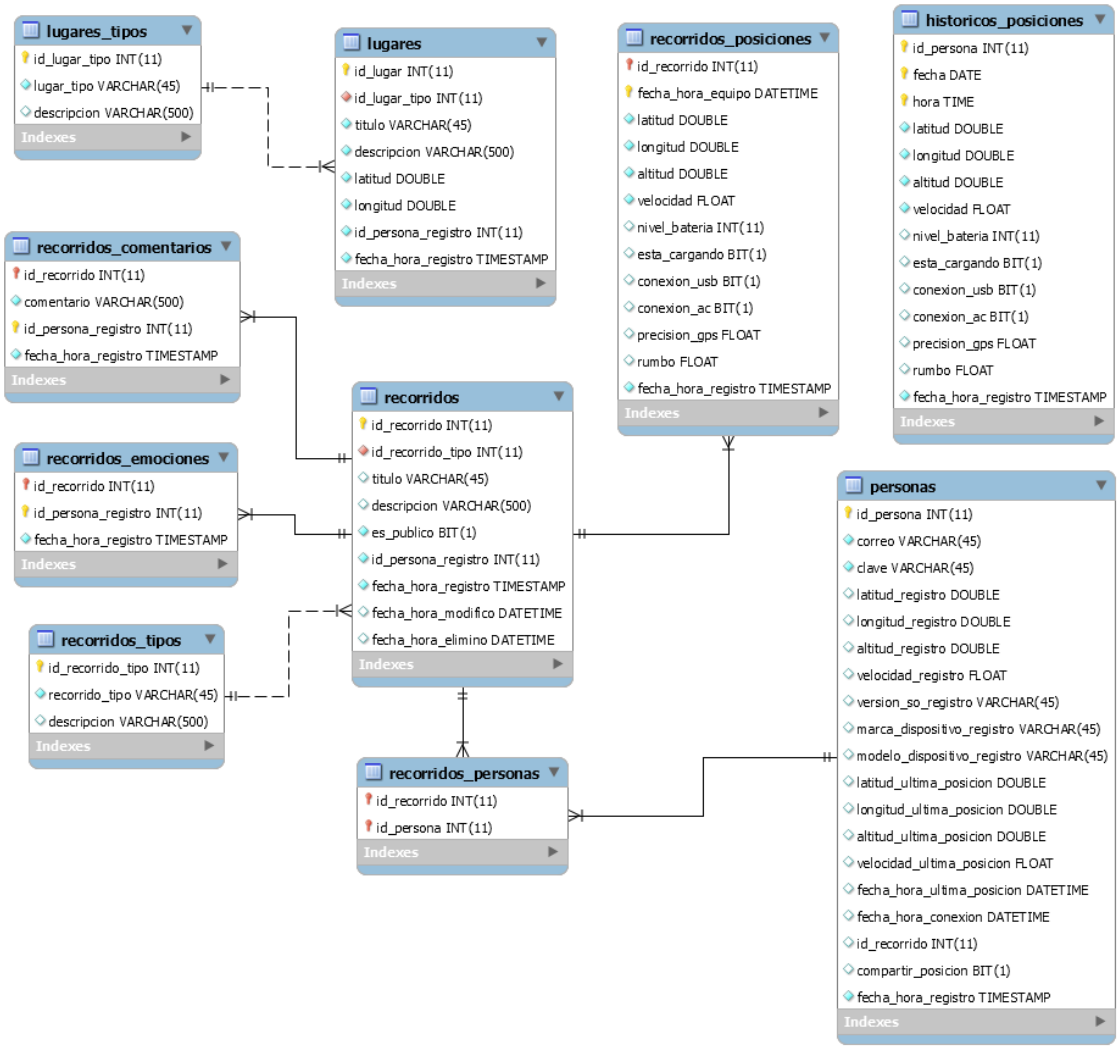

Figura 20. Modelo Relacional de la base de datos relacional Geoamigo
Fuente y elaboración por: Autor

A continuación se especifica cada una de las tablas y su respectiva función dentro de la plataforma:

- **Tabla recorridos.-** En esta tabla se alberga el detalle general del recorrido realizado por los miembros de la comunidad.
- **Tabla personas.-** Aquí se almacena información detallada del usuario que se ha registrado en la plataforma, así como metadata obtenida del dispositivo en uso.
- **Tabla recorridos_tipos.-** Se ha creado esta tabla para poder contar con una gran diversidad de tipos de recorridos tales como: de transporte público, de transporte privado, de caminatas por senderos, recorridos vehiculares personales entre otras

- **Tabla recorridos_emociones.-** En vista de proveer una mejor aceptación y confiabilidad a los recorridos públicos que los usuarios realicen, se ha creado esta tabla para almacenar el grado de aceptación.
- **Tabla recorridos_comentarios.-** Muy similar a la tabla anterior, se utiliza para realizar un feedback de los recorridos por medio de los usuarios, y estar en constante retroalimentación.
- **Tabla recorridos_posiciones.-** En esta tabla se almacena cada una de las posiciones geo-referenciadas que conforman una ruta, acompañada de la respectiva metadata obtenida del dispositivo móvil.
- **Tabla historicos_posiciones.-** A medida que las posiciones georeferenciadas son transmitidas al servidor, estas se van almacenando para que los usuarios puedan obtener reportes basados en un rango de fechas.
- **Tabla lugares.-** Debido a que a medida que realiza un recorrido por distintos lugares, se ha brindado una funcionalidad en la cual el usuario pueda almacenar puntos de interés para una futura visita o visualización.
- **Tabla lugares_tipos.-** Así como los recorridos cuentan con un tipo, se ha acopla un contexto similar a los lugares o puntos de interés que el usuario almacene, para poder abarcar varios escenarios en los cuales la plataforma se pueda utilizar.

Una vez comprendido el Modelo de la Base de Datos relacional utilizado, se procede a especificar los canales de comunicación utilizados para la interconexión entre los distintos módulos que conforman la plataforma.

3.5. Servidor de recursos web (Glassfish 4.1)

Este servicio se utiliza principalmente para comunicarse con la información almacenada en la base de datos. Se han generado varios recursos web bajo REST-FULL tales como: autenticación, recorridos entre un rango de fechas, recorridos públicos o privados realizados por el usuario, configuración de los recorridos entre otros que son utilizados tanto por la aplicación web como por la aplicación móvil, aplicando principios de interoperabilidad para lo cual fueron diseñados. Este servicio se enfoca a ser utilizado más en contextos relacionados a reportes, de información previamente almacenada.

Para levantar estos recursos web se utiliza el servidor Glassfish en su versión 4.1 sobre el puerto 8080, el cual nos permite “desplegar y ejecutar” nuestro servidor para así tener activa la conexión. Además posee una comunicación hacia la base de datos

para establecer la lectura y escritura de la información a medida que el usuario interactúa con el sistema.

3.6. Servidor Web Sockets (NodeJs 7.4)

Para establecer una conexión más directa entre la aplicación móvil y los distintos usuarios con los cuales se desea compartir la ubicación durante un recorrido, se implementó una comunicación por medio del protocolo HTTP conocida como (Web Sockets) bajo el servidor NodeJs v. 7.4 que nos permite compilarlo y ponerlo en marcha para realizar el stream de datos entre los miembros de la comunidad.

Este servidor se caracteriza por tener potentes librerías tales como **socket.io**, la cual permite realizar una conexión real-time y bidireccional. Este servidor está funcionando sobre el puerto 8081 en nuestro servidor, pero puede ser configurado en cualquier puerto disponible.

El uso principal de este servicio en la plataforma lo realiza la aplicación móvil, para transmitir los datos geo-referenciados y la respectiva metadata a todos los miembros de la comunidad permitidos por el usuario.

3.7. Servidor de aplicaciones web (Apache 2.4)

Para poder visualizar la información provista por cada uno de los miembros de la comunicada, se ha implementado una plataforma web, la misma que cuenta con varios sub-módulos que permiten dar una mejor perspectiva de la información almacenada.

Apache proporciona al usuario una interfaz liviana, en nuestro servidor este servicio esta levantado sobre el puerto 80, ya que es el puerto común para la visualización de aplicaciones web.

En cuanto a cada uno de los sub-módulos en la aplicación web, se detalla a continuación.

- **Mapas Geo-referenciados (Google Maps y Mapbox).-** El uso de 2 distintos mapas, se lo ha realizado debido a que Google Maps proporciona una interfaz fácil para plasmar reportes históricos de recorridos y posición actual pero no cuenta con funcionalidades interactivas para trazar un recorrido y visualizarlo en tiempo real, para lo cual se ha optado por el uso de Mapbox, el cual cuenta con una gran gama de capas intuitivas para el usuario.
- **Reportes.-** Se utiliza reportes obtenidos por medio de recursos web bajo REST-FULL mencionado en el apartado 4.2. Además de Opciones para exportar la información en formato KML (Google Earth) y Excel.

- **Vistas.-** Mediante estas vistas se puede ver la cantidad de afluencia en los distintos sitios por los cuales recorre el usuario.
- **Dataset.**- Este sub-modulo nos muestra la información publicada y enlazada con distintos sitios por los cuales el usuario a pasado mientras realiza un recorrido.

3.8. Generador de RDF (Apache Jena 2.0)

Para poder generar un archivo RDF con toda la información almacenada en nuestra base de datos relacional, se desarrolló una aplicación sobre Java utilizando la librería APACHE JENA, la cual nos permite crear clases de nuestro grafo y una vez cargada la información nos exporta un archivo RDF, para la posterior carga en nuestra base de datos semántica.

Para contar con una mejor flexibilidad en cuanto al uso de distintos **datasets**, la aplicación que hemos desarrollado nombrada como "geoamigo-rdf" cuenta con un archivo de configuración para colocar el link del **dataset** que creamos conveniente y la respectiva consulta a utilizar, enlazando nuestra información con el **dataset** que mejor se acople a nuestras necesidades.

La aplicación "geoamigo-rdf", cuenta con un algoritmo que permite conectar con la base de datos relacional y obtener todos los recorridos públicos de los usuarios con sus respectivas posiciones, luego obtiene la información del dataset previamente configurado en el archivo de configuración; procede hacer una comparativa entre el inicio y fin del recorrido hacia los distintos lugares, sitios, localizaciones o posiciones geo-referenciadas publicadas en los dataset, obteniendo así las posiciones más cercanas al lugar y enlazando los recorridos a estas, para posteriormente exportarlas en un RDF.

A continuación, en la Figura 22 se visualiza el grafo utilizado para la generación del archivo RDF, y el mapeo respectivo en la aplicación "geoamigo-rdf".

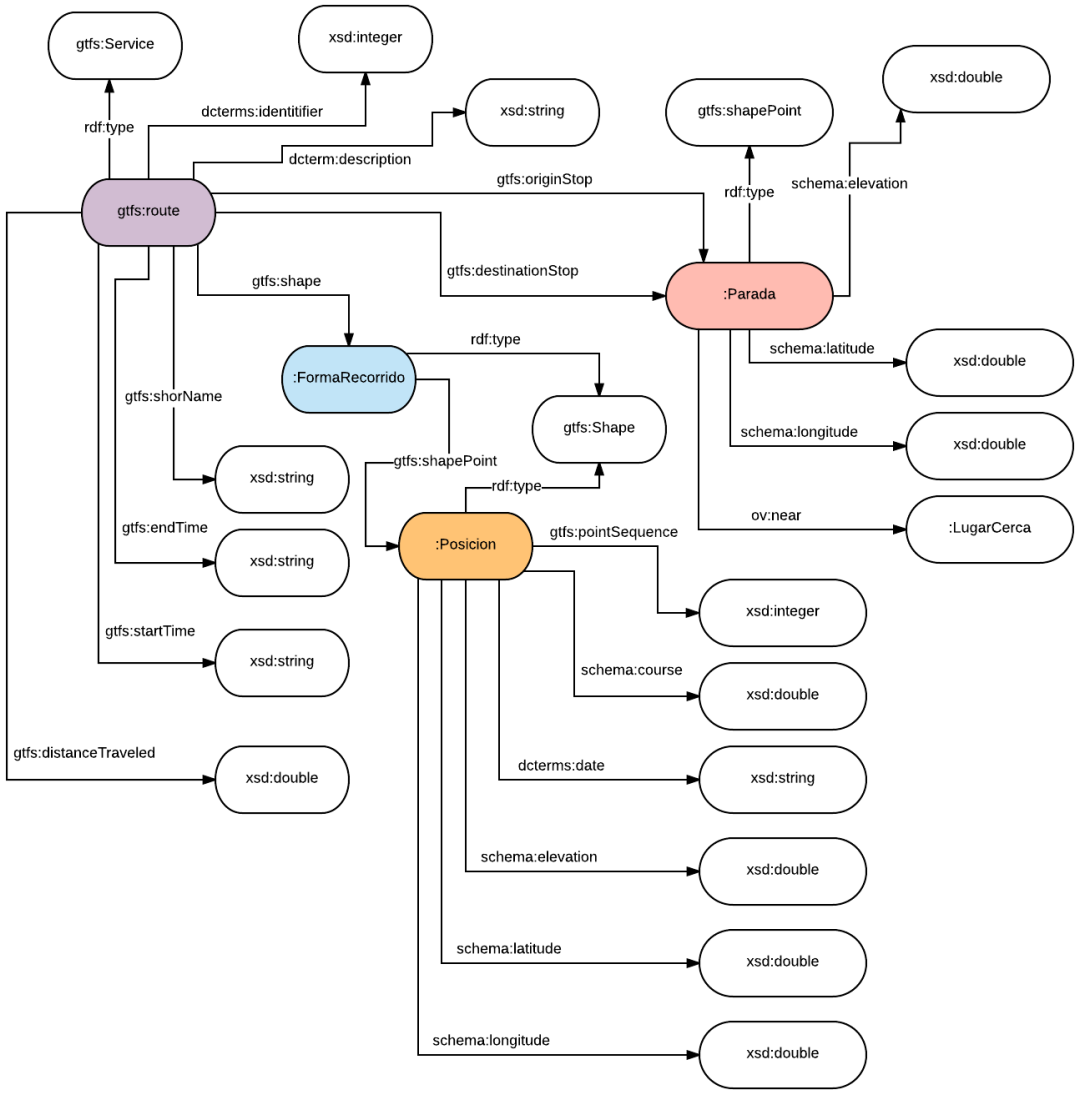

Figura 21. Grafo Geoamigo
Fuente y elaboración por: Autor

3.9. Servidor de Base de Datos Virtuoso (Virtuoso)

Esta base de datos semántica utiliza el puerto por defecto 8890 para levantar el servicio, pudiendo ser cambiado a otro puerto mientras este se encuentre disponible. La ventaja que nos proporciona esta base de datos, es permitir cargar un archivo RDF que contiene los datos obtenidos en nuestra base de datos relacional y procesado sobre la librería de Java Apache Jena para la respectiva correlación de la información.

Una vez cargado el archivo RDF, se configura un link por medio del cual podrán conectarse distintas interfaces tales como Pubby (mencionado en el apartado 4.5) o la librería PHP-SPARQL que también permite la conexión con nuestra base de datos semántica.

Si planteamos algún termino de comparación entre Pubby y PHP-SPARQL, nos encontramos con la simple diferencia que Pubby nos provee una estructura lista para configurar, mientras que PHP-SPARQL solo nos permite la conexión y el respectivo CRUD basado en SPARQL, sin una interfaz lista para visualizar la información tal y como lo hace Pubby.

3.10. Servidor Tomcat (Pubby)

Tomcat permite levantar el proyecto Pubby, mismo que nos permite publicar una interfaz conectada a la base de datos semántica, logrando así una mejor visualización para el usuario a través de un archivo RDF previamente cargado. En nuestro servidor **Tomcat** trabaja sobre el puerto 8082 pero puede ser cambiado por cualquier puerto disponible.

Pubby cuenta con un archivo de configuración para establecer los valores que se requieran como: el **dataset** al cual nos queremos conectar, y los prefijos a utilizar para cargar los enlaces hacia otros recursos. La flexibilidad que se maneja en este proyecto facilitó la presentación del dataset, y además otorga una interfaz disponible para cualquier persona que requiera ocupar nuestro dataset.

CAPÍTULO 4. PLATAFORMA ORIENTADA A LINKED DATA

En este capítulo detalla cada una de las características con las que cuentan la plataforma “Geoamigo”, que permitirán conocer cada uno de los módulos y sub-módulos que la conforman.

4.1. Aplicación móvil

Esta aplicación está desarrollada para dispositivos móviles con Sistema Operativo Android desde la versión 4.1 hasta la actual 6.0.2, el objetivo principal de esta es recolectar información que transmitan los sensores del dispositivo en el cual se instale, tales como: GPS, Batería, Giroscopio y Acelerómetro.

Para realizar la respectiva comunicación con el servidor, es necesario establecer una comunicación Web Sockets, la misma que se hace mención al final del Capítulo 2. Se ha optado por implementar esta tecnología, para reducir el consumo de megas durante la transmisión de la data recolectada desde los sensores del dispositivo y enviadas al servidor. Para ello se utiliza NodeJS v. 7.4, que es un servidor basado en JavaScript junto con la librería socket.io la cual nos provee los métodos y funciones necesarios para establecer una comunicación robusta y controlada, no solo desde la aplicación móvil, sino hasta la transmisión directa hacia la aplicación web.

La interfaz de la aplicación cuenta con varios sub-módulos sencillos que le permitirán al usuario una interacción natural y simple con la App. Además para que sea aún más sencilla el uso e instalación de esta, se ha optado por realizar una publicación de la versión final en el PlayStore.

A continuación se detalla cada uno de los sub-módulos e interfaces que el usuario una vez instalada la App, será capaz de visualizar.

4.1.1. **Formulario de registro**

Esta interfaz está desarrollada con la finalidad de que el usuario pueda crear una cuenta con un correo electrónico de uso y una contraseña para dar seguridad al acceso, esta cuenta que se crea permitirá almacenar toda la información recolectada mientras este en uso. El sistema cuenta con controles que le permiten validar si alguna cuenta ya se encuentra existente.

Podemos ver la Figura 23, que muestra los campos requeridos para la apertura de la cuenta.

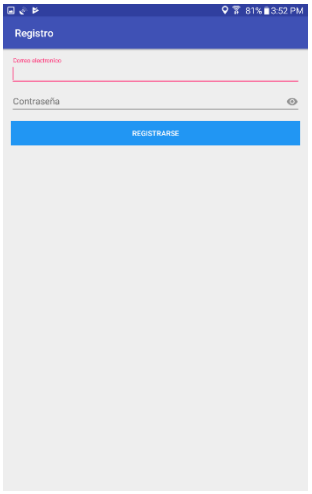

Figura 22. Interfaz de registro
Fuente y elaboración por: Autor

4.1.2. **Autenticación**

Esta interfaz permite al usuario dirigirse al Formulario de Registro en el caso de necesitar crear una cuenta nueva, o colocar las respectivas credenciales de autenticación para acceder a las funcionalidades de la aplicación móvil. Podemos ver su diseño en la Figura 24.

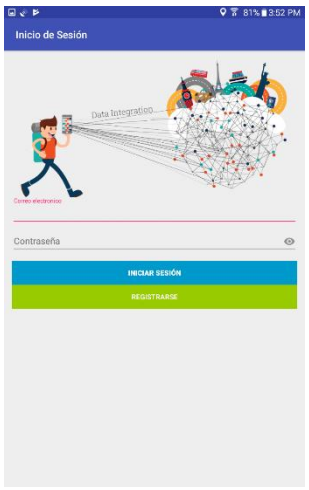

Figura 23. Interfaz de autenticación
Fuente y elaboración por: Autor

4.1.3. **Rastreo**

Una vez que el usuario se haya autenticado correctamente podrá acceder a las funcionalidades del sistema, previamente a esto y para que pueda cumplir el objetivo de la recolección de información, es necesario activar el sensor del GPS, una vez el sistema verifique que el sensor está activo, procederá a guardar toda la data (usuario que ha iniciado sesión) y metadata como (latitud, longitud, altitud, velocidad, rumbo, conexión a una fuente de poder, batería del teléfono y fecha y hora actual del

dispositivo) en una base de datos local, conocida como Sqlite, misma que viene junto con el sistema operativo de Android.

La interfaz cuenta con un mapa geo-referenciado el cual contara con algunas opciones como: Ubicarse en la última posición y Seguir la última posición. Además en la parte inferior, se encuentra una barra de opciones relacionadas con el inicio de un nuevo recorrido, e informativa, ya que muestra al usuario a qué velocidad va en distintos todos de color, para que pueda guiarse por ellos. En la Figura 25 podemos observar el diseño que muestra la ubicación del usuario en el mapa del dispositivo móvil.

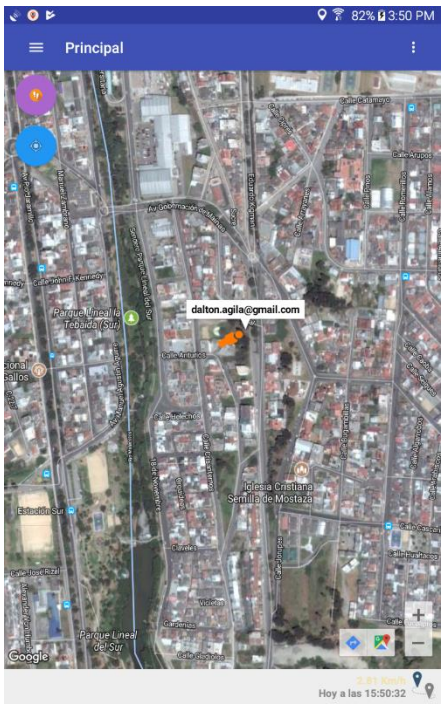

Figura 24. Interfaz de rastreo
Fuente y elaboración por: Autor

4.1.4. **Puntos de interés**

Una de las funciones de la interfaz de rastreo es permitir al usuario ir guardando puntos de interés a medida que va recorriendo el Mapa geo-referenciado, para ser capaz de acceder a esta funcionalidad, el usuario deberá mantener presionado la posición en el mapa que desee guardar. Posteriormente procediendo a llenar el formulario de registro del punto de interés como se muestra en la Figura 26.

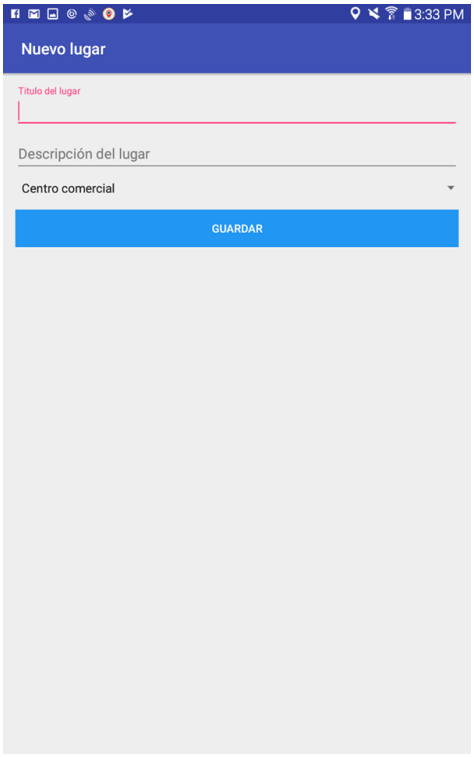

Figura 25. Interfaz de registro de nuevo lugar
Fuente y elaboración por: Autor

4.1.5. **Recorridos**

En el panel inferior de la interfaz de rastreo, el botón o imagen ubicada en la parte derecha permitirá iniciar un nuevo recorrido en tiempo real. Una vez se active esta opción, en el mapa se empezará a trazar el recorrido por el cual se encuentra dirigiendo el usuario, con los respectivos colores que van especificando la velocidad a la cual va, con los respectivos colores y tonos de intensidad, desde el color más suave que son velocidades menores a 10Km/h hasta velocidades mayores a 60Km/h, tal como podemos observar en la Figura 27.

La transmisión de la información se realiza en un intervalo de 5 seg con el servidor por medio de websockets, y en el caso de no contar con una conexión a estable, procederá almacenar los datos en memoria, para que luego sean enviados al servidor apenas la conexión se estabilice.

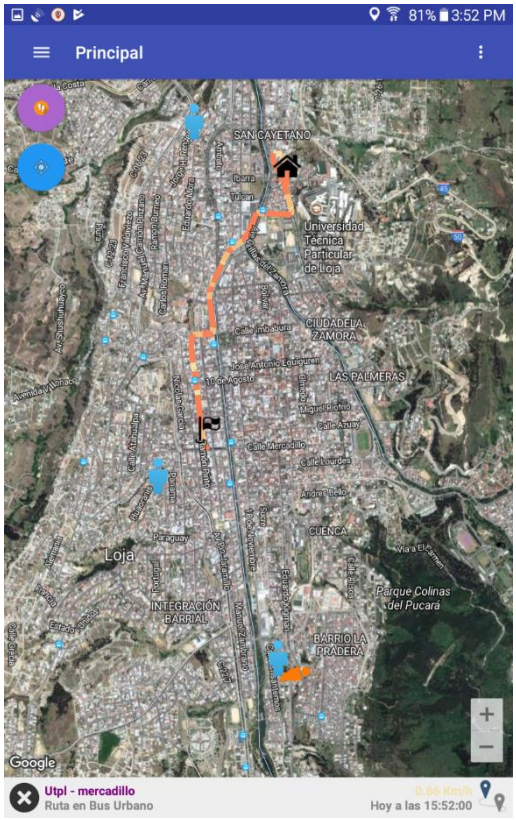

Figura 26. Trazado de un recorrido
Fuente y elaboración por: Autor

Cada recorrido puede ser editado o eliminado por el usuario. En el caso de requerir modificar la información del recorrido podrá hacerlo en el formulario que se muestra en la Figura 28, Cuenta con varios campos como el título del recorrido, descripción de recorrido, que tipo de recorrido o ruta posee y una de sus características es la privacidad del mismo, tal como ser público (que se comparte con todos los usuarios de la comunidad) o privado (solo se comparte con personas especificas), y la especificación del medio de transporte que usa para realizar ese recorrido.

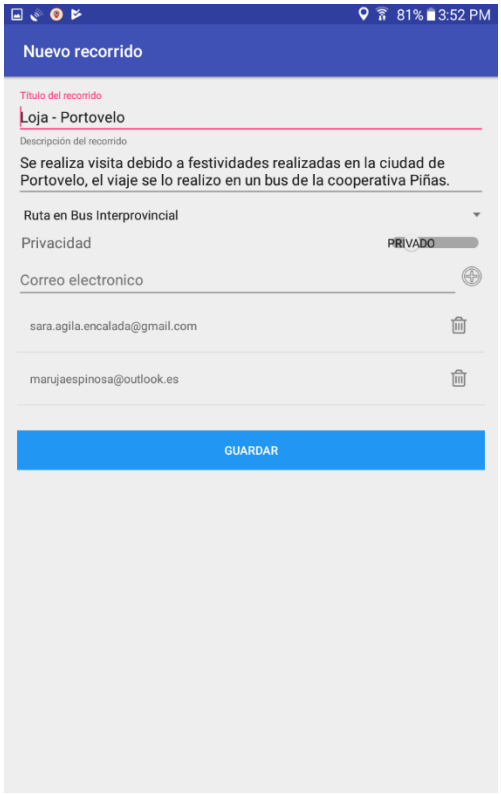

Figura 27. Formulario de edición de recorrido
Fuente y elaboración por: Autor

Una característica en particular con la que cuenta el sistema es que permite a los usuarios unirse a un recorrido siempre y cuando este haya sido compartido por otro miembro de la comunidad. Podemos ver la Figura 29 que muestra estas características, y se puede acceder a ellas desde interfaz de rastreo, por medio del botón de menú.

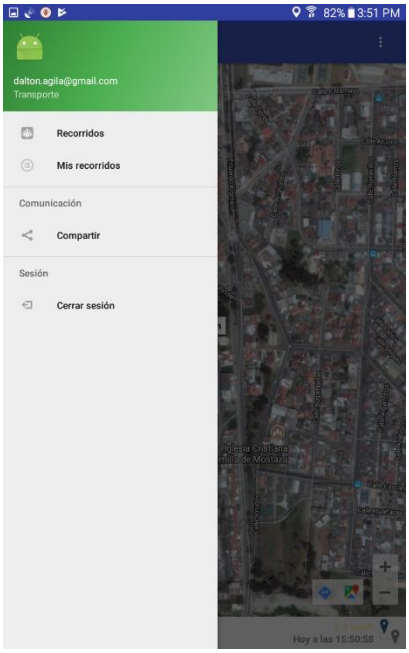

Figura 28. Menú de opciones
Fuente y elaboración por: Autor

Una de las opciones es la de **Recorridos** la cual permite al usuario tener una vista general de todos los recorridos compartidos por los demás miembros de la comunidad. Mostrando detalles del recorrido tales como: Cuantas personas están unidas al recorrido, cuando se creó el recorrido, que tipo de transporte uso para trazar el recorrido y opciones para unirse (Consiste en que el usuario puede visualizar un recorrido y todos los miembros que se encuentren unidos, podrán ver su posición actual) y compartir un recorrido (que permitirá la visualización del recorrido en un navegador). Para ver el diseño nos podemos fijar en la Figura 30.

Figura 29. Interfaz de Recorridos
Fuente y elaboración por: Autor

Otra opción es **Mis Recorridos** la cual permite al usuario tener una vista general solo de sus recorridos, con características similares a la opción previamente mencionada pero con funcionalidades para Eliminar y Editar un recorrido tal como vemos en la Figura 31.

Figura 30. Interfaz de Mis Recorridos
Fuente y elaboración por: Autor

Cuando un usuario se une a un Recorrido, autoriza al sistema que comparta su ubicación actual con todos los usuarios que también se encuentren unidos a ese recorrido. Una vez unido, se desbloquean nuevas opciones o funcionalidades en el mapa de rastreo para poder conocer quienes se encuentran en ese momento conectado y su ubicación geográfica; podemos ver estas características en la Figura 32. Además puede ver el trazado del recorrido. Y opciones para cancelar la respectiva unión al mismo.

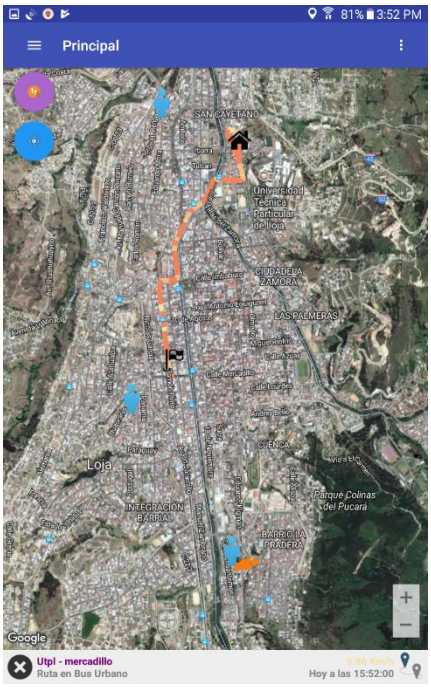

Figura 31. Unión a un recorrido
Fuente y elaboración por: Autor

4.1.6. **Cierre de sesión**

Mientras el usuario haya iniciado sesión, un servicio interno empezara a correr, para capturar toda la información que se necesita. Pero si ya no necesita de ello, podrá cerrar la sesión y el servicio finalizaría y volvería a la interfaz de autenticación.

4.2. Aplicación Web

Para poder realizar una visualización correcta y entendible de toda la información recolectada por los dispositivos móviles se ha visto necesario desarrollar una interfaz web, que le permitirá al administrador extraer y mostrar los datos, ya sean en un mapa geo referenciado estático o uno en tiempo real.

Para este desarrollo se ha realizado bajo lenguajes de programación como son: HTML5, JavaScript, PHP7, CSS3 conjuntamente con mapas Geo referenciados de Google Maps (Ambiente estático) y Mapbox (Ambiente en tiempo real).

Para establecer las características principales de la interfaz web, se procedió con un análisis de la información a recolectar por los dispositivos móviles, y basándose en que se contaba con datos geo referenciados, se propuso el uso de mapas más dinámicos en cuanto a la carga de datos, tal como lo es MapBox. Esta opción permitió darle más vida a la visualización de la información y poder detectar zonas de calor en las cuales se convergen varios puntos geográficos.

Una vez se estabilizo una de las versiones, y se quedó claro en el objetivo se vio conveniente publicar el sistema junto con un dominio en internet, para tener acceso desde cualquier navegador con conexión a internet.

4.2.1. **Módulo de Autenticación**

En la interfaz de la Figura 33 se muestra la autenticación de usuario para acceder a la visualización de los datos compartidos por los distintos miembros de la comunidad. Esto se realiza por medio de las credenciales de autenticación que se colocaron al crear la cuenta desde el dispositivo móvil.

Figura 32. Interfaz de autenticación
Fuente y elaboración por: Autor

4.2.2. **Módulo de Visualización**

Para mostrar la información se ha procedido con la carga de la data de todos los usuarios en un mapa geo referenciado y así conocer la última posición de cada uno de ellos. Además la respectiva lista de usuario con múltiples opciones que permitirán mostrar, los recorridos realizados con toda la meta data capturada desde los dispositivos móviles.

Para poder cargar este mapa, se procedió a crear una cuenta en Google Maps y así obtener una llave para utilizarla en el proyecto web, la carga de información en las distintas formas, se tomó del API de Google Maps V3, cada uno de los usuarios esta expresado como un marcador en el Mapa, y para el trazado de la ruta se utilizaron Polilineas dinámicas para colorear las diferencias de velocidad, podemos ver su interfaz en la Figura 34.

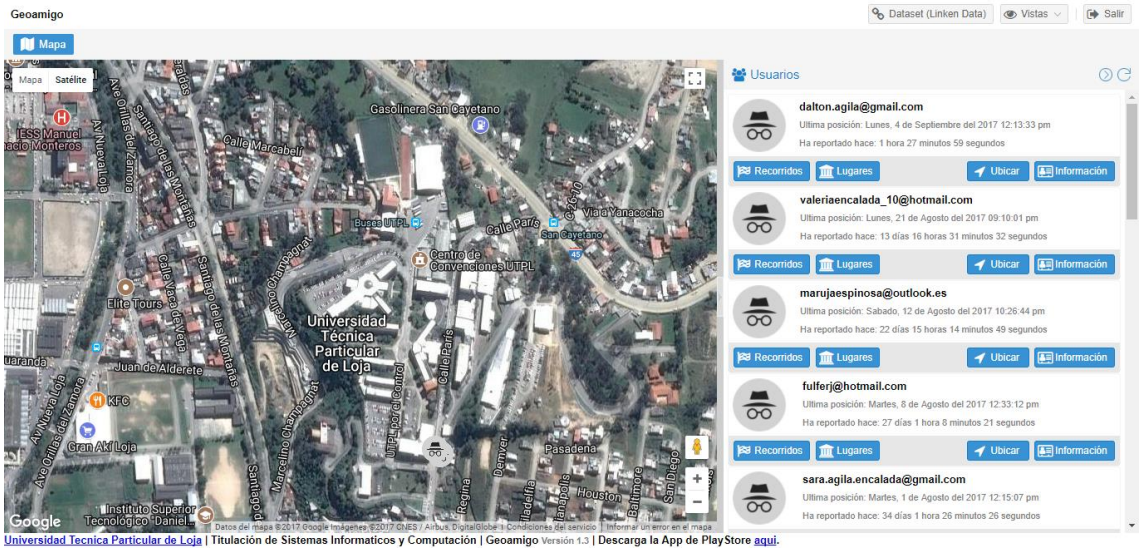

Figura 33. Interfaz de visualización estática
Fuente y elaboración por: El autor

A continuación en la Figura 35, se puede mostrar una lista de usuarios en orden ascendente basados en quien ha transmitido la última información. Cada uno de estas cuentas con opciones de visualización en el mapa, carga de lugares y recorridos realizados.

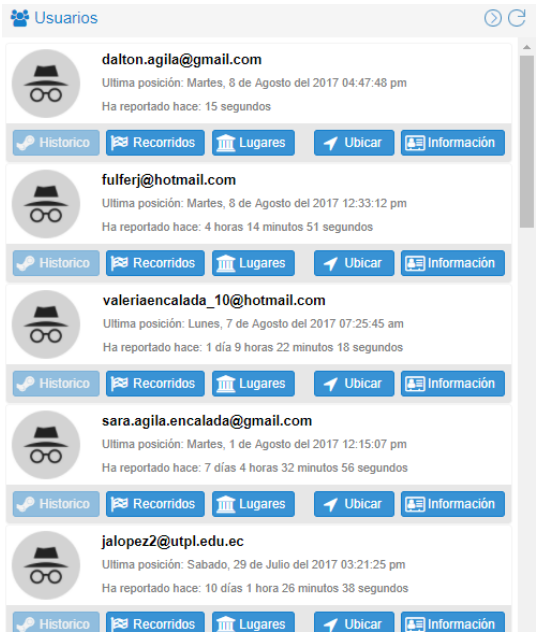

Figura 34. Lista de usuarios
Fuente y elaboración por: Autor

4.2.3. **Módulo de Reportes**

Las características de cada uno de los recorridos y detalle especifico de la meta data capturada durante la trasmisión de información es capaz de ser mostrada en el sistema web. Los recorridos guardados por los usuarios pueden ser visualizados en el mapa u obtener un detalle del recorrido con un resumen de toda la información obtenida. Expresado en distintos tonos de color para campos que son necesarios expresar de mejor manera, tal y como se muestra en la Figura 36.

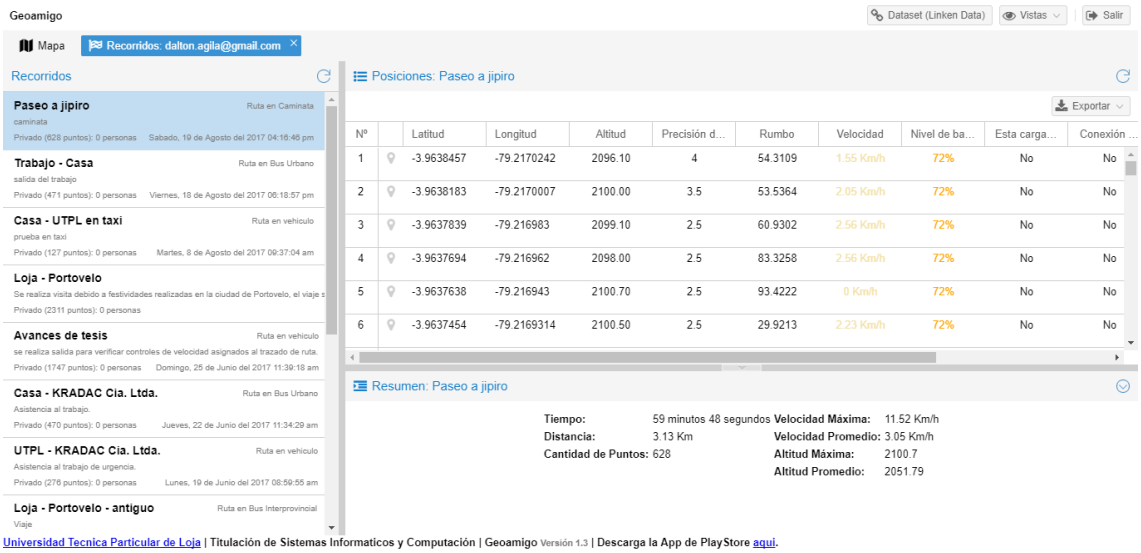

Figura 35. Módulo de Reporte de Recorridos
Fuente y elaboración por: Autor

4.2.4. **Módulo en tiempo real**

Para poder visualizar la información en tiempo real, se ha visto necesario implementar toda la data en un mapa de geo-referenciación dinámico, como lo es MapBox. Esta implementación permite una carga masiva de datos, de forma sencilla y liviana para que no ralentice la carga de datos masivos al intentar graficarlos.

Para cada una de las cargas de información se trazara los recorridos, con opciones de la visualización de los datos, basados en fechas, tiempos de carga y una actualización de los datos.

El objetivo principal de modelar la información de esta manera se debe a que se puede notar el como todos los datos convergen en puntos específicos para un mejor entendimiento de la información. En la parte inferior se puede notar la actualización de la información que cada uno de los usuarios está compartiendo en ese momento. Para ver su diseño, ubíquese en la Figura 37 que se encuentra a continuación.

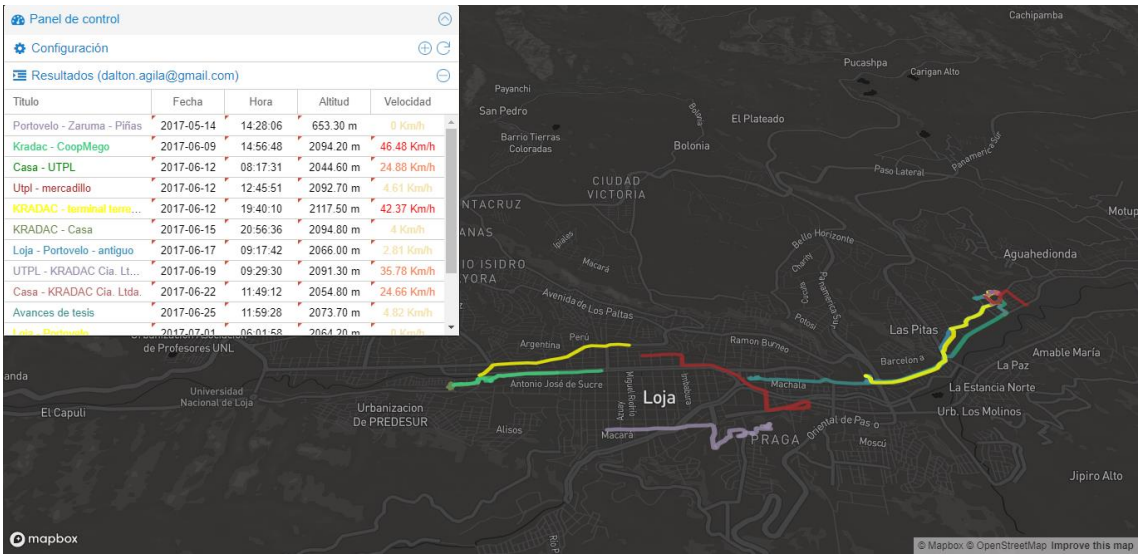

Figura 36. Interfaz en tiempo real
Fuente y elaboración por: Autor

4.2.5. **Dataset**

Para la publicación de los datos se ha hecho uso del frondend Pubby, desplegado y ejecutado sobre Tomcat 9 en el puerto 8082, Pubby nos proporciona una configuración hacia el dataset y genera una interfaz web de toda la información a publicar. Los datos publicados actualmente hacen referencia a las rutas compartidas públicamente por los usuarios, cada una de las cuales posee un enlace al lugar más cercano hacia punto de origen y destino de la ruta, como podemos ver en la Figura 38.

Figura 37. Interfaz publica del dataset Geoamigo
Fuente y elaboración por: Autor

CONCLUSIONES

A continuación se detallas las conclusiones relacionadas a la investigación e implementación de la plataforma.

1. Basado en la implementación de Web Sockets (Notificación en tiempo real), se puede observar que el consumo de megas es mucho menor al usar la aplicación móvil con el uso de esta tecnología, ya que nos proporciona un canal abierto para la comunicación bidireccional entre servidor y clientes.
2. En cuando al desarrollo nativo de la aplicación móvil bajo el Sistema Operativo de Android, entre las versión 4.1 (Jelly Bean) hasta la 8.0 (Oreo) existe una estabilidad de la aplicación, considerando que a partir de la versión 6.0 (Marshmallow) Android gestiona los permisos en tiempo de ejecución (Esto implica que puede revocar los permisos hacia los sensores mientras la aplicación se encuentre ejecutando).
3. Para poder contar con una buena visualización de los datos geo referenciados obtenidos por los dispositivos móviles, se debe cargar previamente la información para evitar retardo en la visualización. Además el uso de Mapas mucho más interactivos para el usuario (Mapbox), proveyendo colores a los trazados y poder distinguir los distintos recorridos almacenados.
4. Linked Stream Data al acoplarse con la plataforma, genera una fuente de datos geo referenciados abierta por medio de dispositivos móviles (uso del sensor GPS), y que estas puedan ser reutilizadas por ambientes afines permitiendo un mayor enlace de datos.
5. Basados en los resultados obtenidos y controles para evitar capturar información repetida, se puede concluir que una vez los usuarios compartan su información, esta podrá ser enlazada y publicada en el dataset, sin realizar una previa limpieza de la data obtenida.

RECOMENDACIONES

Finalizamos con las respectivas recomendaciones para proyectos futuros que permitirán hacer una buena implantación de la plataforma.

1. Para realizar aplicaciones móviles para la recolección de información mediante los sensores de dispositivos, optar por un desarrollo nativo, ya que los framework híbridos aún están surgiendo y no poseen los mecanismos necesarios para la correcta extracción de datos.
2. Al orientarse a Linked Stream Data, procurar seguir las recomendaciones y principios básicos que complementan Linked Data, para evitar confusiones y realizar una correcta publicación de los datos generados, Así mismo intentar hacer que la información sea abierta y no presente restricciones a otros usuarios.
3. Realizar un desarrollo modular y con tecnologías estándar (JSON, Web Sockets, Servicios REST) para la implementación de una plataforma de streaming de datos, permitiendo así que cualquier sensor pueda utilizar los distintos canales o medios de comunicación disponibles, evitando distintos desarrollos por sensor.
4. Al utilizar mapas geo referenciados procurar verificar la libre utilización de los mismos, evitando así posibles problemas legales. Además constatar que el API sea robusta, para una mejor implementación de los distintos requerimientos que puedan proporcionarse.
5. Se puede utilizar la información adquirida, procesada y publicada a través de los dataset para que otras personas o instituciones realicen análisis de la data, y desarrollen aplicaciones para tener una visualización enfocada desde diversos puntos de vista.

BIBLIOGRAFÍA

Aduna. (2009). OpenRdf.Org. Retrieved from http://www.openrdf.org/index

Aguilar, E., & Davila, D. (2013). *Analisis Diseño e implementación de la aplicacion web para el manejo del distributivo de la facultad de ingenieria*. Retrieved from http://dspace.ucuenca.edu.ec/bitstream/123456789/4

Alvarez, S. (2006). Tipos de lenguajes de programación. Retrieved from http://www.desarrolloweb.com/articulos/2358.php

Beckett, D., & Barstow, A. (2001). N-Triples. Retrieved from https://www.w3.org/2001/sw/RDFCore/ntriples/

Berners-Lee, T. (2006). Design Note: Linked Data. Retrieved from http://www.w3.org/DesignIssues/LinkedData.html

Berners-Lee, T. (2009). Putting Government Data online. Retrieved from http://www.w3.org/DesignIssues/GovData.html

Berners-Lee, T., & Connolly, D. (2011). Notation3 (N3): A readable RDF syntax. Retrieved from https://www.w3.org/TeamSubmission/n3/

Berners-Lee, & Tim. (2010). Linked data-connect distributed data across the web. Retrieved from http://linkeddata.org/

Brickley, D., Guha, R. V., & McBride, B. (2014). RDF Schema 1.1. Retrieved from https://www.w3.org/TR/rdf-schema/

Brickley, D., & Libby Miller. (2014). FOAF Vocabulary Specification 0.99. Retrieved from http://xmlns.com/foaf/spec/

Broekstra, Jeen; Kampman, A. (2004). RDF(S) Manipulation, Storage and Querying using Sesame. Retrieved from http://iswc2004.semanticweb.org/demos/03/index.html

Cientifica, P. E. de I. C. y T. (2013). IPHealth. Retrieved from http://www.esp.uem.es/ipHealth/

Consortium, W. W. W. (2011). Linked Open Data Vocabularies. Retrieved from http://lov.okfn.org/dataset/lov/

Consortium, W. W. W. (2017). Guía Breve de Linked Data. Retrieved from http://www.w3c.es/Divulgacion/GuiasBreves/LinkedDa

Cyganiak, Richard; Bizer, C. (2011). Pubby. A Linked Data Frontend for SPARQL Endpoints.

Cyrille, Axel; Ngomo, N. et ál. (2010). A time-efficient approach for large-scale link discovery on the web of data. Retrieved from http://svn.aksw.org/papers/2011/WWW_LIMES/public

Dan Brickley. (2003). W3C Semantic Web Interest Group. Retrieved from https://www.w3.org/2003/01/geo/

Danh, L. P., Minh, D. T., Minh Duc, P., Boncz, P. A., Thomas, E., & Michael, F. (2012). Linked {Stream} {Data} {Processing}: {Facts} {And} {Figures}. *Proceedings of International Semantic Web Conference 2012*, *1380*(24761).

Data, L. (2012). Las tecnologías de Linked Data y sus aplicaciones en el gobierno

electrónico, *1*(1), 49–61.

Dave Beckett. (2014). RDF 1.1 XML Syntax. Retrieved from https://www.w3.org/TR/rdf-syntax-grammar/

Devsaran. (2016). EUCLID. Retrieved from http://www.euclid-project.eu/

Erling, O. (2010). Linked Data and Virtuoso.

Foundation, Knowledge, O. (2014). In LOV at a glance. Retrieved from http://lov.okfn.org/dataset/lov/about

Foundation, O. K. (2011). Ambar Linked Open Data Portal. Retrieved from http://ambar.utpl.edu.ec/

Frank Manola, E. M. (2004). REC RDF Primer. Retrieved from https://www.w3.org/TR/2004/REC-rdf-primer-20040210

Gayo, J. E. L. (2011). Towards an architecture and adoption process for Linked Data technologies in Open Government contexts - A case study for the Librery of Congress of ChileNo Title. Retrieved from https://www.slideshare.net/jelabra/towards-an-arch

Graham, K., & Carroll, J. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. Retrieved from http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

Group, S. or. C. (2015). Schema.org. Retrieved from https://schema.org/

Guillermo, E., Riera, A., Alfredo, D., & Garzón, D. (2013). Análisis, Diseño e Implementación de la Aplicación Web para el manejo del Distributivo de la Facultad de Ingeniería.

Gutiérrez, J. J. (2014). ?` Qué es un framework Web? *Available in: Http://www. Lsi. Us. Es/~ javierj/investigacion_ficheros/Framework. Pdf Accessed May*, *12*, 1–4.

Hazaël-Massieux, D. (2016). JAVASCRIPT WEB APIS. Retrieved from https://www.w3.org/standards/webdesign/script

Heath, Tom; Bizer, C. (2011). Linked Data: Evolving the Web into a Global Data Space, Morgan & Claypool.

Heatj, T., & Bizer, C. (2011). *Linked Data: Evolving the Web into a Global Data Space, tomo 1.*

Hendler, J. et ál. (2011). TWC LOGD: A Portal for Linked Open Government Data Ecosystems. Retrieved from https://tw.rpi.edu//

Iannella, R., & James McKinney. (2014). vCard Ontology - for describing People and Organizations. Retrieved from https://www.w3.org/TR/vcard-rdf/

Jaenicke, N. (2009). Triplify. Retrieved from http://triplify.org/Overview

Kioskea. (2014). El protocolo HTTP. Retrieved from http://es.ccm.net/contents/264-el-protocolo-http

Lee, T. B., Roy, F., & Larry, M. (2005). *Uniform Resource Identifier (URI): Generic Syntax. Request For Comments (RFC)*.

Manola Frank, Miller Eric, M. B. (2014). Primer RDF. Retrieved from https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140

Mathieu d'Aquin, Kessler, C., & Kauppinen, T. (2014). Linked Universities. Retrieved

from http://linkeduniversities.org/index.html

Muster's, U. de. (2013). LODUM. Retrieved from http://lodum.de/

Nacional, B. del C. (2014). Linked Open Data. Retrieved from http://datos.bcn.cl/es/informacion/que-es

Navarra, U. de. (2014). IT Services. Retrieved from http://www.unav.edu/web/it/guias-manuales

Rafael Veiga Cid. (2015). Node.js y Websockets. Retrieved from https://manuais.iessanclemente.net/index.php/Node.

Seaborne, E. P. y A. (2008). SPARQL Query Language for RDF. Retrieved from https://www.w3.org/TR/rdf-sparql-query/

Sequeda, J., & Corcho, O. (2009). Linked Stream Data: A Position Paper. *Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN09) at ISWC 2009*, *522*, 148–157. Retrieved from http://ceur-ws.org/Vol-522/p13.pdf

Sheridan, J. (2010). Legislation.gov.uk. Retrieved from http://blog.law/cornell.edu/voxpop/tag/legal-linked-data/

Tim Berners-Lee, James Hendler, O. L. (2001). The Semantic Web. *Scientific American*.

University, T. O. (2017). Linked Data from The Open University. Retrieved from http://data.open.ac.uk/

W3C. (2007). Guía Breve de Web Semántica. Retrieved from http://www.w3c.es/Divulgacion/GuiasBreves/WebSeman