Trabajo de Fin de Grado

Publicación de datos sociosanitarios: Una API basada en Open Linked Data

Health Data Publish: an API based Open Linked Data

Salomé González Rodríguez

La Laguna, 7 de Julio de 2015

D. **Jose Luis Roda García**, con N.I.F. 43356123-L profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y Sistemas de la Universidad de La Laguna, como tutor.

D. **Laura María Gutiérrez Medina**, con N.I.F. 78635632-N miembro del grupo Taro de la Universidad de La Laguna, como cotutora.

C E R T I F I C A (N)

Que la presente memoria titulada:

"Publicación de datos sociosanitarios: Una API basada en Open Linked Data."

ha sido realizada bajo su dirección por D. **Salomé González Rodríguez**, con N.I.F. 78633492-B.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 7 de julio de 2015.

Agradecimientos

Este proyecto de final de grado se lo quiero dedicar sobre todo a mis padres que han estado en todo momento apoyándome y motivándome para llegar hasta donde he llegado.

También quiero agradecer a mi tutor José Luis Roda por guiarme en la realización del proyecto y enseñarme los conceptos de Linked Open Data.

Además, gracias a la ayuda y el asesoramiento de mi coturora Laura María Gutierrez Medina he podido realizar la implemetación de la API REST sin problemas.

Finalmente, agradecer a Luis Alberto Julio Rodríguez y Adrián Muñoz Barrera por hacer la interfaz de la aplicación web.

Licencia

Resumen

En este proyecto se ha desarrollado una API REST que permite consultar datos de múltiples fuentes en el ámbito de los centros sociosanitarios, para que pueda ser consumida por cualquier dispositivo o aplicación. Para ello, se han utilizado las tecnologías de Web Semántica, Linked Data y Open Data. También, se ha creado una aplicación web que es capaz de realizar llamadas a los servicios implementados en la API REST. De esta manera, cualquier usuario a través de su móvil, tablet u ordenador podrá ver en una interfaz sencilla y usable, los resultados obtenidos tras interactuar con ella.

El principal objetivo es facilitar a los usuarios la geolocalización e información más relevante de los centros sociosanitarios de Tenerife en un mismo punto de acceso. Por un lado, como fuentes principales se utilizan los datos obtenidos de las administraciones públicas (IASS y Servicio Canario de Salud): farmacias, botiquines, centros de discapacitados, centros de mayores y otros centros sociosanitarios. Por otro lado, la información adicional se ha obtenido de una fuente externa a través del endpoint de DBpedia. Finalmente, para la implementación de la solución propuesta se han utilizado las siguientes tecnologías: Java, Maven, Jersey, jQuery, AJAX, Apache Jena, Apache Tomcat entre otros.

Palabras clave: *Web Semántica, Linked Data, Open Data, Centros Sociosanitarios, datos, URI, RDF, SPARQL, Ontología, API REST, buscar información.*

Abstract

In this project has been developed an API REST that allow us to get information from different health centers, with the goal that it can be used for any device or application. For that, has been used the technologies of Semantic Web, Linked Data and Open Data. Also, a web application has been created that allow the user for call the different services implemented in the API-REST. This way, anyone can use any device (smartphone, tablet or computer) can see in easy and usable interface, the different results returned after interacting with it.

Besides, the goal of this application is to provide the users with geolocation data and relevant information about the health centers of Tenerife on a single access point. On the one hand, how main sources we used the data obtained from public administrations (IASS and "Servicio Canario de Salud"): pharmacy, medicine cabinets, disabled centers, senior centers, and other health centers. On the other hand, additional information has been obtained from external sources such as DBpedia. Finally, for the implementation the following technologies have been used: Java, Maven, Jersey, jQuery, AJAX, Apache Jena, Apache Tomcat and others.

***Keywords**: Semantic Web, Linked Data, Open Data, Health center, data, URI, RDF, SPARQL, Ontology, API REST, search information.*

Índice General

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

En la actualidad, para la sociedad la Web es una herramienta de uso diario puesto que podemos acceder a ella desde cualquier dispositivo de manera muy intuitiva dado que los diseños se hacen pensando en la usabilidad, incluso las personas con discapacidades pueden acceder gracias a que muchos sitios web añaden accesibilidad a los mismos. Por ello, la Web ha cambiado la forma en la que los usuarios buscan información, se comunican entre ellos, realizan su trabajo, disfrutan de su tiempo libre, etc.

A medida que pasan los años, la información que se encuentra en la Web va aumentando exponencialmente gracias a que más usuarios van accediendo e incorporando nuevos datos. Por ello, realizar una búsqueda con éxito es cada vez menos efectiva, ya que muchas veces los resultados obtenidos no son los deseados, haciendo que el usuario tenga más difícil la tarea de buscar lo que realmente necesita. Para solventar este problema, se han desarrollado un conjunto de estándares en relación a la Web Semántica. Entre ellos, se encuentran las tecnologías RDF, OWL y SPARQL y los conceptos de Open Data y Linked Data.

Con la introducción de las tecnologías Open Data y Linked Data para la Web Semántica, los datos pueden estar interconectados a través de las URIs y además, se puede acceder a la información que publican las administraciones públicas y otras entidades, respetando la confidencialidad de los mismos. De esta manera, los datos abiertos y enlazados facilitan las contribuciones externas por lo que aumenta la calidad y eficiencia de los mismos. Desgraciadamente, aún son muy pocas las administraciones que publican datos en formatos enlazados y, menos aún, las que disponen de servicios de consulta de datos enlazados.

En el trabajo que presentamos se muestra cómo se podrían aplicar las estrategias de Open Data y Linked Data para el ámbito sociosanitario. Tras un estudio de los centros sociosanitarios se ha detectado que no existe una API relacionada con este ámbito. Por ello, siguiendo el concepto de Linked Data, se ha desarrollado una API REST que permite consultar datos del ámbito de los centros sociosanitarios para que pueda ser consumida por cualquier dispositivo y aplicación que entienda HTTP [88]. Los datos públicos ofrecidos mediante la API REST se han obtenido de tres fuentes diferentes.

Por un lado, el IASS ha publicado datos socio sanitarios en el portal Open Data Canarias (datos disponibles en open data). Por otro lado, el Servicio Canario de Salud ofrece en su web, información de sus centros (datos obtenidos directamente de su web). Y por último, de DBpedia como fuente externa (datos obtenidos de una fuente Linked Data).

Con el prototipo presentado demostramos la enorme potencia que tendrá la web de los datos.

Capítulo 2.
Conceptos básicos

En el capítulo anterior se expone **brevemente la API desarrollada**, por lo que en este capítulo se presentarán de manera ordenada los conceptos básicos y las tecnologías necesarias para poder entender en profundidad el proyecto.

2.1 Web Semántica (Semantic Web)

La Web (World Wide Web) desde su aparición se ha convertido en un instrumento de uso cotidiano, por esa razón, ha cambiado profundamente la forma en la que la sociedad se comunica, hace relaciones comerciales, se entretiene, disfruta de su tiempo libre, difunde cultura, se instruye (accediendo a la información) y realiza su trabajo.

Desde su comienzo, la Web ha experimentado un crecimiento espectacular y ha ido evolucionando tanto en tecnologías (HTML, HTTP, CSS, ASP, PHP, Java, Ajax, etc.) como en el uso y significado de la misma.

En primer lugar, la Web 1.0 se basaba en la idea de consultar información de manera unidireccional y su principal elemento eran las páginas estáticas.

En segundo lugar, la Web 2.0 trata de centrarse en la participación y colaboración de los usuarios de manera que puedan producir contenido y a la vez consumirlo a través de redes sociales, cloud computing, blogs, etc.

En último lugar, la Web 3.0 o denominada la web semántica o web inteligente [2], promueve que la información que se encuentra en la web pueda procesarse de manera automática. Por ejemplo, los usuarios delegan las tareas en el software mediante agentes inteligentes y éste es capaz de procesar el contenido, razonar de manera lógica y combinarlo sin operadores humanos.

En otras palabras, se trata de un software conjunto de tecnologías y estándares que son capaces de resolver problemas cotidianos que están bien definidos, a través de operaciones especificadas que se llevarán a cabo sobre datos (metadatos) existentes y que tienen un formato estándar.

Asimismo, la Web Semántica se basa en enlazar datos o conceptos en lugar de documentos que es lo que ocurre en la Web 2.0. Tim Berner Lee creador de la World Wide Web en 1998 y director del W3C pretende que los datos enlazados puedan ser consultados como si de una gran base de datos global se tratase [22].

Por tanto, la web semántica ayuda a resolver problemas de sobrecarga de información y heterogeneidad de las fuentes de información. De esta manera, la infraestructura de la web semántica al estar basada en metadatos (información de los datos) aporta una extensión a las capacidades de la misma, pudiendo los datos ser interpretados tanto por agentes humanos como por agentes computarizados.

Para obtener una adecuada definición de los datos, la web semántica utiliza los estándares OWL, RDF y SPARQL entre otros mecanismos para convertirla en una gran infraestructura global, en la cual es posible compartir y reutilizar los datos. Cada uno de estos términos los veremos en la presente memoria.

La Web Semántica usa los Datos Enlazados (Linked Data) para vincular los diferentes datos que están en la web. Además, entre los diferentes datos que hay, se encuentran los Datos Abiertos (Open Data) obteniéndose la mayor parte de ellos de entidades públicas.

2.2 Datos Abiertos (Open Data)

Se trata de una iniciativa que persigue el objetivo de poner a disposición de la sociedad datos científicos, estadísticos y los datos que gestiona la administración pública en formatos de fácil manipulación. Es decir, es una estrategia que busca la manera de que los datos estén libres y accesibles para

todo el mundo de manera que no tenga permisos específicos y así puedan ser reutilizados.

Para que se consideren datos abiertos, un grupo de trabajo denominado "Open Government Working Group" en Diciembre de 2007 [11] resumieron los ocho principios básicos de la apertura de datos:

1. **Completos:** todos los datos deben estar disponibles y accesibles y, además, no pueden estar sujetos a limitaciones de privacidad, seguridad o propiedad.
2. **Primarios:** los datos obtenidos deben estar con un gran nivel de detalle y no deben de modificarse o añadir nuevo contenido.
3. **Actualizados:** deben estar a disposición de la sociedad lo más rápido posible para preservar su valor.
4. **Disponibilidad:** los datos deben estar disponibles tanto para todos los usuarios como para todos los propósitos.
5. **Automatizados:** los datos no pueden estar procesados y deben estar estructurados para favorecer el procesamiento por parte de las máquinas.
6. **No discriminatorios:** deben estar disponibles para todos los usuarios sin necesidad de un registro o licitación previa.
7. **No propietarios:** los datos deben estar en formato libre (xml, csv, rdf, ...), es decir ninguna entidad puede tener el control exclusivo de los mismos.
8. **Licencia Libre**: no pueden estar sujetos a derechos de autor, patente, marca comercial o regulación de secretos comerciales, a menos que haya razones de privacidad, seguridad o derechos.

A la colección de datos abiertos que están organizados para que sean tabulados, indexados y localizados se denominan datasets (conjunto de datos). Para los datasets se ha definido una clasificación de los diferentes tipos de formatos que pueden tener:

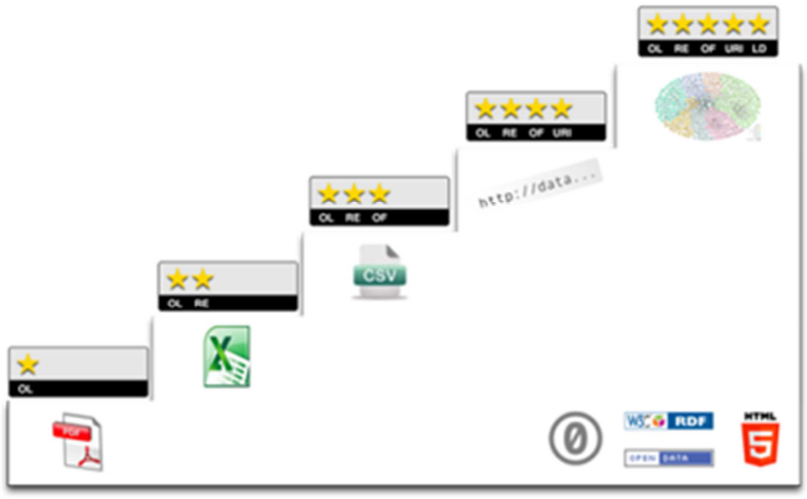

Figura 2.1. Esquema de desarrollo de 5 estrellas para Datos Abiertos

- Una estrella: Disponible en internet bajo cualquier formato y con una licencia abierta.
- Dos estrellas: Disponible en internet de manera estructurada (formato XLS).
- Tres estrellas: Disponible en internet de manera estructurada y en formato no propietario (CSV).
- Cuatro estrellas: Disponible en internet de manera estructurada, en formato no propietario y además deben de seguir los estándares establecidos por W3C, es decir usar URIs para denotar cosas y así los usuarios puedan apuntar a esos datos (RDF).
- Cinco estrellas: Contiene la estrella anterior pero además se deben de vincular los datos con los de otros usuarios y de esta manera proveer contexto (Linked Data).

Por una parte, se puede obtener grandes beneficios si se tiene los datos en formato de cinco estrellas. Por ejemplo, pueden ser enlazados desde cualquier otro sitio web y combinarlos con otros y así, descubrir más datos. Además, se pueden reutilizar los datos, las herramientas e incluso las librerías disponibles.

Por otra parte, los beneficios que aporta como publicador y desarrollador son los siguientes: poder optimizar el acceso a los datos, hacer que tus datos sean descubiertos por otros publicadores y éstos puedan enlazarse a tus datos y, además, incrementar el valor de los mismos.

En conclusión, liberar los datos facilita la creación de nuevas aplicaciones, ya que se acelera la tasa de descubrimiento con mejores accesos a la información. Además, los sistemas abiertos hacen sencillas las contribuciones externas por lo que se aumenta en eficiencia y en calidad en los mismos.

2.3 Datos Enlazados (Linked Data)

En los últimos años se ha producido un aumento en la publicación de datos abiertos del sector público. Por ello, se ha experimentado una mayor cantidad de datos enlazados y publicados en la Web Semántica sobre este sector. Además, también se está extendiendo a otros sectores como los medios de comunicación, infraestructuras y logística, el ámbito universitario y científico y los datos geográficos. Algunos ejemplos son los portales de Aragón, Nature, BBC, etc.

Linked Data [24] trata de interconectar los distintos datos estructurados que están distribuidos en la web por medio de la web semántica para que puedan ser más útiles, ya que pueden ser interpretados por el software a través de agentes inteligentes y aplicaciones. Es decir, permite construir una gran base de datos enlazados y compartidos en la Web.

La idea de los datos enlazados se basa en la manera en la que se pueden mostrar, intercambiar y conectar los datos a través de URIs. Por tanto, para poder conseguir que los datos estén interconectados y poder reutilizar la información se han de seguir los siguientes cuatro principios básicos que ayudarán a fomentar el crecimiento de la Web y además ofrecerá un valor añadido:

1. Usar URIs para identificar los recursos y así poder evitar ambigüedades y ofrecer una forma estándar y única para referirse a cualquier recurso.
2. Usar URIs HTTP para asegurar que cualquier recurso pueda ser consultado y accedido a través de la Web.
3. Ofrecer información útil sobre los recursos usando el modelo de datos para metadatos RDF. Además, para poder consultar los datos establecidos en el grafo RDF se debe utilizar la tecnología SPARQL.

4. Incluir enlaces a otras URIs para poder incrementar la conectividad de los datos que están en sitios Web de forma que no se queden aislados y se puedan compartir con otras fuentes externas y, además que otros sitios Web puedan enlazar nuestros propios datos. Es decir, permite la recuperación y agregación de información relacionada.

2.4 Marco de Descripción de Recursos (RDF)

Marco de Descripción de Recursos o RDF [30] es un modelo estándar para el intercambio de datos en la Web. Se trata de un lenguaje para especificar metadatos de manera estructurada.

Uno de los aspectos fundamentales de RDF es que la información puede ser tratada por aplicaciones que intercambian los datos y, de esta manera puedan ser procesados tanto por máquinas como por seres humanos.

Asimismo, RDF es un método normalizado por W3C que se basa en la idea de hacer declaraciones sobre los recursos web en forma de tripletas, es decir sujeto-predicado-objeto. En otras palabras, RDF crea grafos dirigidos y etiquetados para representar la información en la Web.

Por un lado, el sujeto, predicado y objeto pueden ser URIs y, de esta manera se puede hacer el enlazado con otros recursos (Linked Data). Finalmente, los nodos objeto pueden ser literales (texto sin formato).

De esta manera, una tripleta se representa mediante nodos que están conectados por flechas. Los nodos sujeto y objeto representan los recursos y el predicado (las flechas) denotan los rasgos o aspectos del mismo.

Figura 2.2. Esquema de desarrollo de 5 estrellas para Datos Abiertos.

En la siguiente imagen se muestra un ejemplo muy simple de una tripleta en RDF. “Juan es Científico”, el sujeto es “Juan”, el predicado es “es” y, finalmente, el objeto es “científico”.

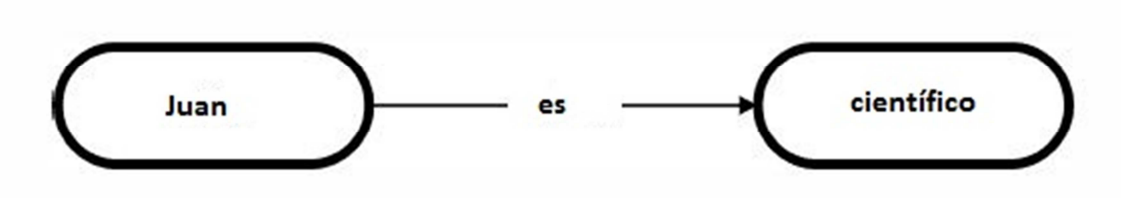

Figura 2.3. Ejemplo de tripleta RDF.

Existen diversos tipos de formatos RDF entre los que se encuentran los siguientes:

- RDF/XML: sintaxis definida por W3C para expresar un grafo RDF como un documento XML.
- N3 o Notation3: diseñado pensando en la legibilidad por parte de humanos. Además, es un superconjunto de Turtle.
- Turtle: solo puede serializar grafos RDF válidos. Se trata de un subconjunto de N3 y un superconjunto de N-Triples.
- N-Triples: diseñado por W3C para ser un formato más simple que N3 y Turtle. Además, es un subconjunto de Turtle.

A continuación, se muestra un ejemplo expresado en RDF/XML:

```
1: <?xml versión=”1.0” ?>
2: <rdf:RDF
3:          xmlns:rdf=”http://www.w3.org/1999/02/22-rdf-syntax-ns#”
4:          xmlns:dc=”http://purl.org/dc/elements/1.1/”>
5: <rdf:Description rdf:about=”https://en.wikipedia.org/wiki/Barack_Obama”>
6:     <dc:title>Barack Obama</dc:title>
7: </rdf:Descripion>
8: </rdf:RDF>
```

Para comenzar, se establece la etiqueta xml donde se indica que va a tratarse de un documento XML. En la siguiente línea, se indica que el documento además será un modelo RDF. En la tercera y cuarta línea, se especifican mediante el espacio de nombres *xmlns* los prefijos *rdf* y *dc* para hacer referencia a las URIs correspondientes. En la quinta línea, se declara el comienzo de la descripción de un recurso y además, mediante *rdf:about* se indica que recurso mediante URI se va a describir. En la sexta línea, se

declara el predicado mediante *dc:title* el cual es el literal Barack Obama. En las siguientes líneas, se cierran las etiquetas de descripción de recurso y de rdf.

2.5 Ontologías

Una ontología [33] hace referencia a una jerarquía de conceptos con atributos y la relación que tiene entre esos conceptos. Además, proporciona un vocabulario de clases y relaciones para definir un dominio con la finalidad de facilitar la comunicación y el intercambio de información entre aplicaciones y entidades.

Las ontologías son usadas para capturar conocimiento sobre los dominios de interés. Además, debe de proveer un buen entendimiento del dominio al que representa. Es decir, describe los conceptos del dominio y las relaciones que tienen entre ellos.

En la actualidad, solo los humanos son capaces de extraer información en la web 2.0, puesto que no existe una semántica en la cual los agentes inteligentes, aplicaciones web o apps puedan reconocer dicha información. Por ello, por medio de las ontologías en la web semántica las máquinas son capaces de extraer la información y de razonarla para la resolución de problemas.

A continuación, se muestran los lenguajes de ontologías más destacados:

- RDFS [35]
- OWL [38]

2.5.1 RDF Schema (RDFS)

RDF Schema [35] proporciona un vocabulario de modelado de datos para los datos RDF, es decir, es una extensión semántica de RDF. Proporciona mecanismos para la descripción de grupos de recursos y las relaciones entre esos recursos, usando propiedades y valores.

De esta manera, provee de mecanismos para especificar que las clases, propiedades y sus relaciones son parte de un vocabulario. Por una parte, las clases pueden representar cualquier categoría de cosas y se describen por recursos *rdfs:Class* y *rdf:Resource*. Por otra parte, las propiedades se describen usando la clase *rdfs:Property*.

2.5.2 Lenguaje de Ontologías Web (OWL)

OWL [39] se trata de un lenguaje de marcado recomendado por W3C que está diseñado para representar el contenido de la web de manera que es interpretable por agentes inteligentes, aplicaciones web, apps, etc. Es decir, se utiliza OWL cuando se requiere que la información sea procesada por el software y no para representarla.

En la actualidad, OWL posee sub-lenguajes que aumentan la expresión del mismo, ya que incorporan diferentes funcionalidades: OWL Lite, OWL DL y OWL Full.

Al igual que el lenguaje RDF, OWL emplea las tripletas de RDF pero el mecanismo de OWL es mejor, debido a que permite interpretar mejor los contenidos de la web, puesto que proporciona un mecanismo adicional además de una semántica formal. En conclusión, OWL es un lenguaje que tiene más poder expresivo que RDF y XML.

2.6 SPARQL

SPARQL [47] es un lenguaje de consulta estandarizado y normalizado por W3C para acceder a la información que contiene un grafo dirigido RDF.

La consulta que se establece con el lenguaje SPARQL devuelve la información en forma de URIs, literales o un grafo RDF.

2.6.1 SPARQL Endpoint

Permite a los usuarios (humanos o máquinas) consultar bases de conocimientos [46] mediante el lenguaje SPARQL. Además, los resultados obtenidos después de ejecutar la consulta, normalmente son en formato procesable por máquinas.

En conclusión, SPARQL Endpoint es un servicio para que los humanos y las máquinas puedan realizar consultas y se pueda presentar el resultado de manera legible para ambos. En este proyecto se ha utilizado el endpoint de DBpedia [89] pero existen diversos endpoint como el de Open Data Canarias [90].

2.6.2 Estructura de consultas SPARQL

A continuación, se presenta la estructura que pueden tomar las consultas SPARQL.

```
#Directiva base
BASE <URI>
#Declaraciones de los prefijos
PREFIX pref: <URI>
#Definición del conjunto de datos
FROM . . .
#Resultado de la cláusula
SELECT . . .
#Patrón de consulta
WHERE {
        OPTIONAL { . . .  }
        FILTER ( . . . )
}
#Modificadores de la consulta
ORDER BY . . .
```

Para comenzar, la estructura empieza con dos partes opcionales que sirven para abreviar URIs, una directiva BASE y la definición de uno o varios prefijos. A continuación, se definen los conjuntos de datos en los cuales se va a realizar la consulta, a través de un grafo RDF. Además, la cláusula FROM

también es opcional, por lo que si se omite se toma por defecto el conjunto de datos que se haya creado previamente a la consulta.

Posteriormente, mediante la cláusula SELECT se especifica que información (variables y valores) debe devolver como resultado la consulta, si se ejecuta con éxito.
La cláusula WHERE especifica el patrón de la consulta, es decir, establece que conjunto de datos se va a obtener a partir de las indicaciones establecidas dentro de esta cláusula. Esta cláusula no es opcional pero la palabra WHERE sí lo es, aunque se recomienda por claridad no omitirla.

La palabra clave OPTIONAL sirve para especificar los patrones de la consulta que pueden hacer que la misma falle si no se encuentran. Se trata de una cláusula opcional.

Cuando se quiere añadir restricciones para filtrar la solución se utiliza la palabra reservada opcional FILTER. Esta palabra reservada utiliza condiciones que devuelven verdadero o falso para filtrar los resultados de la consulta deseados. Se pueden utilizar los operadores lógicos && y ||.

Para finalizar, se definen de manera opcional los modificadores de la consulta que proporcionan la manera de organizar los resultados de la misma. Además, esta cláusula es opcional. La cláusula ORDER BY ordena de manera ascendente o descendente los resultados de la consulta. Otros ejemplos de modificadores son la cláusula LIMIT que limita la cantidad de resultados a mostrar y DISCTINCT que elimina las soluciones duplicadas.

I.2.6.2.1 Formas de consultas

- **Cláusula SELECT**

Como se mencionó anteriormente, SELECT devuelve un conjunto de valores en forma de tabla en concordancia con el patrón establecido de búsqueda.

- **Cláusula DESCRIBE**

Cuando no se conoce el grafo RDF se utiliza la cláusula DESCRIBE para pedir al procesador de consultas SPARQL que describa el recurso. Dicha consulta devuelve un grafo RDF con los recursos encontrados.

- **Cláusula CONSTRUCT**

La cláusula CONSTRUCT devuelve un nuevo grafo RDF. Se trata de una alternativa a la cláusula SELECT. Es decir, en lugar de devolver una tabla de valores con los resultados obtenidos, devuelve un grafo RDF.

- **Cláusula ASK**

La cláusula ASK devuelve verdadero o falso dependiendo de si se cumple el patrón de la consulta y encuentra alguna coincidencia en el conjunto de datos. Es decir, según la pregunta realizada devuelve la respuesta en forma de booleano. Por ejemplo: ¿Es el río Amazonas más largo que el río Nilo?

```
PREFIX prop: <http://dbpedia.org/property/>
ASK{
        <http://dbpedia.org/resource/Amazon_River> prop:length ?amazon .
        <http://dbpedia.org/resource/Nile> prop:length ?nile .
}
```

2.6.3 SPARQL 1.1

El lenguaje de consultas SPARQL descrito anteriormente tiene bastantes limitaciones por lo que W3C ha creado la versión 1.1 con más opciones, puesto que la versión anterior solo podía hacer operaciones de lectura. Esta versión incluye las operaciones:

- Actualizar, insertar y borrar.
- Negación.
- Funciones agregadas.
- Sub-consultas.

En este proyecto, las operaciones realizadas para realizar las consultas han sido relativamente sencillas y no ha sido necesario introducir nuevas funcionalidades de SPARQL 1.1.

Capítulo 3.
Caso a estudio: Publicación de Datos de los Centros Sociosanitarios

Una vez se ha interiorizado los conceptos básicos para la realización de nuestra API, en este capítulo se pretende explicar la motivación, los objetivos y el alcance de la misma y, además, la problemática que se ha tenido para poder llevarla a cabo.

3.1 Antecedentes y Motivación

Cualquier usuario en algún momento tiene la necesidad de buscar de manera rápida y sencilla los centros sociosanitarios más cercanos a su ubicación, puesto que estos forman parte de una de las necesidades básicas, la salud.

Por una parte, el sector sociosanitario abarca un conjunto de servicios para asistir a las personas que se encuentran en situación de dependencia como la tercera edad, los enfermos crónicos y las personas con alguna discapacidad física, psíquica o sensorial.

Por otra parte, cualquier usuario se puede ver beneficiado por este sector, ya que ofrece prestaciones como pueden ser clínicas dentales, farmacias, botiquines, centros ópticos, fisioterapeutas, etc.

Hasta el momento, las búsquedas en la web de centros sociosanitarios se hacen demasiado tediosas cuando no sabes realmente lo que debes buscar y cómo debes buscarlo y, además, la información de cada recurso se encuentra muy dispersa. De esta manera, enlazando unos datos con otros (usando la estrategia Linked Data) los usuarios pueden realizar de manera sencilla,

búsquedas y, como resultado, obtener toda la información más importante al instante. Además, ocurre que en la mayoría de búsquedas en la web no se obtienen los resultados que se desean.

Un ejemplo muy común es buscar farmacias y centros, cerca de nuestro domicilio, donde puedan tratar a nuestras personas de la tercera edad más allegadas, para que el acceso a cada uno de los servicios sea muy rápido. Otro ejemplo frecuente, sería buscar información relativa a los centros más cercanos a un lugar.

Por ello, la creación y desarrollo de esta API REST surge como necesidad de facilitar la geolocalización y búsqueda de información más relevante en tiempo real, en mismo punto de acceso y visualización, acerca de los centros sociosanitarios en Tenerife en función de la ubicación del usuario o de alguna especificada por el mismo.

Para ello, es necesario aplicar tecnologías como Linked Data para enlazar unos datos con otros en un mismo punto de acceso y éstos sean más útiles, es decir enlazar datos de DBpedia como fuente externa, con los datos locales obtenidos desde el IASS y desde el Servicio Canario de Salud.

3.2 Objetivos

En este apartado se exponen los objetivos tanto generales como específicos que abarca el proyecto.

3.2.1 Objetivo General

Desarrollar una API REST que permita a cualquier usuario acceder a la información más relevante en tiempo real, en un único punto, acerca de los centros sociosanitarios geolocalizados en Tenerife. Además, podrá acceder a la información de otros centros que se encuentren cercanos a la búsqueda principal. Una vez obtenida la información de los centros que buscamos, el sistema debe ofrecer información de utilidad al usuario dsiponible en Internet.

3.2.2 Objetivos Específicos

Por un lado, se ha de desarrollar una API REST con la finalidad de tener un único punto de acceso a los datos en tiempo real y que cualquier aplicación puede consumirla realizando las correspondientes llamadas a los servicios implementados a través de las correspondientes URLs. Estos servicios son búsquedas de centros sociosanitarios según los parámetros establecidos:

- Buscar por el nombre del centro y devuelve el nombre, URI, latitud y longitud del mismo.
- Realizar una búsqueda por URI del centro y debe devolver el nombre, dirección, municipio, código postal, teléfono, fax, página web, latitud y longitud.
- También a través de la URI del centro y el radio en metros, obtener los nombres de los centros más cercanos en ese radio con sus respectivas URIS.
- Cuando se utiliza el servicio de DBpedia a través de una URI del centro, éste debe devolver las camas, los centros afiliados y la descripción del mismo.
- Al consumir el servicio de buscar centros categorizados según las especialidades, devuelve cada uno de ellos con el nombre y la URI correspondiente.

Por otro lado, se ha realizado una interfaz web que consume la API REST descrita anteriormente y se puede acceder a ella mediante cualquier dispositivo que disponga de un navegador. Esta aplicación web muestra en un mapa todos los recursos geolocalizados para que el usuario pueda elegir en el mapa la información del recurso seleccionado mediante un icono identificativo e intuitivo. Al entrar en la aplicación, por defecto aparecerán todos los recursos en el mapa (Farmacias, Botiquines, Centros para Discapacitados, Centros de Mayores y otros Centros Sociosanitarios), pero mediante la selección de los mismos que se encuentran categorizados se puede acotar la búsqueda a los que sean requeridos.

La aplicación debe ser capaz de mostrar la información más relevante de un centro sociosanitario al seleccionar el icono identificativo en el mapa. Esta información se mostrará en forma de tablas:

- Información general: Contiene los datos obtenidos de las administraciones públicas (IASS y Servicio Canario de Salud) existentes en el portal Open Data Canarias en el caso del IASS.
- Información adicional: Los datos se extraen de una fuente externa (DBpedia) y, de esta manera, se hará el enlazado de estos datos con los descritos anteriormente.
- Lugares cercanos en un radio de 500 metros: Muestra los centros más cercanos al recurso seleccionado. De esta manera, el usuario puede hacerse una idea de lo que se encuentra a su alrededor, en lugar de minimizar la información del recurso para poder visualizar los centros más cercanos. Es decir, se agiliza el proceso de búsquedas de recursos colindantes.

Finalmente, mediante los buscadores, se podrá buscar por el nombre del recurso y éste se mostrará en el mapa geolocalizado con su respectivo icono quedando el resto de recursos ocultos y además, se podrá acceder a cualquier punto del mapa poniendo la ubicación deseada.

3.3 Alcance

El alcance de este proyecto abarca la implementación de una API REST y de una aplicación web en la cual haga llamadas a los servicios de la API REST y presente los resultados en una interfaz con un diseño usable.

Por una parte, se desarrolla una API REST que puede ser consumida por cualquier aplicación que entienda de HTTP. Los servicios implementados harán posible obtener la información y geolocalización los centros sociosanitarios en una ubicación preestablecida y en las cercanías de los mismos. Además, se podrán realizar búsquedas de centros según las especialidades de los mismos.

Por otra parte, se desarrollará una aplicación web que consuma la API anterior a través de los servicios implementados en la API. Los resultados se mostrarán en una interfaz en la cual el usuario pueda interactuar con ella, de

manera sencilla, y con los datos obtenidos de las consultas realizadas a la API.

3.4 Problemática

En la realización del proyecto se han encontrado diversos problemas que se han ido solventando en la medida de lo posible. A continuación, se exponen los dos problemas más significativos:

Por un lado, cuando se comenzó a hacer el análisis de la API REST a desarrollar, se partió de la idea de realizar una aplicación enfocada a la búsqueda de información en un mismo punto de acceso para el beneficio social, obteniendo datos de ONGs y centros sociosanitarios. Los datos proporcionados por la administración pública IASS de los centros sociosanitarios estaba en formato XML. Por el contrario, la información obtenida de ONG no estaba estructurada ni categorizada y, además, el formato era PDF, por lo que la fase del tratamiento de datos era muy complicada. Por esto, se tomó la decisión de hacerlo solo de centros sociosanitarios.

Por otro lado, encontrar información de fuentes externas para poder realizar el enlazado con otros datos ha sido muy difícil, puesto que solo se ha encontrado información en DBpedia que puede ser de interés para el usuario. Incluso la información obtenida a través del Endpoint [46] de DBpedia es muy escasa y solo es posible enlazar dos recursos. Por consiguiente, las administraciones públicas deberían exponer todos los datos posibles en portales (como Open Data Canarias, DBpedia, ...) para que el valor de sus datos se incrementase pudiendo hacer Linked Data con otros datos externos.

Capítulo 4.
Desarrollo del proyecto

En el capítulo anterior se han detallados los objetivos y el alcance de esta API, por lo que en este capítulo se pretende explicar los pasos seguidos para la realización e implementación de la misma. Además, se ha expuesto el diseño técnico y las tecnologías usadas para la implementación de la aplicación web.

4.1 Metodología

Aunque existen diversos procedimientos para establecer los pasos a seguir en la tecnología Linked Data, se ha creado una metodología simplificada de otras bien conocidas como GLD Life Cycle [98] y Pila LOD2 [99] dado que se trata de un proyecto pequeño. Por consiguiente, el conjunto de procedimientos seguidos para la **creación de sistemas basados en Linked Data** han sido los siguientes:

4.1.1 Dominio de centros sociosanitarios

Al comenzar el proyecto se estableció el dominio sobre el cual se va realizar el desarrollo de la API REST. Dado que el ámbito de los centros sociosanitarios podría ser de bastante utilidad para cualquier ciudadano, se buscó información relacionada con el mismo. Además, se buscó páginas web relacionadas, como el Servicio Canario de Salud[96], el IASS [97] entre otros.

4.1.2 Conjuntos de datos

A continuación, se obtienen los conjuntos de datos necesarios para obtener la información a presentar en la aplicación.

Por un lado, a través del portal Open Data Canarias se obtuvieron los datos del Instituto Insular de Atención Social y Sociosanitaria (IASS) que estaban en formato XML.

Por otro lado, mediante la tecnología Web scraping [91] se han extraído más datos relevantes de la página del Servicio Canario de Salud para la API REST que se pretende crear.

Finalmente, los conjuntos de datos que se han obtenido de las administraciones públicas son los siguientes:

- Farmacias.
- Botiquines.
- Centros de discapacitados.
- Centros de mayores.
- Conjuntos de centros de varios tipos (clínicas dentales, farmacias, botiquines, centros ópticos, fisioterapeutas, etc.)

Finalmente, este proyecto puede ser de utilidad para las administraciones para que puedan publicar sus datos y ofrecerlos en tiempo real. De esta manera, lo ideal sería que cada conjunto de datos mencionados anteriormente fuese mantenido por cada administración.

4.1.3 Identificar atributos

Una vez se han obtenido los conjuntos de datos de centros sociosanitarios, con la información necesaria para poder crear la API REST, se pretende identificar en cada uno de ellos, los atributos más importantes. Finalmente, los atributos más relevantes son los siguientes:

- Nombre
- Teléfono
- Fax
- Municipio
- Dirección
- Código Postal
- Página Web
- Email
- Latitud
- Longitud

4.1.4 Vocabularios y ontologías

Posteriormente, se busca los vocabularios/ontologías que permiten describir cada conjunto de datos especificado anteriormente.

Para ello, se intentó buscar una ontología específica que describiera centros según su especialidad en páginas como LOV [92]. Es decir, se buscó vocabulario relacionado con los centros socio sanitarios pero solo se encontraba ontologías relacionadas con enfermedades y esto no aportaba nada al ámbito del proyecto.

Finalmente se optó por las siguientes ontologías, en vez de crear una nueva ya que uno de los principios fundamentales de la Web Semántica es la de reutilizar código y, por consiguiente, las descritas a continuación, cubren perfectamente con el ámbito de los centros socio sanitarios.

- ORG: Describe las estructuras organizativas y la información relacionada con las mismas.
- FOAF: Vocabulario que vincula a personas y a la información que se utiliza en la Web.
- GEO: Espacio de nombres que se utiliza para representar la latitud, longitud y otra información sobre lugares.
- vCard: Describe organizaciones y personas.

4.1.5 Mapear conjunto de datos

Seguidamente, al encontrar el vocabulario adecuado se debe de mapear los conjuntos de datos (farmacias, botiquines, …) con cada atributo definido por las ontologías que se han descrito anteriormente según las especificaciones RDF.

Por ejemplo, para describir cada conjunto de datos, se hace mediante el vocabulario ORG el cual estructura cada centro como una organización. A continuación, se muestra un sencillo ejemplo de organización que pretende simular un centro sociosanitario:

```
1:<org:Organization
rdf:about="http://taro.ull.es/resource/centers/CLINICA_DENTAL_JOSE_DOMINGO_
MENDEZ">
2:    <org:hasRegisteredSite>
3:      <org:Site>
4:        <vCard:Name>CLINICA DENTAL JOSÉ DOMINGO MÉNDEZ</vCard:Name>
5:       <org:siteAddress>
6:          <vCard:Organization>
7:            <vCard:hasTelephone rdf:parseType="Resource">
8:              <vCard:hasValue rdf:resource="922 373 618" />
9:              <rdf:type rdf:resource="http://www.w3.org/2006/vcard/ns#Home" />
10:             <rdf:type rdf:resource="http://www.w3.org/2006/vcard/ns#Voice" />
11:           </vCard:hasTelephone>
12:           <vCard:hasAddress>
13:             <vCard:Address>
14:               <vCard:locality>PUERTO DE LA CRUZ</vCard:locality>
15:               <vCard:postal-code>38400</vCard:postal-code>
16:                <vCard:street-address />
17:             </vCard:Address>
18:           </vCard:hasAddress>
19:           <vCard:geo>28.4157901, -16.5504486</vCard:geo>
20:           <geo:lat>28.4157901</geo:lat>
21:           <geo:long>-16.5504486</geo:long>
22:         </vCard:Organization>
23:       </org:siteAddress>
24:     </org:Site>
25:   </org:hasRegisteredSite>
26: </org:Organization>
```

Para empezar, se muestra las etiquetas de una organización donde *rdf:about* es la URI del centro a describir. En la segunda y tercera línea se indica que se trata de un sitio registrado legalmente. En la cuarta línea, se especifica el nombre del centro sociosanitario. En las líneas cinco y seis se abren las etiquetas correspondientes la indicación de dirección de una organización. Desde la línea siete a la once, se establecen diferentes formas de indicar el teléfono del centro. A continuación, en las líneas 14 y 15 se establece la localidad y el código postal. Finalmente, las líneas 19, 20 y 21 mediante los vocabularios *vCard* y *geo* indican la latitud y la longitud.

4.1.6 Transformar conjuntos de datos

Para poder hacer de manera automatizada el mapeo de datos, se ha utilizado la herramienta XSLT [93], el cual permite transformar documentos XML a otros formatos, en este caso RDF.

Los formatos de los conjuntos de datos obtenidos de las administraciones públicas han sido: CSV y XML. Por lo que, los que estaban en CSV se pasaron a XML mediante el conversor de formatos online [94] para poder usar XSLT.

Aunque existen diversas herramientas para transformar datos, se ha escogido XSLT puesto que en anteriores proyectos se había utilizado y resultaba de fácil aprendizaje y uso. Además, para comprobar que el RDF resultante es correcto se hace a través del validador de W3C [95].

4.1.7 Consultas SPARQL

Seguidamente, se desarrollan los servicios que va a tener la API REST para los centros sociosanitarios. Cada servicio implementado contiene una consulta SPARQL a los conjuntos de datos ya convertidos en RDF y, ésta consulta devuelve el resultado con los datos en formato JSON para que sean utilizados para crear aplicaciones web o móviles.

4.1.8 Atributos descritos con 5 estrellas

Posteriormente, se analizan los atributos que pueden ser descritos con 5 estrellas y así proveer contexto. Es decir, deben estar en formato RDF y utilizar URIs para denotar a las cosas. Además, deben estar vinculados con información de otros usuarios. Para ello, se puede realizar el descubrimiento de fuentes de datos enlazados mediante las herramientas Silk [78] y Limes [79] basados en el paradigma de Linked Data.

No todos los centros descritos en el proyecto contienen las 5 estrellas. Aunque cada uno de ellos está identificado por una URI y cada conjunto de datos forma parte de un RDF preestablecido, no se ha podido hacer el enlazado con recursos externos debido a que esos centros no se encontraban en fuentes

externas. Por consiguiente, en el siguiente punto se explica el enlazado con otros datos.

4.1.9 Linked Data

A continuación, se enlazan los conjuntos de datos obtenidos de las administraciones públicas con la fuente externa BBpedia. De esta manera, se consigue obtener las 5 estrellas estandarizadas por Open Data.

Se ha intentado buscar más fuentes de datos para poder seguir enlazando, pero dado que las administraciones públicas no publican datos y no se encuentran más información relevante, no se ha podido seguir realizando el concepto de Linked Data.

Para poder enlazar con datos del exterior se ha utilizado la ontología *owl:sameAs* indicando la URI del recurso, como se muestra a continuación:

<owl:sameAs
rdf:resource="http://es.dbpedia.org/page/Hospital_Universitario_de_Canarias"/>

4.1.10 Publicar datos

Finalmente, se deben publicar los conjuntos datos en un catálogo de dato, desarrollar una API REST y/o la aplicación web.

En este proyecto se han realizado las tres opciones descritas anteriormente, puesto que se han publicado los datos en el portal de Open Data Canarias y se ha desarrollado una API REST consumida por una aplicación web donde se presentan geolocalizados los centros sociosanitarios y permite la búsqueda de información acerca de los mismos.

A continuación, se presenta un diagrama con las fases de la metodología que se ha seguido para la realización del proyecto.

Figura 4.1. Metodología: diagrama de fases.

4.2 Implementación de la API REST

En la sección anterior, se ha explicado los pasos seguidos para la realización del proyecto. A continuación, se presenta los elementos principales para la implementación del mismo. Para poder hacer la implementación se ha contado con la ayuda y el asesoramiento de Laura Mª Gutiérrez Medina.

4.2.1 Aplicación Java

La implementación de la API REST se ha estructurado de la siguiente manera: El proyecto padre principal es css (centros sociosanitarios) a continuación se establecen dos subproyectos hijos:

- css-core: Contiene los servicios, recursos y clases necesarias para la obtención de datos de fuentes externas e internas. Compone la vista del modelo MVC.

- css-rest: Dispone de métodos que llaman a las clases del css-core para obtener los datos y pasarlos a la aplicación o interfaz web. Es decir, basándonos en el modelo MVC, se trata del controlador.

Figura 4.2. Estructura de la API REST de los centros sociosanitarios.

4.2.2 Recursos

Como ya se ha mencionado anteriormente, se han obtenido una gran parte de los recursos, de los centros sociosanitarios, de las administraciones públicas IASS y Servicio Canario de Salud. Para que estos puedan ser consumidos por la API REST es necesario ponerlos en la sección de recursos en el subproyecto css-core como se muestra a continuación.

Figura 4.3. Recursos RDF en la estructura css-core.

Se tratan de cinco RDF (botiquines, centros de discapacidad, centros de mayores, farmacias y otros centros) con la información más relevante acerca de cada centro. Es decir, en cada uno de ellos podemos encontrar la siguiente información: nombre, teléfono, fax, municipio, dirección, código postal, págin web, email, latitud y longitud. Evidentemente, la situación ideal sería que esta información la publicasen las administraciones públicas en sus portales de datos.

La información de fuentes externas se obtiene del endpoint de DBpedia, de la cual extrae los datos de las camas, organizaciones afiliadas y la descripción del mismo.

4.2.3 Servicios

Una vez explicado donde se encuentran los ficheros RDF con la información de cada centro sociosanitario y de donde se extraen la información externa para aplicar el concepto de Linked Data, se pretende exponer los servicios implementados en la API REST.

Por un lado, para cada fuente de datos obtenidos de las administraciones públicas se han implementado los siguientes servicios:

- Obtener todos los recursos a partir de un nombre dado y como respuesta devuelve el nombre, URI, latitud y longitud de cada recurso que cumpla la condición.
- Conseguir toda la información de un recurso a partir de la URI. Los datos devueltos son: nombre, código postal, latitud, longitud, municipio, domicilio, web, email, teléfono y fax.
- Obtener todos los recursos que se encuentran alrededor en X metros de un recurso dado.
- Al estar categorizados los centros sociosanitarios se puede hacer una búsqueda según una categoría dada. Es decir, se puede buscar por especialización. Por ejemplo, el usuario necesita encontrar un centro donde tengan la especialidad de neurología, por lo que escogerá esa opción y se visualizarán sólo los que cumplan esa condición.

Por otro lado, para poder hacer Linked Data, se ha obtenido datos de la fuente externa DBpedia. De esta manera, al hacer una consulta a los RDF de las fuentes de las administraciones públicas se encuentra la etiqueta <owl:sameAs> con una URI asociada, esto significa que hace referencia al mismo recurso que hay en DBpedia pero con distinta información que puede ser de utilidad. Los datos obtenidos son los siguientes:

- Descripción.
- Organizaciones afiliadas.
- Número de camas.

A continuación, en la imagen se presenta un ejemplo de una llamada al servicio farmacias en un navegador y ésta llamada devuelve un JSON con los datos obtenidos.

Figura 4.4. Llamada al servicio de farmacias para obtener la información de la misma

Finalmente, se presenta un ejemplo de cada tipo de las llamadas a los servicios implementados en la API REST según la búsqueda solicitada:

- Buscar el recurso solicitado según el nombre especificado en formato alfanumérico:
  GET **/v1/center?name=**[alphanumeric]

- A de la URI de un centro sociosanitario, solicitar la información del mismo:
  GET **/v1/centerUri?uri=**[alphanumeric]

- Realizar una búsqueda de los recursos más cercanos (en metros) de un centro especificado por medio de la URI:
  GET **/v1/centerAround?uri=**[alphanumeric]**&radius=**[float]

- Acceder a los centros sociosanitarios a través de la especialización (categoría) del mismo:
  GET **/v1/centerCat?category=**[alphanumeric]

- Buscar información adicional del centro especificado por medio de la URI:
  GET **/v1/info?uri=**[alphanumeric]

4.3 Diseño técnico

Este punto trata de explicar la arquitectura, el patrón de diseño y las diferentes tecnologías utilizadas para la realización de la aplicación web.

4.3.1 Arquitectura

Los usuarios podrán conectarse a la API REST y a la aplicación web mediante un ordenador conectado a la red y que disponga de un navegador. La aplicación cliente se encuentra hospedada en el servidor Apache Tomcat el cual recibe peticiones de los usuarios (clientes) para obtener información bajo

un modelo de arquitectura cliente-servidor, es decir, el proveedor de recursos o servicios (servidor) da respuesta a las peticiones de otro programa (navegador) llamado cliente.

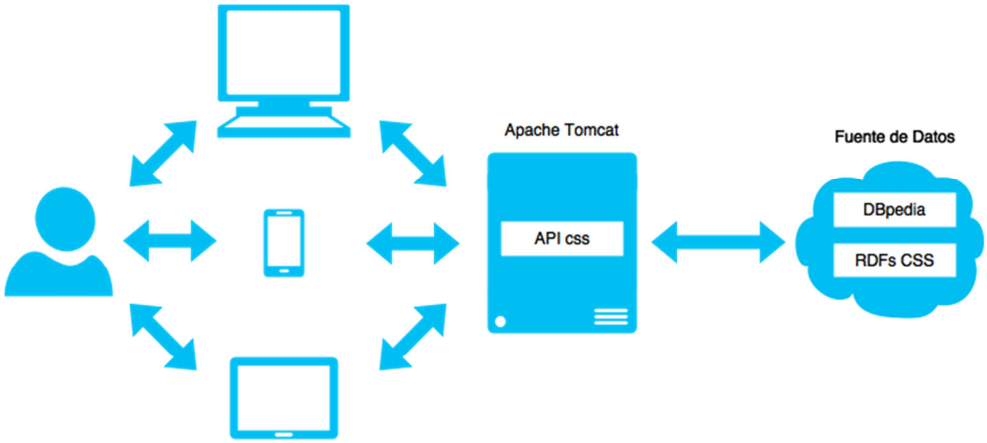

Figura 4.5. Arquitectura de la aplicación.

La información requerida por el cliente se extrae tanto de fuentes externas (DBpedia) como de fuentes internas almacenadas en el servidor en formato RDF y se mostrará en el navegador por medio de páginas HTML. Las páginas tendrán contenido estático (secciones que son iguales en todas las páginas) y dinámico (secciones que tendrán información que se obtendrá a través de la tecnología AJAX y de esta manera no se recarga la página al hacer peticiones al servidor).

4.3.2 Modelo Vista Controlador

El patrón de diseño MVC [54] es la solución para la programación web más utilizada actualmente para reutilizar y organizar código, es decir para separar conceptos y características y de esta manera facilitar el desarrollo y el mantenimiento de aplicaciones. Separa el código a desarrollar en tres capas:

- Modelo: Define la lógica de negocio, es decir gestiona los accesos a los datos, su procesamiento, validación, asociación o cualquier otra tarea que conlleve a la manipulación de los datos.
- Controlador: Módulo encargado de gestionar eventos y las comunicaciones. Realiza peticiones al modelo cuando se hace alguna

solicitud de la información y se los pasa a la vista para que muestre la información al usuario.

- Vista: Presenta la información generalmente en HTML o en plantillas para que el usuario pueda interactuar con la interfaz.

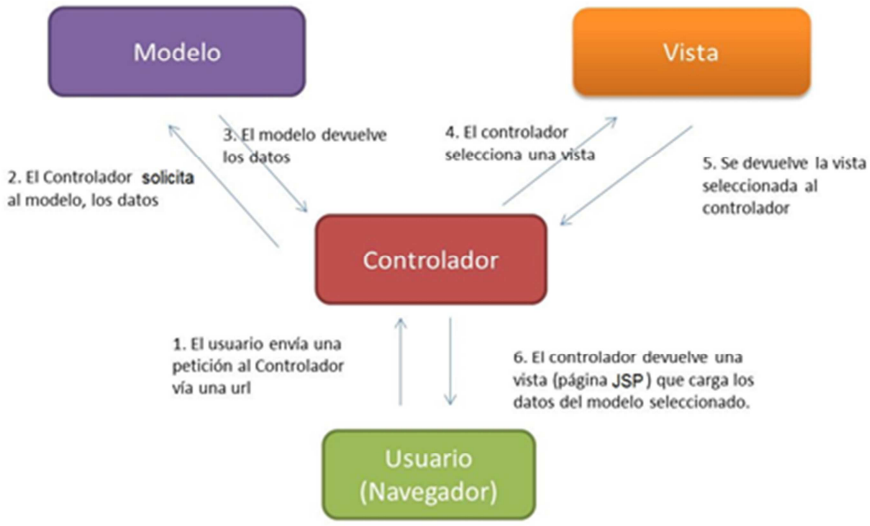

Figura 4.6. Explicación Modelo Vista Controlador.

4.4 Tecnologías

Para poder realizar el desarrollo y la implementación de la API REST como de la aplicación web se han utilizado las siguientes tecnologías:

4.4.1 Java

Se trata de un lenguaje de programación, desarrollado en 1995 por Sun Microsystems (Oracle Corporation), de propósito general, concurrente, orientado a objetos y, además, funciona en cualquier plataforma [61].

Se ha escogido este lenguaje puesto que aporta una librería (Jena) que agiliza el desarrollo con RDF y SPARQL.

4.4.2 Maven

Se ha desarrollado la aplicación utilizando el lenguaje de programación Java, bajo la gestión y construcción de proyectos Maven [60]. Se trata de una

herramienta que tiene un modelo llamado POM basado en el formato XML y que se utiliza para describir las dependencias y construir el proyecto en el orden indicado.

4.4.3 Apache Tomcat

Tomcat [48] es una aplicación de software de código abierto y funciona como un contenedor de web con soporte de servlets [86] y JSPs (JavaServer Pages) de Oracle Corporation.

La decisión de escoger el servidor Apache Tomcat 8 ha sido porque es de código abierto y funciona en cualquier sistema operativo que disponga de la máquina virtual Java, puesto que la aplicación desarrollada está en este mismo lenguaje.

4.4.4 JavaServer Pages

JSP [52] es una tecnología Java orientada a crear páginas web dinámicas en forma de documentos HTML o XML para servidores web compatibles, como Apache Tomcat o Jetty.

Dado que la aplicación web está basada en el patrón MVC y éste separa en diferentes niveles los datos, la interfaz de usuario y el módulo encargado de controlar los eventos y comunicaciones, se ha optado por usar la tecnología JSP puesto que permite separar la aplicación web en varios niveles dejando la parte de la interfaz de usuario que genera el documento HTML en el archivo JSP, en nuestro caso la vista es un archivo JSP creado dinamicamente. Además, hereda la portabilidad de Java y esto hace que sea posible ejecutar aplicaciones en otras muchas plataformas sin cambios.

4.4.5 Apache Jena

Este framework [74] se utiliza para construir aplicaciones para la Web Semántica en Java. Proporciona un entorno de programación o API para extraer datos de RDF, RDFS y OWL, y realizar consultas a través de SPARQL y GRDDL, e incluye un motor de inferencia basado en reglas.

Se ha utilizado Apache Jena para extraer datos de las fuentes proporcionadas por las autoridades públicas a través de consultas SPARQL

4.4.6 Jersey RESTful Web Service

Se trata de un framework [63] de código abierto para desarrollo de servicios web RESTfull sobre el protocolo HTTP en Java que da soporte para JAX-RS API. Se utiliza para simplificar el servicio REST y el desarrollo de clientes.

Dentro del proyecto centros sociosanitarios se encuentra el subproyecto css-rest, el cual contiene el archivo API.java. Este archivo contiene el código de funcionamiento del servicio web, es decir, será el que indique que hacer al recibir peticiones por los tipos (put, get, delete, post, etc) de mensaje HTTP

A continuación, se presentan las anotaciones utilizadas para la realización de la aplicación:

- @Path (nombre): Especifica la ruta que se debe usar para invocar al método.
- @GET: Indica el tipo de método HTTP que se ejecutará.
- @Produces (MediaType.APPLICATION_JSON): Indica el tipo de salida que tiene el método, en nuestro caso es un JSON.
- @QueryParam: Sirve para inyectar parámetros a la consulta a través de la URI.

4.4.7 JSON

JSON (JavaScript Object Notation) [70] es un formato ligero para intercambiar datos de manera sencilla. Para los humanos resulta de fácil lectura y escritura y para las máquinas es simple de interpretar y de generar. Está constituido por una colección de pares de clave-valor y una lista ordenada de valores.

En la aplicación web desarrollada, el subproyecto css-rest envía la información en formato JSON a la vista.

{
"URL":"http://taro.ull.es/resource/centers/CLINICA_DENTAL_JOSE_DOMINGO_MENDEZ",

"Nombre":" CLINICA DENTAL JOSÉ DOMINGO MÉNDEZ<",
"Latitud":" 28.4157901",
"Longitud ":" -16.5504486",
}

4.4.8 IDE Eclipse

Para el desarrollo del proyecto se ha utilizado el entorno de desarrollo integrado Eclipse [71]. Se trata de una aplicación informática de código abierto compuesta por un conjunto de herramientas de programación. Los lenguajes de programación utilizados en eclipse son: Java, ANSI C, C++, JSP, sh, perl, php, etc.

4.4.9 Navegadores

El navegador [87] es un programa que permite acceso a internet y es fundamental para poder acceder a la aplicación web. Éste se encarga de interpretar la información y de visualizarla. A través del navegador se realizarán las peticiones al controlador que se encuentra en el servidor y éste último nos devuelve el resultado para que lo muestre en el navegador.

4.4.10 HTML5 y CSS3

Por una parte, HTML5 [85] es un lenguaje de marcas usado para estructurar y presentar una interfaz para el usuario en el navegador. Además, permite una mayor interacción con las páginas web y con el contenido multimedia.

Por otra parte, CSS3 es un lenguaje que permite definir el estilo y la presentación de un documento HTML para hacer más amena la presentación de información en la web.

Por tanto, para crear la interfaz de la aplicación web se han usado estas dos tecnologías.

4.4.11 Bootstrap

Se trata de un framework [76] creado por Twitter que permite crear interfaces web con CSS y JavaScript, adaptando la interfaz a los diferentes dispositivos (móvil, tablet u ordenador).

Se ha escogido este framework por la sencillez y la gran capacidad de adaptar las interfaces a los diferentes tamaños de pantalla según el dispositivo y por ofrecer una interfaz ya creada con un estilo predefinido, al cual solo debes adaptar a tu diseño.

4.4.12 JavaScript y jQuery

El lenguaje JavaScript [84] funciona del lado del cliente y los navegadores se encargan de interpretar el código. Se utiliza principalmente para crear páginas web dinámicas.

A su vez, jQuery [83] se basa en un conjunto de librerías JavaScript que se han diseñado para simplificar las animaciones, la interacción con el árbol DOM (interfaz para acceder, añadir y modificar dinámicamente el contenido de un HTML), la gestión de eventos y las interacciones AJAX.

En este proyecto es muy importante la utilización de Jquery puesto que aporta sencillez a la hora de implementar las interacciones con AJAX.

4.4.13 Git y GitHub

Por un lado Git [67] es un software que sirve para el control de versiones, pensado para proyectos con un gran número de archivos de código fuente.

Por otro lado, GitHub [68] es una plataforma que ayuda al desarrollo colaborativo de software para alojar proyectos mediante el sistema de control de versiones de Git. Proporciona auditoría del código, es decir en todo momento se puede saber quién ha modificado el código y la fecha de cuando lo hizo. También, permite volver hacia versiones anteriores si se ha tenido algún problema. Además ayuda a mejorar la capacidad de trabajar en equipo,

puesto que permite hacer diferentes ramas y cada desarrollador puede trabajar en una de ellas.

4.5 Prototipo final

A continuación, se presenta el prototipo final de la aplicación web que consume los servicios de la API REST, en la cual se va a mostrar las diferentes funcionalidades que lo contiene. La realización de la interfaz de la aplicación ha sido llevada a cabo por Luis Alberto Julio Rodríguez y Adrián Muñoz Barrera.

Cuando se inicia la aplicación, se muestra la pantalla principal en la cual se presentarán todas las interacciones con la misma. Por un lado, el usuario puede elegir si quiere que la aplicación busque su ubicación actual mediante la geolocalización para mostrar los recursos de los alrededores de su posición o, por el contrario, mostrar todos los recursos disponibles en la isla de Tenerife.

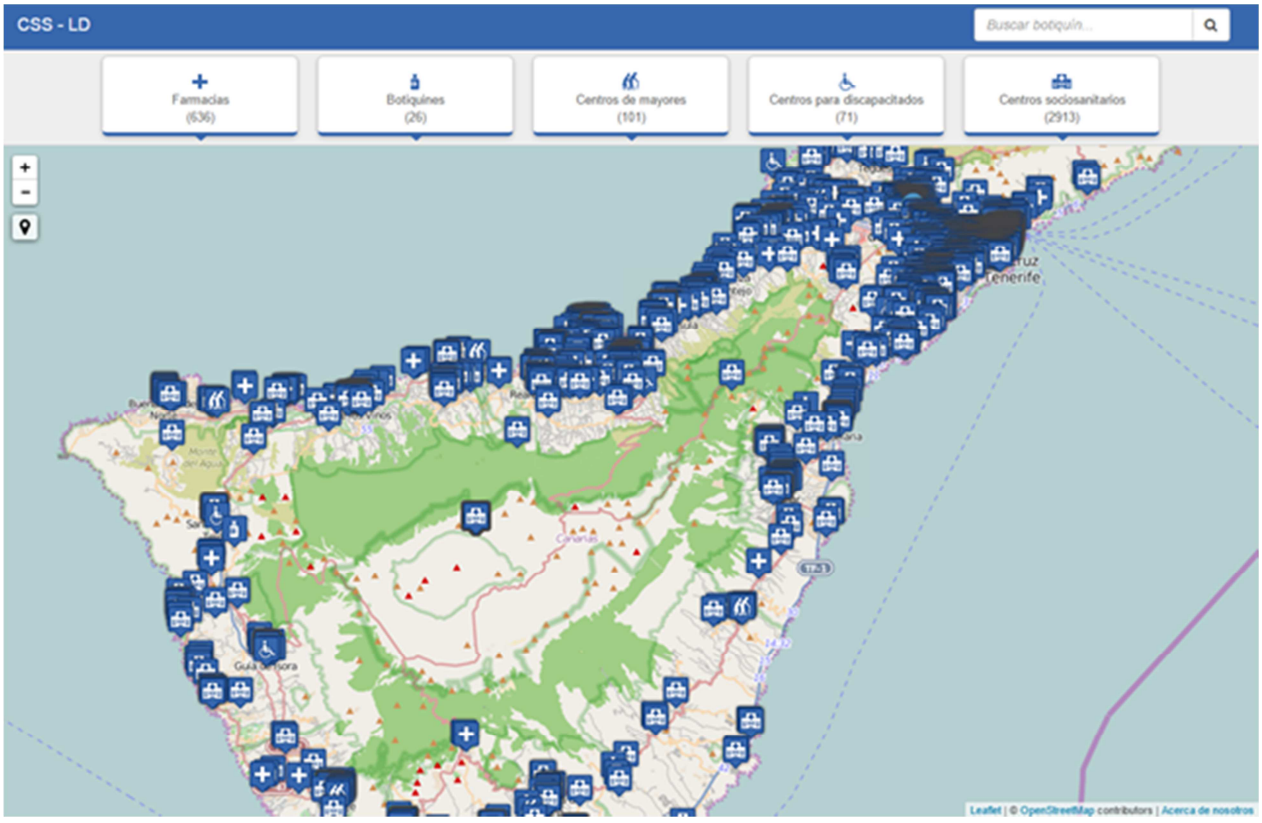

Figura 4.7. Página principal de la aplicación web sin geolocalizar la ubicación del usuario.

Por otro lado, en la misma pantalla principal, se muestra en el mapa todos los recursos geolocalizados según el tipo al que pertenezcan. Además, cada recurso estará identificado por un icono dependiendo a la categoría que representen.

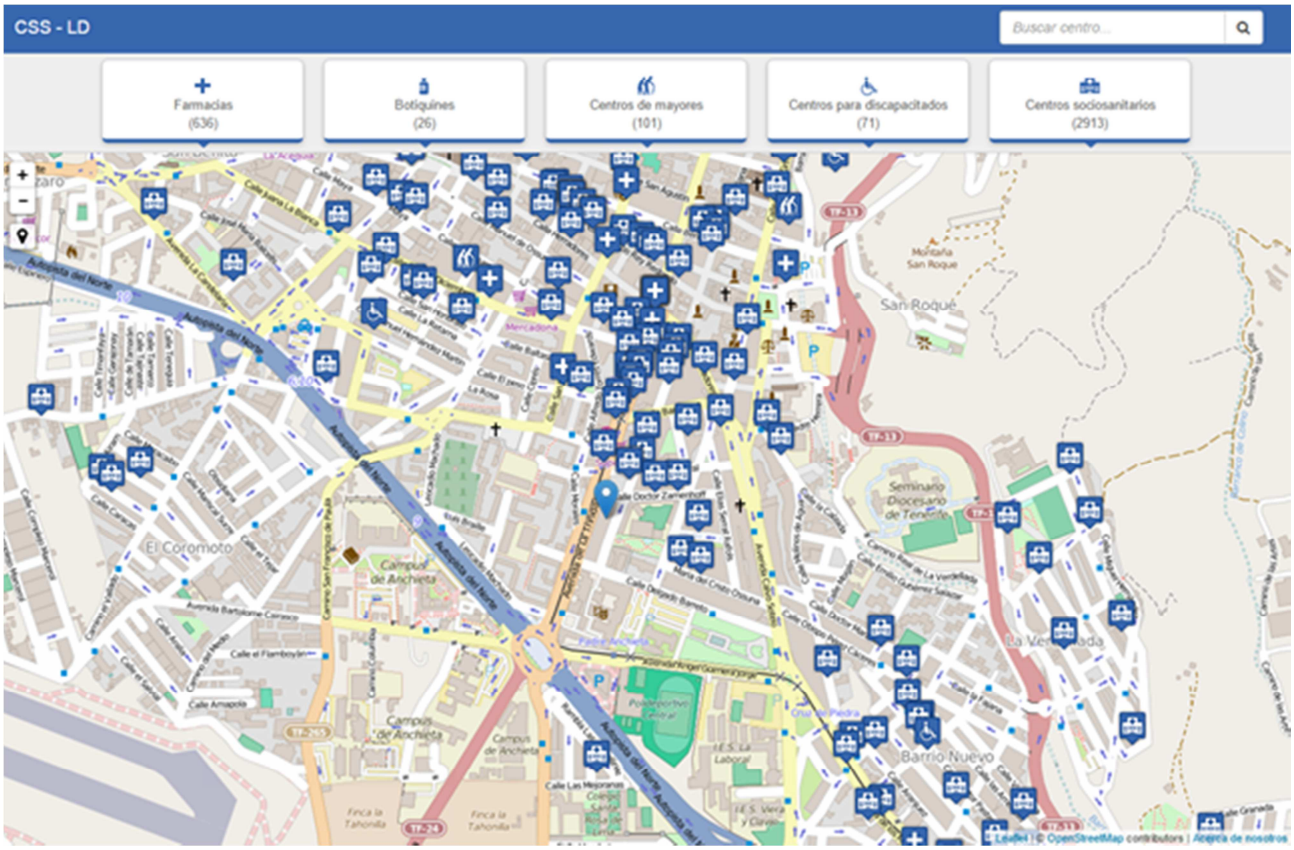

Figura 4.8. Mostrar la ubicación actual del usuario.

En esta aplicación, se pueden realizar tres tipos de búsqueda personalizada:

- Por categorías.
- Buscar por nombre del centro sociosanitario.
- Buscador del mapa a través de municipio, calle, etc.

Por una parte, se presentará una lista, en la parte superior del mapa, con los diferentes recursos (farmacias, botiquines, centros de mayores, centros para discapacitados y otros centros sociosanitarios) en la cual se muestra, en cada uno de ellos, la cantidad de centros que hay en la isla de Tenerife. Asimismo,

para hacer una búsqueda más personalizada acerca de los centros sociosanitarios deseados, se puede elegir de esa lista los recursos que sean necesarios para el usuario. Es decir, si dejamos sin seleccionar una categoría, los iconos del mapa que la represente se desaparecen, quedando solo las categorías seleccionadas. Esto es de bastante utilidad para los usuarios que se encuentren en una posición y deseen buscar, por ejemplo, las farmacias más cercanas a su ubicación.

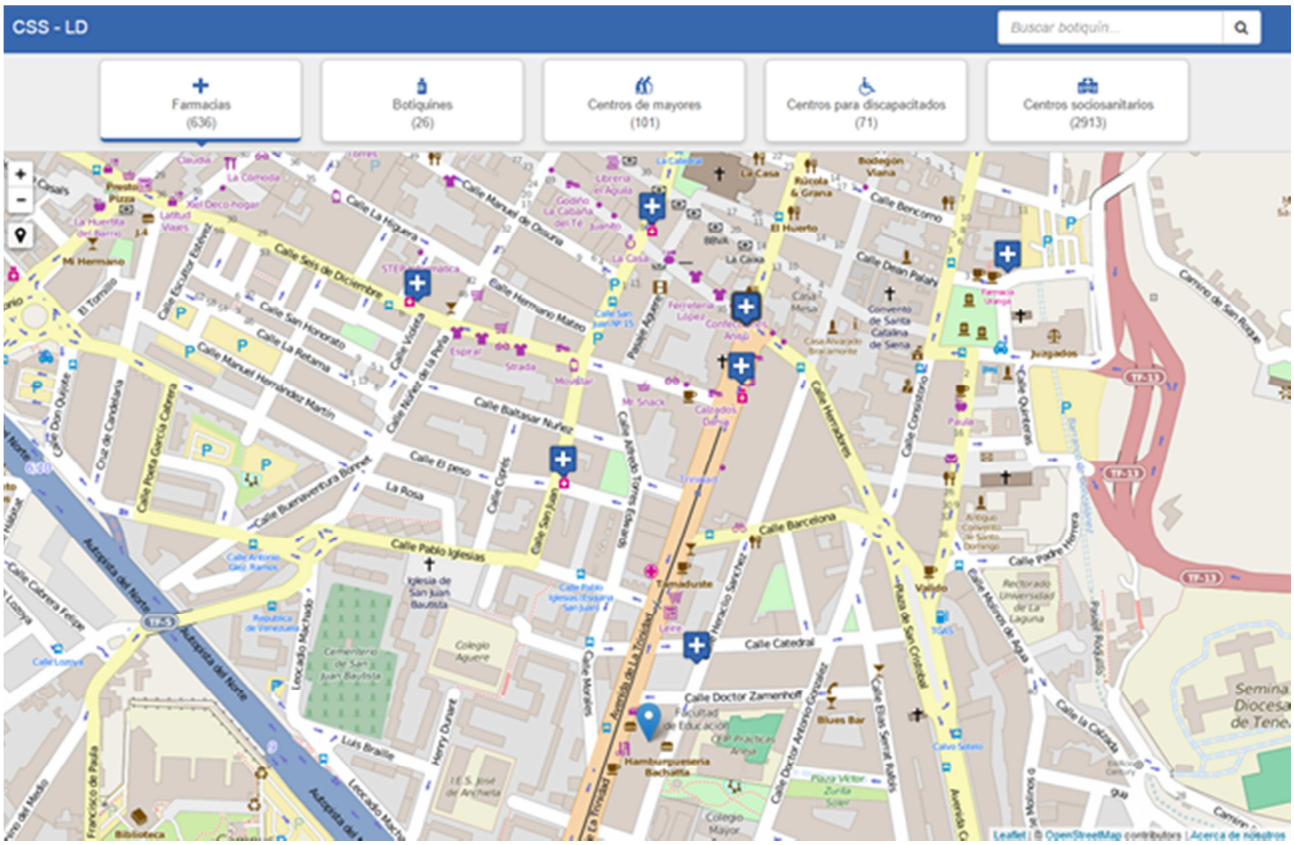

Figura 4.9. Farmacias más cercanas a la posición actual del usuario.

Otra búsqueda personalizada es la de poder buscar por nombre los recursos de los centros sociosanitarios. En siguiente ejemplo, se hace una búsqueda al *Hospital Universitario Nuestra Señora de Candelaria*, mostrando la ubicación del mismo.

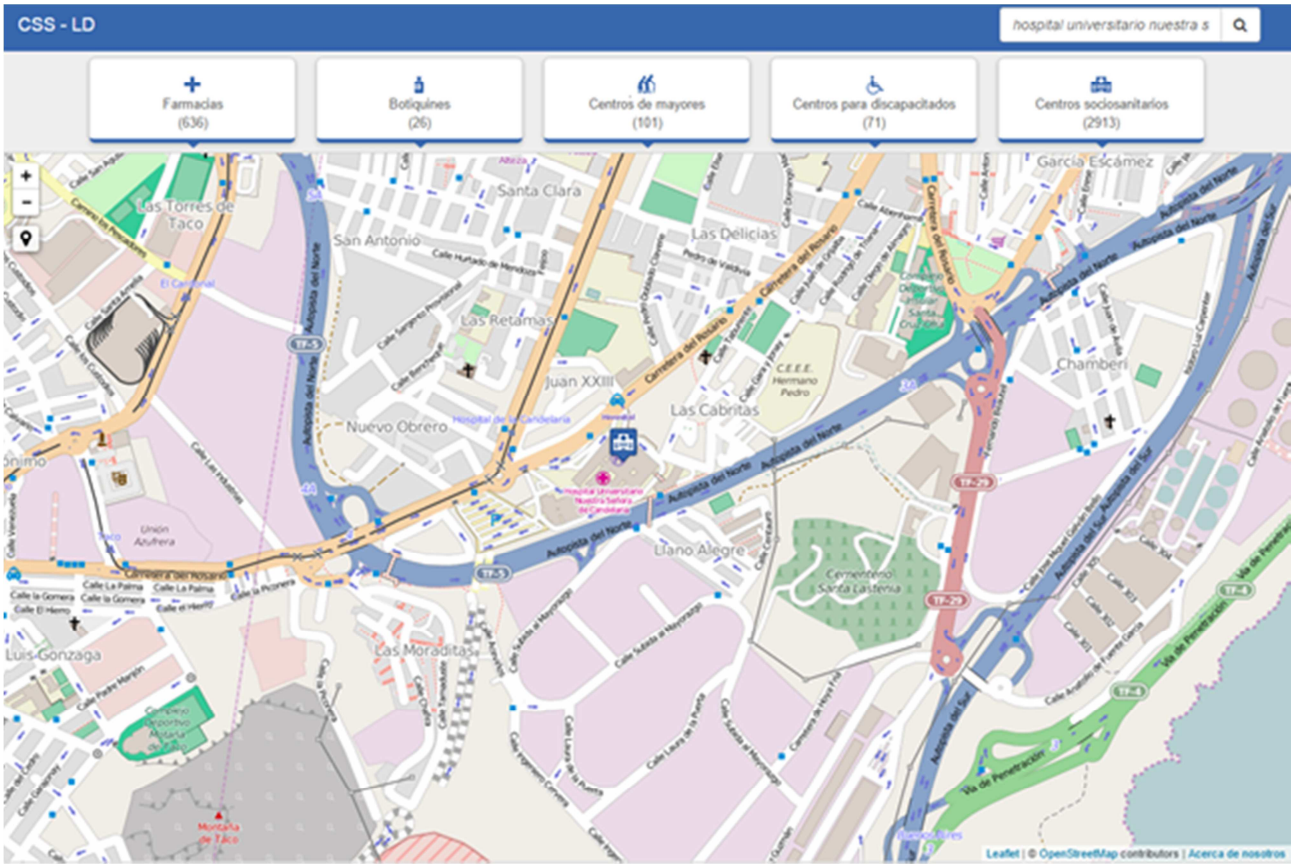

Figura 4.10. Geolocalización del Hospital Universitario Nuestra Señora de Candelaria.

Por otra parte, se puede acceder a una ubicación deseada a través del buscador del mapa, especificando el lugar exacto. En el ejemplo posterior, se hace una búsqueda en el *Puerto de la Cruz* de todos los centros disponibles en la zona.

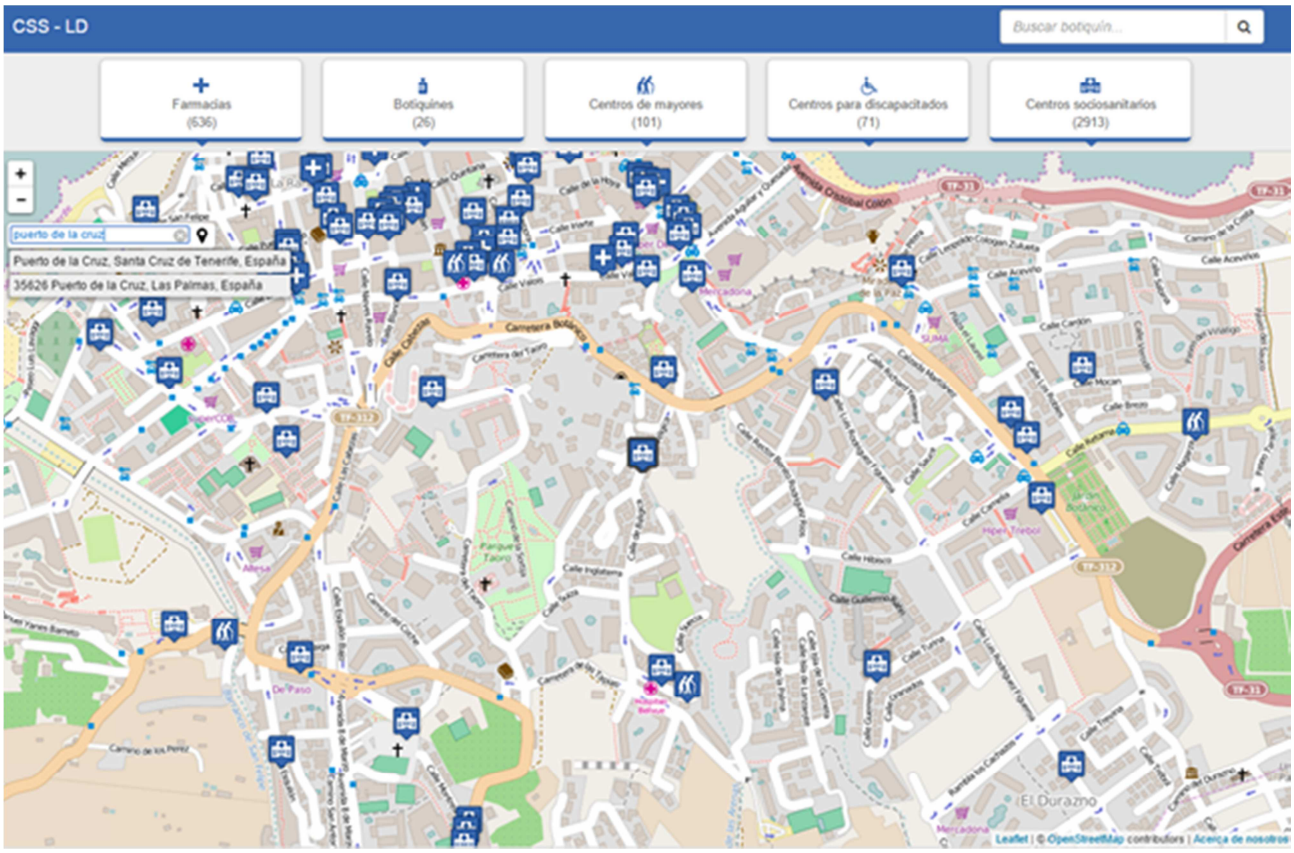

Figura 4.11. Recursos del Puerto de la Cruz a través del buscador del mapa.

Finalmente, una vez realizada la búsqueda correspondiente, se podrá acceder a la información deseada del recurso. Para ello, el usuario deberá cliquear encima del icono del recurso en el mapa y se desplegará una sección dividida en tres subsecciones:

- Información general: Muestra la información más relevante del recurso seleccionado.
- Información adicional: Muestra información extraída de fuentes externas.
- Lugares cercanos en un radio de 500 metros: Busca los recursos más cercanos al

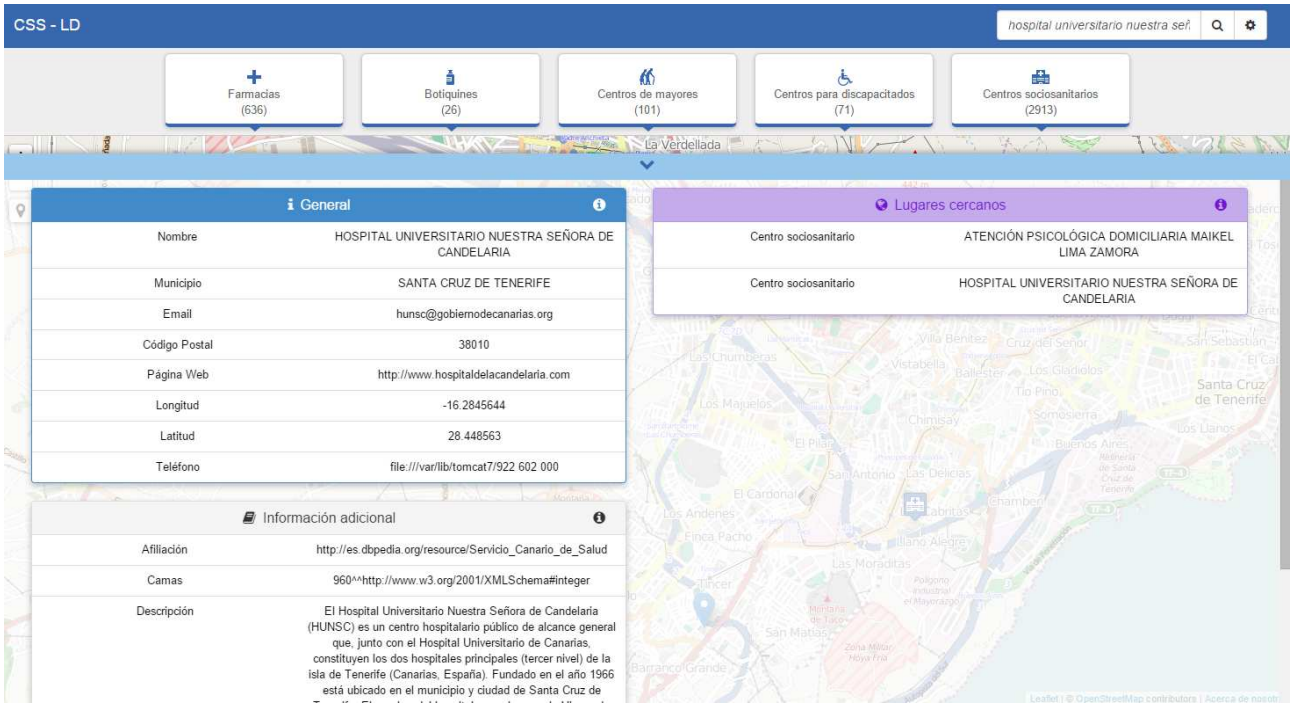

Figura 4.12. Información del centro seleccionado por el usuario.

Capítulo 5.
Conclusiones y líneas futuras

5.1 Conclusiones

La aplicación web desarrollada que consume los servicios de la API REST implementada, es un prototipo de una aplicación que podría llegar a ser un producto basado en la web de datos, destinada a la búsqueda de centros sociosanitarios geolocalizados y de la información más relevante de cada uno de ellos en la isla de Tenerife.

Un punto importante a tener en cuenta para poder seguir creando APIs y aplicaciones innovadoras, es que tanto las administraciones públicas como el resto de entidades deberían de seguir publicando datos que tienen ahora como privados o recopilados de las personas o empresas, en portales a través del estándar Open Data. Además, los desarrolladores tienen la tarea importante de seguir el concepto establecido con Linked Data para poder relacionar datos y así aumentar la calidad y la eficiencia de la información que se encuentra en la Web.

Por consiguiente, la API REST implementada cuenta con las cinco estrellas de Open Data, ya que los datos se encuentran disponibles en internet de manera estructurada y en formato RDF. Los recursos se denotan con URIs únicas pertenecientes al dominio *http://taro.ull.es/resource* y, además, se vinculan los datos obtenidos de fuentes públicas con los datos extraídos de fuentes externas, utilizando el concepto Linked Data.

Finalmente, este proyecto puede ser un manual para las administraciones públicas, ya que se muestra un ejemplo de cómo publicar sus datos y ofrecerlos en tiempo real.

5.2 Líneas Futuras

Para conseguir que en el futuro la aplicación sea un producto el cual se pudiese vender, es necesario implantar una serie de mejoras para enriquecer cada aspecto que se mencionará a continuación.

En cuanto a la API REST, dado que ha sido un trabajo difícil encontrar conjuntos de datos y fuentes externas que puedan proporcionar información relevante al ámbito sociosanitario, en el futuro se debería de seguir ésta línea, buscando conjuntos de datos útiles para poder hacer una herramienta de la cual puedan consumir las aplicaciones web y móviles. Por ello, es necesario hacer entender a las administraciones públicas que publiquen datos en sus portales para que puedan ser utilizados por los desarrolladores y así crear aplicaciones innovadoras.

En cuanto a la interfaz, se debería de crear una aplicación móvil, a parte de la aplicación web ya creada, dado que en la sociedad actual, el dispositivo que más se usa diariamente es el móvil. Además, el dispositivo dado su tamaño el usuario lo lleva a todas partes, por lo que buscar información y ver la geolocaliación de los centros es más cómodo desde una aplicación móvil que en la aplicación web.

En cuanto al hardware, para mejorar los tiempos de respuesta de las aplicaciones sería necesario disponer de un servidor caché en el cual se guarden las consultas más usadas. De esta manera, la carga de datos en la interfaz mejorará bastante y el usuario no tendrá que esperar para ver los resultados.

Capítulo 6.
Summary and Conclusions

6.1 Conclusions

The developed web application consumes services from implemented API REST, it is a prototype of application that could become a product based on the Data Web, intended to search health centers geolocated and the relevant information of each in Tenerife.

An important point to keep in mind to further develop APIs and innovative applications, is that both public administrations and other entities should continue publishing data that they have as private or collected from people or companies now, in portals through standard Open Data. Further, the developers have the important task to continue with the established concept of Linked Data to relate data and in this way, increase the quality and efficient of information that found in the Web.

Therefore, the implemented API REST has Five Star Open Data, since these data is available from the internet in a structured way an RDF format. The resources are denoted with unique URIs belonging to the domain http://taro.ull.es/resource, and linked the data from public resources with the extracted data of external sources too, using the concept Linked Data.

Finally, this project can be a manual for public administrations, in the way that it shows an example of how to publish it's data and offer it in real time.

6.2 Summary

To get that in the future the application be a product which could be sold, is necessary implement some improvements to enrich every aspect that be mentioned below.

Referring to API REST, knowing that find data set and external sources that can provide relevant information of geriatric field is a difficult task, in the future should continue in this way, searching sets of helpful data for make a tool for can be used from web applications and smartphones. For it, is necessary get understand public administrations that publish data in his portals for can be used by developers and they can build innovate applications.

As for the interface, it should create a phone application, apart from the web application already created, as today's society the most daily used device is the mobile phone. Also, the device is taken to everywhere for his size, so search information and see geolocation of centers is more comfortable from a mobile application than a web application.

Respect to hardware, for improve the response time of the applications, would be needed of server cache in which could be saved the most used queries. In this way, the loading data in the interface will improve quite and the user hasn't to wait for the results.

Capítulo 7. Presupuesto

Para la elaboración del presupuesto se tendrán en cuenta tres factores básicos, ambos relacionados entre sí: Recursos humanos, Recursos materiales y el tiempo necesario para la realización del mismo.

Definido un tiempo máximo de realización del proyecto, se estimará la cantidad de recursos humanos necesarios para llevar a cabo la labor, mientras que estos, necesitarán de un material mínimo para el desarrollo del proyecto. Cabe mencionar, que el proyecto se desarrollará bajo software de licencia libre, esto conlleva el ahorro de la obtención de licencias de pago, abaratando costes.

7.1 Recursos Humanos

La duración del proyecto, considerando el nivel de complejidad y las tareas a desarrollar, se fijará en un tiempo aproximado de cuatro meses, en los que se necesitará de profesionales con los siguientes perfiles:

- Jefe de proyecto / analista programador: Será el encargado de organizar y estructurar el proyecto, asignar tareas a los miembros y coordinar la integración de los distintos módulos.
- Ingeniero de Software: Se necesitará de una persona con los conocimientos básicos para la explotación de los datos, su manipulación y la devolución de ellos a través del servicio API-REST.
- Diseñador gráfico / Técnico en desarrollo web: Este perfil aportará la experiencia y conocimiento de la manipulación de la interfaz web, haciendo uso de técnicas de usabilidad y accesibilidad para el logro de un diseño de calidad.
- Administrador de sistemas: Los componentes de software deberán ejecutarse y estar disponibles al público desde una infraestructura

hardware sólida. El encargado de sistemas deberá cumplir la función de la puesta en funcionamiento y mantenimiento de la misma, garantizando la alta disponibilidad del servicio.

Fases de realización del proyecto según los meses:

- Mes 1: Análisis y documentación
- Mes 2: Desarrollo de arquitectura
- Mes 3: Desarrollo de arquitectura
- Mes 4: Integración.

Mes / Persona	Mes 1	Mes 2	Mes 3	Mes 4	Coste
Jefe de Proyecto (gestión)	1200€	700€	700€	700€	3300€
Ingeniero de Software (core)	600€ *	1000€	1000€	1000€	3600€
Técnico en desarrollo web (interfaz)	-	-	900€	900€	1800€
Administrador de sistemas (arquitectura)	600€ *	1000€	1000€	1000€	3600€
				Total	**12300€**

Tabla 7.1. Presupuesto de los Recursos Humanos

*Tiempos parciales

7.2 Recursos Materiales

A continuación se presentan los recursos materiales necesarios para la implantación de la API REST y la aplicación web.

Herramienta	Información	Cantidad	Precio	Coste
Servicio de Cloud Computing (hosting)	1 Gb RAM 1Núcleo Procesamiento 30 Gb SSD Disk 2 TB Transferencia	1	120€ /año	120€
Ingeniero de Software (core)	4 Gb Ram 2 Núcleos Procesamiento 250 Gb Disco duro	4	460€	1840€
Pantalla	-	4	120€	480€
Ratón + Teclado	-	4	20€	80€
			Total	**2360€**

Tabla 7.2. Recursos Materiales

Bibliografía

[1] L. Yu A Developer's Guide to the Semantic Web. Springer-Verlag Berlin Heidelberg, 2011.

[2] Wikipedia. *Web Semántica* . [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Web_semántica

[3] W3C. *Guía Breve de Web Semántica*. [Último acceso: 29/06/2015] http://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica

[4] María Jesus Lamarca Lapuente. *Hacia la Web Semántica*. [Último acceso: 29/06/2015] http://www.hipertexto.info/documentos/web_semantica.htm

[5] Ontology Engineer ingGroup. *Web Semántica y Linked Data*. [Último acceso: 29/06/2015] http://mayor2.dia.fi.upm.es/oeg-upm/index.php/es/researchareas/4-semanticweb

[6] Human Level Communications. Ramón Saquete. *El impredecible futuro de la Web Semántica*. [Último acceso: 29/06/2015] http://www.humanlevel.com/articulos/desarrollo-web/el-futuro-de-la-web-semantica.html

[7] Wikipedia. *Web 1.0*. [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Web_1.0

[8] Wikipedia. Web 2.0 [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Web_2.0

[9] Wikipedia. Web 3.0 [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Web_2.0

[10] Facilware. *La evolución de la Web: 1.0, 2.0 y 3.0*. [Último acceso: 29/06/2015] http://www.facilware.com/la-evolucion-de-la-web-1-0-2-0-y-3-0.html

[11] Wikipedia. *Datos Abiertos*. [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Datos_abiertos

[12] Open Data HandBook. *¿Qué son los datos abiertos?*. [Último acceso: 29/06/2015] http://opendatahandbook.org/guide/es/what-is-open-data/

[13] Open Data. Portal de dades obertes.. *¿Qué es Open Data?.* [Último acceso: 29/06/2015] http://opendata.cloudbcn.cat/MULTI/es/what-is-open-data

[14] Gobierno Abierto de Navarra. *Open Data y la Reutilización de Información del Sector Público.* [Último acceso: 29/06/2015] http://www.gobiernoabierto.navarra.es/es/open-data/que-es-open-data/open-data-y-risp

[15] El Blog de Sergio Cruz. Sergio Cruz Moral. *¿Qué es Open Data? Introducción a los Datos Abiertos.* [Último acceso: 16/06/2015] http://sergiocruz.codigofuerte.net/que-es-el-open-data-introduccion-a-los-datos-abiertos/

[16] TICs y formación. Alfredo Vela. *Los 8 Principios del Open Data.* [Último acceso: 29/06/2015] http://ticsyformacion.com/2012/12/17/los-8-principios-del-open-data-infografia-infographic-internet/principiosopendata/

[17] Ayuntamiendo de Pamplona. Iruñeko Udala. *Open Data y Reutilización de Datos Públicos en el Sector Local.* [Último acceso: 15/06/2015] http://www.finodex-project.eu/sites/default/files/sitefiles/events/infodaypamplona/05-FINODEXPAMPLONA_OpenData_AytoPamplona_20141002.pdf

[18] Datos con Inteligencia. *Principios Básicos del Open Data (Datos Abiertos).* [Último acceso: 15/06/2015] http://datosconinteligencia.blogspot.com.es/2010/09/apertura-de-datos-principios-basicos.html

[19] Biblioteca del Congreso Nacional de Chile. *Las cinco estrellas de los datos abiertos.* [Último acceso: 15/06/2015] http://datos.bcn.cl/es/informacion/las-5-estrellas

[20] EC FP7 Support Action LOD-Around-The-Clock. *5 Estrellas Datos Abiertos* [Último acceso: 15/06/2015] http://5stardata.info/es/

[21] *Web Semántica y Linked Data. Tecnologías Semánticas.* [Último acceso: 15/06/2015] http://citius.usc.es/sites/default/files/formacion/BD&DS_ManuelLama.pdf

[22] Wikipedia. *Datos enlazados.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Datos_enlazados

[23] Linked Data. Connect Distributed Data across the Web. *Linked Data* [Último acceso: 15/06/2015] http://linkeddata.org/

[24] W3C. *Guía Breve de Linked Data.* [Último acceso: 15/06/2015] http://www.w3c.es/Divulgacion/GuiasBreves/LinkedData

[25] Red Temática Española de Linked Data. *Red temática Española de Linked Data.* [Último acceso: 15/06/2015] http://red.linkeddata.es/web/guest;jsessionid=68C82C727C6D5FDE40D3A06629808E4A

[26] Universidad de Oviedo. Jose Emilio Labra Gayo. *Linked Open Data.* [Último acceso: 15/06/2015] http://es.slideshare.net/jelabra/linked-open-data-datos-abiertos-enlazados

[27] Comunidad de Madrid. Madridmasd. *Apúntate al reto de los Datos Enlazados en la Web.* [Último acceso: 15/06/2015] http://www.madrimasd.org/informacionIdi/analisis/analisis/analisis.asp?id=44078

[28] Wikipedia. *Resource Description Framework.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Resource_Description_Framework

[29] Semantizando la Web. *¿Qué es RDF y para qué es bueno?* [Último acceso: 15/06/2015] https://semantizandolaweb.wordpress.com/2011/11/07/que-es-rdf-y-para-que-es-bueno/

[30] María Jesús Lamarca Lapuente. *RDF.* [Último acceso: 15/06/2015] http://www.hipertexto.info/documentos/rdf.htm

[31] RDF: *Un Modelo de Metadatos Flexible para las Bibliotecas Digitales del Próximo Milénio*.* [Último acceso: 15/06/2015] http://www.cobdc.org/jornades/7JCD/1.pdf

[32] *Nodos en blanco y valores literales.* [Último acceso: 15/06/2015] https://semantizandolaweb.wordpress.com/2013/03/16/nodos-en-blanco-y-valores-literales/

[33] Wikipedia. *Ontología (informática).* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Ontología_(informática)

[34] Alberto Barrón Cedeño. *Ontologías en acción, Protegé, OWL.* 20 de Septiembre de 2005 [Último acceso: 15/06/2015] http://www.matem.unam.mx/rajsbaum/cursos/web/ontologias2.pdf

[35] Wikipedia. *RDF Schema.* [Último acceso: 06/07/2015] https://es.wikipedia.org/wiki/RDF_Schema

[36] Alberto Gómez López. *OWL, Descripción de la Ontología.* Universidad Rey Juan Carlos. [Último acceso: 15/06/2015] https://ai.wu.ac.at/~polleres/teaching/ri2007/alberto.pdf

[37] W3C. *Lenguaje de Ontologías Web (OWL).* [Último acceso: 15/06/2015] http://www.w3.org/2007/09/OWL-Overview-es.html

[38] Carlos Casamayor. Diseño de Ontologías: *Protégé OWL – Ejemplo de las Pizzas.* [Último acceso: 15/06/2015] http://es.slideshare.net/DocThreeC/diseo-de-ontologas-protg-owl-ejemplo-de-las-pizzas

[39] Wikipedia. *OWL.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/OWL

[40] María Jesús Lamarca Lapuente. *OWL* [Último acceso: 15/06/2015] http://www.hipertexto.info/documentos/owl.htm

[41] Taniana Rodríguez y José Aguilar. *Construcción de una ontología OWL con protégé 4.* [Último acceso: 15/06/2015] http://www.ing.ula.ve/~aguilar/actividad-docente/IA/documentos/presentacionprotege.pdf

[42] Wikipedia. *SPARQL.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/SPARQL

[43] W3C. *SPARQL Lenguaje de consulta para RDF.* [Último acceso: 15/06/2015] http://skos.um.es/TR/rdf-sparql-query/

[44] DBpedia español. *Ejemplos de consultas SPARQL.* [Último acceso: 15/06/2015] http://es.dbpedia.org/Wiki.jsp?page=Ejemplos+de+consultas+SPARQL

[45] Apache Jena. *SPARQL Tutorial.* [Último acceso: 15/06/2015] https://jena.apache.org/tutorials/sparql.html

[46] Semantic Web. *SPARQL endpoint.* [Último acceso: 15/06/2015] http://semanticweb.org/wiki/SPARQL_endpoint

[47] W3C. *SPARQL By Example.* [Último acceso: 15/06/2015] http://www.w3.org/2009/Talks/0615-qbe/

[48] Wikipedia. *Tomcat.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Tomcat

[49] Apache Tomcat. *Apache Tomcat.* [Último acceso: 15/06/2015] http://tomcat.apache.org/

[50] A JBD soft. *Apache Tomcat.* [Último acceso: 15/06/2015] http://www.ajpdsoft.com/modules.php?name=Encyclopedia&op=content&tid=769

[51] Edu4java. *¿Qué es un Contenedor de Servlets? Instalación Apache Tomcat.* [Último acceso: 15/06/2015] http://www.edu4java.com/es/servlet/servlet1.html

[52] Wikipedia. *JavaServer Pages.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/JavaServer_Pages

[53] A JBD soft. *JSP.* [Último acceso: 15/06/2015] http://www.ajpdsoft.com/modules.php?name=Encyclopedia&op=content&tid=831

[54] Wikipedia. *Modelo-vista-controlador.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador

[55] Cake Software Foundation, Inc. *Entendiendo el Modelo – Vista – Controlador.* [Último acceso: 15/06/2015] http://book.cakephp.org/2.0/es/cakephp-overview/understanding-model-view-controller.html

[56] *Patrón de arquitectura Modelo Vista Contolador (MVC)* [Último acceso: 15/06/2015] http://www.lab.inf.uc3m.es/~a0080802/RAI/mvc.html

[57] Juan Pavón Mestras. *El patrón Modelo-Vista-Controlador (MVC)* [Último acceso: 15/06/2015] https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.MVC.pdf

[58] Universitad de Alicante. *Modelo Vista Controlador (MVC)* [Último acceso: 15/06/2015] http://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html

[59] Libros Web. *La arquitectura MVC* [Último acceso: 15/06/2015] http://librosweb.es/libro/jobeet_1_4/capitulo_4/la_arquitectura_mvc.html

[60] Wikipedia. *Maven* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Maven

[61] Wikipedia. *Java (lenguaje de programación)* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

[62] Jersey Java. *Jersey RESTfull Web Services in Java.* [Último acceso: 15/06/2015] https://jersey.java.net/

[63] Wikipedia. *Java API for RESTfull Web Services.* [Último acceso: 15/06/2015] https://en.wikipedia.org/wiki/Java_API_for_RESTful_Web_Services

[64] Wikipedia. *Project Jersey* [Último acceso: 15/06/2015] https://en.wikipedia.org/wiki/Project_Jersey

[65] Un servicio web RESTful en java con Apache Tomcat 8 y Jersey 2 Jax-Rs. Retornando XML, JSON y HTML [Último acceso: 15/06/2015] http://www.programarya.com/Cursos/Java-Avanzado/RESTful/

[66] *Conocimientos de GitHub.* [Último acceso: 02/07/2015] http://conociendogithub.readthedocs.org/en/latest/data/introduccion/

[67] Wikipedia. *Git* [Último acceso: 02/07/2015] https://es.wikipedia.org/wiki/Git

[68] Wikipedia. *GitHub* [Último acceso: 02/07/2015] https://es.wikipedia.org/wiki/GitHub

[69] JsonOrg. *Introducción a JSON* [Último acceso: 15/06/2015] http://json.org/json-es.html

[70] Wikipedia. *JSON* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/JSON

[71] Wikipedia. *Eclipse (software)* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Eclipse_(software)

[72] Wikipedia. *Ambiente de desarrollo integrado.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Ambiente_de_desarrollo_integrado

[73] Apache Jena. *RDF.* [Último acceso: 15/06/2015] https://jena.apache.org/

[74] Wikipedia. *Jena (framework)* [Último acceso: 15/06/2015] https://en.wikipedia.org/wiki/Jena_(framework)

[75] *¿Qué es Apache Jena?* [Último acceso: 15/06/2015] https://unpocodejava.wordpress.com/2012/07/27/que-es-apache-jena/

[76] Wikipedia. *Twitter Bootstrap.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Twitter_Bootstrap

[77] LibrosWeb. *Bootstap 3, el manual official* [Último acceso: 15/06/2015] https://librosweb.es/libro/bootstrap_3/

[78] Silk. *The Linked Data Integration Framework* [Último acceso: 06/07/2015] http://silk-framework.com/

[79] AKSW. *Limes Link discovery framework for Metric Spaces* [Último acceso: 06/07/2015] http://aksw.org/Projects/LIMES.html

[80] *¿Qué es Bootstrap y cómo funciona en el diseño web?* [Último acceso: 15/06/2015] http://www.arweb.com/chucherias/editorial/%C2%BFque-es-bootstrap-y-como-funciona-en-el-diseno-web.htm

[81] Wikipedia. *HTML 5* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/HTML5

[82] Wikipedia. *Hoja de estilos en cascada.* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada

[83] Wikipedia. *jQuery* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/JQuery

[84] Wikipedia. JavaScript. [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/JavaScript

[85] Wikipedia. *AJAX* [Último acceso: 15/06/2015] https://es.wikipedia.org/wiki/AJAX

[86] Wikipedia. *Java Servlet* [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Java_Servlet

[87] Wikipedia. *Navegador Web.* [Último acceso: 29/06/2015] https://es.wikipedia.org/wiki/Navegador_web

[88] Wikipedia. *Hypertext Transfer Protocol.* [Último acceso: 30/06/2015] https://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol

[89] DBpedia. [Último acceso: 02/07/2015] http://es.dbpedia.org/

[90] Open Data Canarias [Último acceso: 02/07/2015] http://www.opendatacanarias.es/

[91] Wikipedia. Web Scraping [Último acceso: 02/07/2015] https://es.wikipedia.org/wiki/Web_scraping

[92] Linked Open Vocabularies (LOV) [Último acceso: 02/07/2015] http://lov.okfn.org/dataset/lov/

[93] Wikipedia. *Extensible Stylesheet Language Transformations* (XSLT) [Último acceso: 02/07/2015] https://es.wikipedia.org/wiki/Extensible_Stylesheet_Language_Transformations

[94] Wikipedia. *Convert CSV to XML* [Último acceso: 02/07/2015] http://www.convertcsv.com/csv-to-xml.htm

[95] W3C. *RDF Validator* [Último acceso: 02/07/2015] http://www.w3.org/RDF/Validator/

[96] Servicio Canario de Salud. [Último acceso: 03/07/2015] http://www3.gobiernodecanarias.org/sanidad/scs/

[97] Instituto Insular de Atención Social y Sociosanitaria [Último acceso: 03/07/2015] http://www.iass.es/iass/DEFAULT/index.jsp

[98] W3C. *GLD Life Cycle.* [Último acceso: 06/07/2015] http://www.w3.org/2011/gld/wiki/GLD_Life_cycle

[99] LOD2. *Pila LOD2 – Ciclo de vida* [Último acceso: 06/07/2015] http://stack.lod2.eu/blog/