



Universidad Técnica Particular de Loja
La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y
COMPUTACIÓN

**Almacenamiento, procesamiento y visualización de datos de geo-
localización en Android en formato RDF e interacción con repositorios
en la nube**

TRABAJO DE TITULACIÓN

AUTOR: Plascencia Arévalo, Alex Daniel

DIRECTOR: Ramírez Coronel, Ramiro Leonardo, Mgs.

LOJA-ECUADOR

2017



Esta versión digital, ha sido acreditada bajo la licencia Creative Commons 4.0, CC BY-NY-SA: Reconocimiento-No comercial-Compartir igual; la cual permite copiar, distribuir y comunicar públicamente la obra, mientras se reconozca la autoría original, no se utilice con fines comerciales y se permiten obras derivadas, siempre que mantenga la misma licencia al ser divulgada. <http://creativecommons.org/licenses/by-nc-sa/4.0/deed.es>

2017

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Ing. Ramiro Leonardo Ramírez Coronel, Mgs.

DOCENTE DE LA TITULACIÓN

De mi consideración,

El presente trabajo de titulación Almacenamiento, procesamiento y visualización de datos de geo-localización en Android en formato RDF e interacción con repositorios en la nube realizado por Plascencia Arévalo Alex Daniel; ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, septiembre de 2017

f)

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

“ Yo, **Plascencia Arévalo Alex Daniel**, declaro ser autor del presente trabajo de titulación: **Almacenamiento, procesamiento y visualización de datos de geo-localización en Android en formato RDF e interacción con repositorios en la nube**, de la titulación de Ingeniería en Sistemas Informáticos y Computación, siendo Ramiro Leonardo Ramírez Coronel 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 investigado 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 que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad”.

f.

Autor: Plascencia Arévalo Alex Daniel

Cédula: 1104635501

DEDICATORIA

Dedico el presente trabajo a mi familia por todo el apoyo y ánimo brindado durante mi formación profesional, a mis padres por todos los consejos, valores que me fueron inculcados desde etapas tempranas de mi vida, a mi hermano por su consejo, guía e inspiración para la elección de mi carrera profesional, a mis docentes por los conocimientos y experiencias compartidas durante estos años de universidad, que me ayudaron al desarrollo de las capacidades y competencias necesarias para mi vida profesional.

AGRADECIMIENTO

Agradezco primeramente a mis padres y a mi hermano por todo el cariño, apoyo y ánimo durante mi formación académica, por dame fortaleza en los momentos en los que sentía que ya no podía más; a mi director de trabajo de fin de titulación por la confianza y apoyo brindados, a mis compañeros por tantos años de colaboración, malas noches y algunas risas.

INDICE DE CONTENIDOS

CARÁTULA	I
APROBACIÓN DEL TRABAJO DE FÍN DE TITULACIÓN	II
DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS.....	III
DEDICATORIA	IV
AGRADECIMIENTO	V
INDICE DE CONTENIDOS.....	VI
ÍNDICE DE FIGURAS.....	X
INDICE DE TABLAS	XI
RESUMEN.....	1
ABSTRACT	2
INTRODUCCIÓN.....	3
1. CAPÍTULO 1: MARCO TEÓRICO	4
1.1. Conceptos Fundamentales.....	5
1.1.1. Web Semántica.	5
1.1.2. DBpedia.....	5
1.1.2.1. Ontologías de DBpedia.	5
1.1.3. Android.....	6
1.1.4. Servicios Web.....	6
1.1.4.1. Protocolo Simple de Acceso a Objetos (SOAP)	6
1.1.4.2. Transferencia de estado representacional (REST)	7
1.1.4.3. Comparación REST y SOAP	7
1.2. Estándares.....	7
1.2.1. Linked Data.	7
1.2.2. Marco de descripción de recursos (RDF).....	8
1.2.3. Notación de Objetos JavaScript (JSON).	8
1.3. Lenguajes	9
1.3.1. JAVA.	9
1.3.2. Preprocesador de Hipertexto (PHP).	9
1.3.3. JAVASCRIPT	9
1.3.4. Protocolo y Lenguaje de Consulta RDF (SPARQL).	9
1.3.4.1. SPARQL endpoints	10
1.4. Librerías, frameworks e Interfaces de programación de Aplicaciones (API)	10
1.4.1. VOLLEY	10
1.4.2. Jenna.....	11

1.4.3.	Codelgniter.....	11
1.4.4.	Google Maps API.....	11
1.5.	Comentarios finales de capítulo.....	11
2.	CAPÍTULO 2: PROPUESTA DE SOLUCIÓN.....	12
2.1.	Problemática Actual.....	13
2.2.	Análisis de trabajos relacionados.....	13
2.2.1.	Semantic driven location services in the Web of Data.....	13
2.2.2.	Investigación y análisis de aplicaciones, para proponer un mejor consumo y visualización de datos enlazados.....	14
2.2.3.	Combining Linked Data and Mobiles Devices to improve access to OCW	15
2.2.4.	Linked Data-driven Geographic Mobile learning Application.....	16
2.2.5.	DBpedia Mobile Explorer.....	16
2.2.6.	Mobile Advertising using Location Based Services.....	17
2.2.7.	DBpedia Viewer - An Integrative Interface for DBpedia Leveraging the DBpedia Service Eco System.....	17
2.2.8.	Mobile Location-Driven Associative Search in DBpedia with Tag Clouds.....	18
2.2.9.	MapXplore: Linked Data on the appstore.....	19
2.2.10.	HDTourist: Exploring Urban Data on Android Elena.....	19
2.2.11.	Comparación de trabajos relacionados.....	20
2.3.	Solución Propuesta.....	21
2.3.1.	Objetivo general.....	21
2.3.2.	Objetivos Específicos.....	21
2.3.3.	Descripción de la solución.....	21
2.3.4.	Metodología de desarrollo.....	22
2.3.4.1.	SCRUM.....	22
2.3.4.2.	Pre-Game.....	23
2.3.4.2.1.	Planeación.....	23
2.3.4.3.	Game.....	23
2.3.4.3.1.	Desarrollo de Sprints.....	23
2.3.4.4.	Post-Game.....	23
2.3.4.4.1.	Cierre.....	23
2.4.	Comentarios finales del capítulo.....	23
3.	CAPÍTULO 3: DESARROLLO DE LA SOLUCIÓN.....	24
3.1.	Pre-Game.....	25
3.1.1.	Planeación.....	25
3.2.	Game.....	25

3.2.1.	C001- Consumir información geo-referenciada.....	25
3.2.1.1.	Diseño de consultas SPARQL.....	25
3.2.1.2.	Diseño del servicio Web.....	27
3.2.2.	A001- Almacenar recursos favoritos	28
3.2.2.1.	Diseño de la Base de datos.....	28
3.2.2.2.	Diseño del servicio web para almacenamiento.....	29
3.2.3.	C002- Realizar las consultas desde dispositivo móvil.	30
3.2.3.1.	Diseñar arquitectura de la Aplicación Móvil.	30
3.2.3.1.1.	Módulo de Administrador localización.....	31
3.2.3.1.2.	Módulo de Cliente REST.	31
3.2.3.1.3.	Módulo de Administrador Mapas.	31
3.2.3.1.4.	Módulo de persistencia.....	31
3.2.3.1.5.	Módulo de Administrador de cuentas.....	31
3.2.3.2.	Desarrollo del Módulo de Administrador de localización. (Móvil)	32
3.2.3.3.	Desarrollo del Módulo ClienteREST (Móvil)	33
3.2.3.4.	Diseño de la Arquitectura de la Aplicación web.	34
3.2.3.4.1.	Servidor web Apache.....	35
3.2.3.4.2.	Servidor de Aplicaciones (PHP).....	35
3.2.3.4.3.	Servidor de base de datos (MYSQL).	35
3.2.3.5.	Desarrollo del módulo de administrador de localización (Web).	35
3.2.3.6.	Desarrollo del módulo de Cliente REST (Web).....	36
3.2.4.	V001-Visualizar información en mapas.....	37
3.2.4.1.	Diseño de la interfaz (Móvil)	37
3.2.4.2.	Despliegue de mapas (Móvil)	38
3.2.4.3.	Diseño de la interfaz (Web).....	38
3.2.4.4.	Despliegue de mapas (Web)	38
3.2.5.	A002-Logueo al sistema utilizando cuenta de Google.	39
3.2.5.1.	Implementar Autenticación con Google (Móvil).	39
3.2.5.2.	Implementar Funcionalidad Almacenamiento (Móvil).	40
3.2.5.3.	Implementar Autenticación con Google (Web)	41
3.2.5.4.	Implementar Funcionalidad Almacenamiento (Web)	42
3.2.6.	A003- Compartir recursos en las redes sociales.....	43
3.2.6.1.	Implementar Funcionalidad Compartir en redes sociales (Móvil).....	43
3.2.6.2.	Implementar Funcionalidad Compartir en redes sociales (Web).....	44
3.2.6.3.	Desarrollo de manual de usuario.....	44
3.2.6.4.	Desarrollo de manual de programador.	44

3.3. Comentarios finales del capítulo.....	44
4. CAPÍTULO 4: PRUEBAS Y VALIDACIÓN	45
4.1. Pruebas.....	46
4.1.1. Pruebas Unitarias	46
4.1.2. Pruebas de Aceptación de usuario.	47
4.1.2.1. Pruebas de Aceptación para consumo de información geo-referenciada. .	47
4.1.2.2. Pruebas de aceptación para almacenamiento y visualización de recursos Favoritos. 48	
4.1.3. Pruebas de Integración.....	49
4.1.4. Pruebas de Caja negra.	49
4.2. Validación.	50
4.2.1. Comparación de resultados a nivel de Latino América	50
4.2.2. Modelado y publicación de datos enlazados de prueba en servidor Local. ...	50
4.3. Comentarios finales del capítulo.....	52
CONCLUSIONES	53
RECOMENDACIONES	54
BIBLIOGRAFÍA.....	55
ANEXOS.....	57
Anexo 1. Diccionario de datos	58
Anexo 2. Historias de Usuario.....	60
Anexo 3. Product Backlog.....	63
Anexo 4. Sprint Backlog.....	64
Anexo 5.Estructura desagregada de Tareas en WBS Schedule Pro.....	65
Anexo 6. Diseño de la interfaz Móvil.	67
Anexo 7. Diseño de la interfaz Web.	68
Anexo 8. Pruebas de Aceptación de usuario.	69
Anexo 9. Pruebas de Integración.	72
Anexo 10. Métricas e indicadores analizados por SonarQube.	76
Anexo 11. Programa generador de tripletas.....	77
Anexo 12. Manual de usuario.	80
Anexo 13. Manual de programador.....	87

ÍNDICE DE FIGURAS

Figura 1. Estadísticas globales de Sistemas Operativos Móviles.....	6
Figura 2. Ejemplo de un objeto RDF.....	8
Figura 3. Estructura de un objeto JSON.....	9
Figura 4. Estructura de una consulta en SPARQL.....	10
Figura 5. Gráfico de la solución propuesta	21
Figura 6. Diagrama de secuencia del funcionamiento de las aplicaciones.....	22
Figura 7. Consulta en el servicio web para recursos de Loja en DBpedia	28
Figura 8. Modelo de base de datos	29
Figura 9. Arquitectura de la aplicación móvil	31
Figura 10. Métodos para la geo-localización	32
Figura 11. Obtención de coordenadas actuales en un dispositivo móvil ANDROID. . .	33
Figura 12. Métodos para comunicación con servicio web desde aplicación móvil.	34
Figura 13. Resultados de consulta de recursos cercanos a la ubicación actual a través de un dispositivo móvil.	34
Figura 14. Arquitectura de la Aplicación Web.	35
Figura 15 Métodos de geo-localización de Aplicación web.	36
Figura 16. Resultados de Geo-localización por internet	36
Figura 17. Método de comunicación con el servicio web aplicación Móvil.	37
Figura 18. Recursos cercanos al usuario desde aplicación web	37
Figura 19. Visualización de recursos en dispositivos móviles	38
Figura 20. Despliegue de mapas en aplicación web.	39
Figura 21. Autenticación de cliente OAUTH 2.0	40
Figura 22. Autenticación con cuenta de google (Móvil).....	40
Figura 23. Almacenamiento de recursos favoritos de usuario en aplicación móvil	41
Figura 24. Métodos para autenticación con cliente OAUTH 2.0 en aplicación web.	41
Figura 25. Autenticación con cuenta de google (Web)	42
Figura 26. Almacenamiento de recursos favoritos de usuario en navegador.	42
Figura 27. Compartir recursos utilizando las ventanas nativas de Android.....	43
Figura 28. Compartir recursos a través de dispositivo móvil.	43
Figura 29. Compartir recursos en redes sociales en navegador web	44
Figura 30. Resultados de análisis en SonarQube	50
Figura 33. Cantidad de recursos geo-referenciados por País obtenidos con la aplicación.	50
Figura 31. Modelo ontológico para la publicación de datos enlazados geo-referenciados.....	51
Figura 32. Visualización de instituciones educativas de nivel secundario en Loja Ecuador en aplicación web.	52

INDICE DE TABLAS

Tabla 1. Comparación entre servicios web REST y SOAP.....	7
Tabla 2. SPARQL Endpoints.....	10
Tabla 3. Funcionamiento de una aplicación orientada a servicios para consumo de información geo-referenciada.	14
Tabla 4. Características del buscador de datos enlazados por lenguaje natural.	15
Tabla 5: Funcionamiento de aplicación para mejorar el acceso a recursos Educativos abiertos.....	15
Tabla 6. Funcionamiento de una Aplicación Móvil para aprender geografía.	16
Tabla 7 Funcionamiento del framework DBpedia Mobile Explorer	16
Tabla 8. Funcionamiento de la aplicación móvil para publicidad basada en la geo-localización.	17
Tabla 9. Funcionamiento de DBpedia Viewer.	18
Tabla 10. Funcionamiento de la aplicación para búsqueda asociativa utilizando Tag clouds.	18
Tabla 11. Funcionamiento de MapXPlore	19
Tabla 12. Funcionamiento de HDTourist.....	20
Tabla 13. Comparación de trabajos relacionados al trabajo de fin de titulación.	20
Tabla 14. Características de SCRUM	22
Tabla 15. Consulta base para recursos geo-referenciados	26
Tabla 16. Filtros para consulta.....	26
Tabla 17. Descripción de parámetros del servicio web de consulta.	27
Tabla 18. Descripción de campos del servicio web de almacenamiento.	30
Tabla 19. Comparación de estrategias de geo-localización.	32
Tabla 20. Resultados de pruebas Unitarias.	46
Tabla 21. Pruebas de Aceptación para consumo de información geo-referenciada.	48
Tabla 22. Pruebas de aceptación para almacenamiento y visualización de recursos..	48
Tabla 23. Casos de Prueba de integración planteados.	49
Tabla 24. Clases utilizadas en el modelo para la publicación de datos enlazados geo-referenciados.	51
Tabla 25. Propiedades utilizadas en el modelo para la publicación de datos enlazados geo-referenciados.	51

RESUMEN

Actualmente el consumo de datos enlazados representa un problema para los usuarios comunes, dada la complejidad de las consultas expresadas en SPARQL así como las ontologías y fuentes de datos existentes; por lo tanto, el presente trabajo busca hacer el consumo de datos enlazados de una manera simple a través de interfaces sencillas de utilizar.

Haciendo uso de la metodología de desarrollo SCRUM fueron creados dos servicios web: Un servicio de consultas encargado de enmascarar la complejidad del lenguaje SPARQL y un servicio de almacenamiento para guardar recursos favoritos del usuario en una base de datos en la nube. Tanto los servicios de consulta y almacenamiento son consumidos desde interfaces sencillas, haciendo uso de mapas y coordenadas del dispositivo en una aplicación Web, y una aplicación Móvil para teléfonos inteligentes con sistema operativo ANDROID.

En conclusión, para promover el consumo y publicación de recursos geo-referenciados como datos enlazados, es necesario crear aplicaciones para que un usuario promedio pueda entender y utilizar con facilidad, además de modelos ontológicos para publicar información geo-referenciada.

PALABRAS CLAVE: Datos enlazados Abiertos, Geo-referenciación, RDF, servicios web REST, SPARQL.

ABSTRACT

Actually Linked Data consumption represents a trouble for normal users due to the complexity of SPARQL queries, ontologies and datasets. The present work looks up to make Linked Data consumption in a simple way using simple interfaces.

Using SCRUM development methodology two web services have been created: One query service that hides SPARQL complexity and storage service to save favorite resources of users on a cloud database. Both Query service and Storage service are being consumed by simple interfaces, using maps and device coordinates, on a web application and a mobile application for ANDROID Operative System.

To conclude in order to promote publication and consumption of geo-referenced resources as linked data, is necessary to create applications that a normal user can easily understand and use, also give ontological models for geo-referenced data publication.

KEYWORDS: Linked open data, Georeferencing, RDF, REST Web services, SPARQL.

INTRODUCCIÓN

Desde las bases de datos en archivos, hasta las bases de datos relacionales y no relacionales; los datos siempre han tenido un papel fundamental para todas y cada una de las aplicaciones, generándose millones de datos por segundo en todo el mundo.

La creciente cantidad de datos enlazados generados principalmente en la web y todas las relaciones que existen entre las mismas, hace que sea imprescindible el consumirlos a través de las diversas plataformas disponibles actualmente.

Las plataformas móviles se destacan por características como: portabilidad, seguridad y ubicuidad como medio para consumir datos de la web, siendo ANDROID uno de los sistemas operativos móviles más populares con alrededor del 83% del mercado mundial desde el 2013.

Entre toda la información disponible y abierta en la web sobresale DBpedia como una base de datos enlazados abiertos completa en formato RDF, sin embargo esta información aún no es explotada en todo su potencial, actualmente los datos de DBpedia son consumidos a través de terminales donde se ejecutan complejas consultas escritas en SPARQL un lenguaje de consulta para RDF sobre Datasets, lo que provoca que los usuarios comunes les sea muy difícil consumir datos enlazados.

El presente trabajo de fin de titulación propone una solución para simplificar la búsqueda de información geo-referenciada en fuentes de datos enlazados abiertos, utilizando servicios web: para ocultar la dificultad de las consultas, almacenar y consumir información desde un dispositivo móvil Android o un navegador; mejorando la relevancia de los resultados utilizando estrategias como Geo-Localización y filtros.

El trabajo se encuentra dividido en 5 capítulos:

Marco Teórico: Describe los conceptos fundamentales necesarios para el desarrollo de esta aplicación, así como los estándares, librerías y APIs necesarias.

Propuesta de solución: Realiza un análisis de la problemática actual, haciendo énfasis en los trabajos relacionados y formulando una posible solución.

Desarrollo de la solución: Comprende el desarrollo de las aplicaciones y servicios web, dividida en fases correspondientes según la metodología SCRUM.

Pruebas y validación: Correspondiente a las pruebas realizadas al sistema para asegurar que las aplicaciones cumplen con la funcionalidad esperada.

Conclusiones y recomendaciones: Contiene las conclusiones y recomendaciones del proyecto.

CAPÍTULO 1: MARCO TEÓRICO

El presente capítulo tiene por objetivo definir los temas necesarios para el desarrollo del presente trabajo de fin de titulación, este capítulo está dividido en 4 secciones:

1.1 Conceptos Fundamentales: Son representaciones del conocimiento, necesario para entender el contexto del trabajo de fin de titulación.

1.2 Estándares: Corresponden a patrones, modelos, referencias o buenas prácticas utilizadas para el desarrollo del trabajo.

1.3 Lenguajes de programación: Define brevemente los lenguajes utilizados para la codificación y consultas.

1.4 Librerías frameworks y APIs: Corresponden a los códigos y programas desarrollados por terceros que facilitan la implementación de alguna característica necesaria.

1.5 Comentarios finales del capítulo: Es un breve resumen de los resultados obtenidos durante este capítulo.

1.1. Conceptos Fundamentales.

1.1.1. Web Semántica.

“La Web Semántica es 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”. (World Wide Consortium, 2016), la Web semántica está repleta de una cantidad creciente de datos RDF (Piedra, Chicaiza, Lopez, Tovar, & Martinez-Bonastre, 2012) los cuales entregan información descriptiva y estructurada sobre cada uno de los recursos en la web.

Como resultado de dotar de más significado a la información se pueden realizar consultas cada vez más complejas, que puedan resolver problemas cotidianos de una manera muy sencilla, y que esta información pueda ser intercambiada o procesada fácilmente.

1.1.2. DBpedia.

Es una base de datos que extrae información estructurada y multilenguaje desde Wikipedia y transformarlas en una base de datos de conocimiento; haciéndola disponible de manera gratuita por internet usando Web Semántica y Tecnologías de Linked Data, los artículos de Wikipedia en su mayoría constan de párrafos, cuadros de información e imágenes, pero también ofrecen otros tipos de información como RDF y JSON. (Bizer, Lehmann, Isele, & Jakob, 2012)

1.1.2.1. Ontologías de DBpedia.

Una ontología en sí se considera como una definición de tipos, propiedades y relaciones de las entidades que existen para un concepto; DBpedia consiste en 685 clases formando una jerarquía y con 2 795 propiedades descriptivas.

La importancia de las ontologías radica en como a través de las propiedades definidas para cada recurso se puede obtener la información deseada.

1.1.3. Android.

“Android es un software para dispositivos móviles basado en Linux que incluye sistema operativo, middleware y aplicaciones clave como: cliente de correo electrónico, programa de mensajería SMS, calendario, mapas, navegador, Gps ”.(Saha, 2008) , se puede ver a android como una pila en donde las aplicaciones están en la capa superior y el kernel de Linux en la inferior, por medio de los dispositivos móviles cada vez es más fácil que cualquier persona consuma una gran cantidad de servicios y también que aporte con mucha información a repositorios en la nube desde cualquier lugar y momento.

La Figura 1 muestra las estadísticas globales de ventas de sistemas operativos móviles desde el 2013 teniendo a ANDROID como líder con alrededor del 83% del mercado.

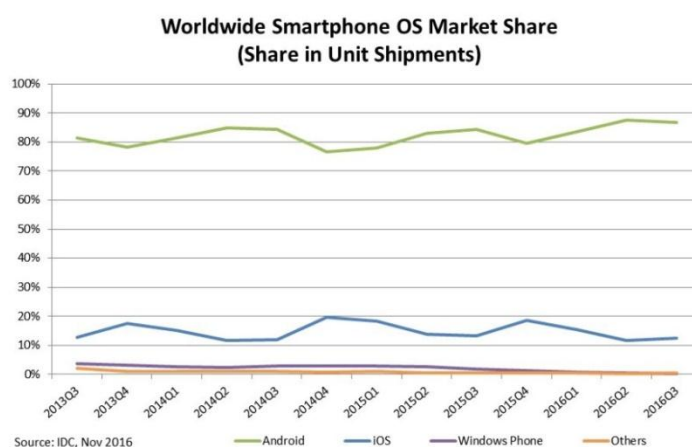


Figura 1. Estadísticas globales de Sistemas Operativos Móviles.

Fuente: (International Data Corporation, 2016)

Elaboración: (International Data Corporation, 2016)

1.1.4. Servicios Web

Un servicio web se define como un conjunto de aplicaciones o de tecnologías con capacidad para inter-operar en la Web, los cuales intercambian datos entre sí, ofreciendo un servicio a través de internet. Su principal función es proporcionar mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario (World Wide Web Consortium (W3c) España, 2016).

1.1.4.1. Protocolo Simple de Acceso a Objetos (SOAP)

Se considera como un paradigma para el intercambio de mensajes de un solo sentido, sin estado, el cual no interfiere en la semántica de cualquiera de los datos específicos de la aplicación que se comunica, ni mucho menos enrutamientos de mensajes, transferencias de datos fiables, y demás. World Wide Web Consortium W3c España,(2016)

Los mensajes SOAP son una transmisión unidireccional entre los nodos, desde un emisor hasta un receptor, es esperado que los mensajes de SOAP sean combinados

por las aplicaciones de tal manera que se puedan implementar patrones de interacción.

1.1.4.2. Transferencia de estado representacional (REST)

Es un patrón de operaciones de recursos que se transformó en el estándar para el diseño de servicios en las aplicaciones Web, mientras que SOAP utiliza un control remoto completo de objetos, con invocación de métodos remotos y encapsulando la funcionalidad, REST solo interactúa con las estructuras de datos y la transferencia de sus estados, Los recursos en un servicio REST son identificados y resueltos con una URL. (Battle & Benson,2008).

1.1.4.3. Comparación REST y SOAP

La Tabla 1 muestra una comparación entre los servicios web REST y SOAP haciendo énfasis en sus características más notables así como las ventajas y desventajas de su implementación.

Tabla 1. Comparación entre servicios web REST y SOAP.

	REST	SOAP
Características	Operaciones definidas en mensajes.	Operaciones definidas como puertos WSDL (Web Services Description Language)
	Operaciones estándar para cada objeto.	Instancias múltiples del proceso que realiza una misma operación.
	Dirección única para cada instancia del proceso.	Dirección única para todas las operaciones.
Ventajas	Fácil de construir y adaptar	Fácil de utilizar.
	Cliente no necesita información de enrutamiento a partir de la URL inicial.	Operaciones complejas escondidas detrás de una fachada.
	Bajo consumo de recursos.	Incrementa la privacidad.
Desventajas	Gran número de objetos.	Clientes deben saber operaciones y semántica.
	Descripción sintáctica/semántica orientada al usuario.	Son necesarios varios puertos para diferentes tipos de notificaciones.

Fuente: El Autor

Elaboración: El Autor.

Dadas las necesidades del presente proyecto se determinó que REST sería la opción más adecuada para la implementación de los servicios web, debido a la facilidad con la que pueden ser adaptados, ofrecen escalabilidad para el proyecto lo que conlleva la posibilidad de agregar nuevas funciones y una fácil invocación desde diferentes plataformas como Web o móvil, también aligera la carga en el servidor a consecuencia que utiliza menos recursos que SOAP.

1.2. Estándares.

1.2.1. Linked Data.

Linked Data se ha establecido a sí misma como un estándar para la publicación de datos estructurados por toda la web disfrutando del increíble crecimiento en términos

del número de organizaciones confirmadas que usan sus mejores prácticas y tecnologías para exponer y relacionar los conjuntos de datos para su intercambio, integración y reutilización (Esther et al., 2014).

Linked Data tiene como meta relacionar y vincular toda clase de conjuntos de datos para ser operados de tal manera que se puedan obtener resultados de búsquedas más precisos y ser consumidos por las aplicaciones que lo requieran.

Los cantidad de recursos de Linked Data en la web han estado incrementando; pero aún hay muy pocas maneras efectivas de presentarlas a los usuarios.(Vagliano, Marengo, & Morisio, 2015), dado que Linked Data utiliza principalmente SPARQL para realizar las consultas y RDF como formato para los datos al referenciar una URI¹ muy pocas veces se obtiene una representación intuitiva del recurso; eso es debido a la complejidad de las consultas, formato de los datos así como la enorme cantidad de propiedades y nodos relacionados al mismo.

1.2.2. Marco de descripción de recursos (RDF).

Es el formato de datos estándar para representación de recursos, “transformar todos las clases de datos a RDF parece ser una tendencia moderna” (Gongolidis, Kommata, Papantoniou, & Loumos, 2012) , conforme más va ganando importancia el uso de RDF como un formato para representar los diversos datos, se ha visto también que las aplicaciones que se encargan de consumir y alimentar información en este formato han ganado popularidad también.

La Figura 2 muestra la estructura de un objeto RDF.

```
<rdf:RDF xmlns:res="http://www.w3.org/2005/sparql-results#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
<rdf:Description rdf:nodeID="rset">
<rdf:type rdf:resource="http://www.w3.org/2005/sparql-results#ResultSet" />
  <res:resultVariable>name</res:resultVariable>
  <res:resultVariable>abstract</res:resultVariable>
  <res:solution rdf:nodeID="r0">
    <res:binding rdf:nodeID="r0c0"><res:variable>name</res:variable><res:value
xml:lang="en">Central University of Ecuador</res:value></res:binding>
    <res:binding rdf:nodeID="r0c1"><res:variable>abstract</res:variable><res:value
xml:lang="es">La Universidad Central del Ecuador, es la universidad más antigua y la segunda
más grande por número de estudiantes de la República del Ecuador. Se ubica en el norte de la
ciudad de Quito, en la llamada ciudadela universitaria, además de comprender sus sedes en el
Sur de Quito, en la ciudad de Santo Domingo de los Colorados, y en las Islas Galápagos.
Afilada desde 2012 a la Red Ecuatoriana de Universidades para Investigación y Postgrados.
Sus orígenes se remontan a la Universidad Central de Quito. La cual se originó de la unión de
las Universidades: Seminario de San Luis y San Gregorio Magno fundada en 1651 por los
Jesuitas y la Santo Tomás de Aquino, fundada en 1681 por los Dominicos. Sobre la base de la
Real Universidad Pública Santo Tomás se fundó la Universidad Central de Quito. para en el año
1836, mediante decreto del presidente Vicente Rocafuerte se cambie la palabra Quito, por
Ecuador y surge ya de forma definitiva la Universidad Central del Ecuador (UCE).</res:value>
  </res:binding>
</res:solution>
</rdf:Description>
</rdf:RDF>
```

Figura 2. Ejemplo de un objeto RDF.

Fuente: El Autor.

Elaboración: El Autor

1.2.3. Notación de Objetos JavaScript (JSON).

“Es un formato de intercambio liviano, basado en el texto e independiente del lenguaje” (Bray, 2014), es utilizado ampliamente para la gestión de datos móviles en Android a diferencia de XML, YAML u otros.

¹ URI: Identificador de Recursos Uniforme.

Un texto JSON es una secuencia de tokens (componentes léxicos) los cuales incluyen caracteres, cadenas, números y otros tres nombres literales (True,False,Null).

La Figura 3 muestra la estructura de un objeto JSON básico.

```
{ "Persona":  
  { "Nombre": "Persona1",  
    "Apellido": "Apellido1",  
    "Edad": 20,  
    "Sexo": "M"  
    "Foto": {  
      "Url": "http://www.ejemplo.com/imagen/foto.jpg",  
      "Height": 200,  
      "Width": 300  
    },  
  }  
}
```

Figura 3. Estructura de un objeto JSON.

Fuente: El Autor

Elaboración: El Autor

1.3. Lenguajes

1.3.1. JAVA.

“Es un lenguaje de programación de propósito general, concurrente, orientado a objetos, basado en C y C++” (Gosling et al., 2014) , es considerado como uno de los lenguajes de programación más populares, especialmente para el desarrollo de aplicaciones tipo Cliente-servidor.

1.3.2. Preprocesador de Hipertexto (PHP).

“Es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML”, (The PHP Group, 2001) una de las características fundamentales de PHP es que el código se ejecuta del lado del servidor generando HTML y enviándolo al cliente, el cual recibirá un resultado después de ejecutar el Script, sin que tenga conocimiento acerca de cuál fue el código ejecutado.

1.3.3. JAVASCRIPT

Es el lenguaje interpretado más utilizado, principalmente en la construcción de páginas Web, con una sintaxis muy semejante a JAVA y a C; no se trata de un lenguaje orientado a objetos, sino que éste está basado en prototipos (Flanagan, 2007). Es utilizado del lado del cliente desde un navegador web también permite mejoras en las páginas o en la interfaz además el código JavaScript en una página web es interpretado por todos los navegadores modernos.

1.3.4. Protocolo y Lenguaje de Consulta RDF (SPARQL).

Se puede utilizar para expresar consultas que permiten interrogar diversas fuentes de datos. Los datos se almacenan de forma nativa como RDF o son definidos mediante vistas RDF a través de algún sistema middleware. SPARQL contiene las capacidades para la consulta de los patrones obligatorios y opcionales del grafo RDF junto con sus conjunciones y disyunciones.

También soporta la ampliación o restricciones del ámbito de las consultas indicando los grafos sobre los que se opera. Los resultados de las consultas SPARQL pueden ser conjuntos de resultados o grafos RDF.(World Wide Web Consortium, 2016)

La Figura 4 muestra la estructura de una consulta en SPARQL.

```
PREFIX prefijo: <URI>
SELECT ?Concepto WHERE {?Recurso prefijo:propiedad ?Concepto}
```

Figura 4. Estructura de una consulta en SPARQL.

Fuente: El Autor.

Elaboración: El Autor.

Las consultas de SPARQL contienen un conjunto de patrones de tripleta denominadas patrón de grafo básico similares a las tripletas RDF (World Wide Web Consortium, 2016); con la diferencia de que cada sujeto , predicado y objeto podrían ser variables.

1.3.4.1. SPARQL endpoints

“Los endpoints pueden ser definidos como servicios Web destinados a la recepción de peticiones para realizar una posterior devolución de datos”.(Quezada, 2016), el objetivo de los endpoints es ofrecer al usuario una manera fácil de consumir información de los diversos triplestores² desde una URI otorgada por el administrador de la misma.

La Tabla 2 contiene un resumen de los endpoints más importantes y los enlaces para acceder a ellos.

Tabla 2. SPARQL Endpoints.

ENDPOINT	ENLACE
DBpedia	http://dbpedia.org/sparql
DBpedia español	http://es.dbpedia.org/sparql
DBpedia Latinoamérica	http://es-la.dbpedia.org/sparql
EEA (European Environment Agency Semantic Data Service)	http://semantic.eea.europa.eu/sparql
Openlink Virtuoso	http://demo.openlinksw.com/sparql

Fuente: (Quezada, 2016)

Elaboración: (Quezada, 2016)

1.4. Librerías, frameworks³ e Interfaces de programación de Aplicaciones (API)

1.4.1. VOLLEY

Volley es una librería HTTP que hace fácil y rápido el manejo de redes para las aplicaciones ANDROID, siendo disponible gracias a un repositorio AOSP⁴ (Android Developers, 2016).

² Triplestores: Bases de datos que almacenan tripletas de información.

³ Framework: Corresponden a conjuntos de herramientas preestablecidas para el desarrollo de actividades.

⁴ AOSP: Proyecto ANDROID de Código Abierto.

Sobresale en operaciones de tipo RPC⁵, como medio para obtener información estructurada desde una página web; se puede integrar fácilmente con cualquier protocolo e incluye soporte para cadenas de texto, imágenes y datos JSON. Libera al programador para concentrarse en la lógica específica de la aplicación.

1.4.2. Jenna.

Es un framework para web semántica de código abierto disponible para el lenguaje JAVA, provee una API para extraer y escribir data hacia grafos RDF, dichos grafos son representados como un modelo abstracto que pueden provenir de diferentes archivos, bases de datos, urls, etc.

Adicionalmente Jenna soporta varios formatos de serialización para grafos RDF como:

- RDF/XML: Utilizando XML, la información RDF puede ser fácilmente intercambiada entre diferentes tipos de sistemas operativos y lenguajes de aplicaciones.
- Turtle (Terse RDF Triple Language): Agrupa tres URIS para construir tripletas de información.
- Notation 3: Es una formato que no utiliza XML, diseñado para que sea fácil de entender por humanos.

1.4.3. CodeIgniter.

“Es un poderoso framework para PHP incluye un conjunto de clases refactorizadas, elaborado para ser un simple y elegante kit de herramientas ”.(EllisLab, 2017), divide la funcionalidad específica de la aplicación, para el desarrollo rápido de sitios y aplicaciones web.

1.4.4. Google Maps API.

Con el API de Google Maps se pueden agregar mapas basados en datos de Google Maps para una aplicación, esta API administra automáticamente el acceso a servidores , descargas de datos, visualización de mapas, así como agregar marcadores, polígonos y superposiciones en un mapa básico (Google Developers, 2016)

1.5. Comentarios finales de capítulo.

Ahora que se tiene un dominio de los conceptos fundamentales, estándares y las herramientas necesarias; se procede con el diseño de una solución que permita un fácil consumo de información recuperando las propiedades más relevantes desde la ontología de DBpedia en un formato JSON debido a que es fácil de operar por lenguajes como: JAVA, PHP y JavaScript; aprovechando el API de Google Maps para presentar la información de una manera comprensible al usuario.

⁵ RPC: Llamada a procesos remotos.

CAPÍTULO 2: PROPUESTA DE SOLUCIÓN

En este capítulo se define el contexto real del problema, haciendo un adecuado análisis de la situación actual para identificar la problemática y ofrecer una solución que facilite el consumo almacenamiento y visualización de datos enlazados geo-referenciados.

Este capítulo se encuentra dividido en 3 secciones:

2.1 Problemática Actual: En esta sección se contextualiza la situación actual, explicando el problema real para justificar el desarrollo del trabajo.

2.2 Análisis de trabajo relacionado: Corresponde al análisis de las soluciones actuales para la problemática, realizando una comparación a las mismas y contrastándolas con las características de la solución presentada en el trabajo.

2.2 Solución propuesta: Es una descripción detallada de la solución planteada.

2.4 Comentarios finales del capítulo: Hace un breve resumen de los resultados obtenidos durante el capítulo.

2.1. Problemática Actual.

Existe una gran complejidad al momento de consumir datos enlazados abiertos, debido a la necesidad de escribir consultas en SPARQL sobre un determinado endpoint, además de que esta información es presentada como cadenas de texto en algún formato de serialización como: RDF/XML o Turtle; obteniendo como resultado que muchas veces no sean fáciles de entender, volviéndose poco interesante o tedioso el consumo de datos enlazados por un usuario común.

2.2. Análisis de trabajos relacionados.

2.2.1. Semantic driven location services in the Web of Data

Este trabajo presenta una aplicación orientada a servicios de localización, capaz de obtener datos relevantes a las preferencias y posición geográfica del usuario desde la DBpedia, para ello se hace uso de Google latitude para localizar al usuario, mientras el API de SPARQL es utilizado para realizar búsquedas en DBpedia de acuerdo a los intereses de los usuarios, los grafos RDF resultantes son serializados en un formato JSON y visualizados a través de Google Maps.(Gongolidis et al., 2012).

La Tabla 3 muestra las fases que realiza la aplicación de consumo de información geo-referenciada para realizar la búsqueda y mostrar los resultados al usuario.

Tabla 3. Funcionamiento de una aplicación orientada a servicios para consumo de información geo-referenciada.

Fases	Descripción
Entrar en el rango de búsqueda de Google Latitude	El usuario debe activar su distintivo de Google Latitude y definir el radio de búsqueda.
Identificar la posición y el país	Una vez activado el distintivo, la localización del usuario es obtenida con precisión y los datos son transmitidos serializados en un formato JSON.
Identificar las categorías de interés para el país.	La localización del país se vuelve un valor inicial para realizar una búsqueda sobre DBpedia y obtener las categorías de interés.
Seleccionar una categoría	Las categorías obtenidas son enviadas en formato JSON y se despliegan ante el usuario para luego seleccionar una categoría.
Obtener información relevante.	Se realiza una búsqueda específica sobre DBpedia, y se obtienen todos los recursos pertenecientes a la categoría seleccionada.
Visualizar	Basado en el código JavaScript que utiliza el API de Google Maps V3, los recursos obtenidos son visualizados, agregando un marcador en la posición de cada uno sobre un mapa.

Fuente: (Gongolidis et al., 2012)

Elaboración: El Autor.

2.2.2. Investigación y análisis de aplicaciones, para proponer un mejor consumo y visualización de datos enlazados.

Este trabajo fue elaborado “Con el objetivo de mejorar el consumo y visualización de datos enlazados en la web, de tal manera que cualquier usuario con o sin ningún tipo de conocimiento del tema, pueda explorar bases de datos semánticas” (Quezada, 2016). Para ello propone la construcción de un software con una arquitectura en tres capas que funciona a manera de buscador transformando el lenguaje natural a una consulta SPARQL, para luego mostrar estos resultados utilizando un grafo.

La Tabla 4 muestra las características principales del buscador semántico de datos enlazados.

Tabla 4. Características del buscador de datos enlazados por lenguaje natural.

Características	Descripción
Buscador.	Integrado con múltiples herramientas para mejorar la precisión de los datos como: Filtros, Origen de los datos y lenguajes.
Consumo de recursos.	Hace consumo de los recursos disponibles en una base de datos semántica en alguno de los DataSets definidos con la posibilidad de agregar nuevos.
Visualización.	Despliega una interfaz: usable, accesible con HTML5, CSS3, JQuery y JavaScript.

Fuente: (Quezada, 2016)

Elaboración: El Autor.

2.2.3. Combining Linked Data and Mobiles Devices to improve access to OCW

En este trabajo se propone una arquitectura cliente servidor de 3 capas para dispositivos ANDROID, para el consumo de los datos enlazados de los recursos educativos abiertos con los beneficios de los dispositivos móviles a través una interfaz para administrar la información en RDF. (Piedra et al., 2012).

La Tabla 5 muestra cada una de las fases que realiza la aplicación para el consumo de información sobre recursos educativos abiertos y su posterior visualización.

Tabla 5: Funcionamiento de aplicación para mejorar el acceso a recursos Educativos abiertos.

Fases	Descripción
Recolectar entradas de usuario a través de la interfaz.	La vista provee de clases de interfaz básicas para gestionar las ventanas y las interacciones con el usuario de OCW ⁶ .
Controlador escucha los eventos del dispositivo móvil.	Un controlador escucha las entradas del usuario en una vista y decide cómo proceder, ejecutando o seleccionando una vista diferente.
Controlador interpreta las entradas del usuario, actualiza el modelo y envía mensajes de vuelta a la vista.	El controlador es responsable de administrar la lógica de las entradas y escucha al modelo por algunos eventos para decidir las acciones a tomar.
El proveedor de contenido puede invocar un servicio web RESTfull.	En esta fase se maneja la interacción entre los terminales SPARQL y los servicios web RESTfull
El proveedor de datos notifica la vista y el controlador.	Convertir la información RDF en una representación compacta de tal manera que la información sea más fácil de entender por el usuario.
La vista es actualizada.	Se utiliza un objeto vista para manejar las propiedades de las vistas generadas.

Fuente: (Piedra et al., 2012)

Elaboración: El Autor.

⁶ OCW: Recursos Educativos Abiertos.

2.2.4. Linked Data-driven Geographic Mobile learning Application.

En este trabajo se muestra el desarrollo de una aplicación móvil para aprender geografía a partir de datos enlazados como fuente de datos, utilizando DBpedia y LinkedGeoData para mejorar la cantidad de información con el mínimo esfuerzo.(Hendrik & Hendratmo, 2015).

La Tabla 6 se muestra de una manera más detallada el funcionamiento de la aplicación para aprender geografía.

Tabla 6. Funcionamiento de una Aplicación Móvil para aprender geografía.

Fases	Descripción
Entradas de usuario.	Un usuario puede realizar búsquedas personalizadas usando una búsqueda textual o seleccionando un en un mapa.
Recolectar información geográfica necesaria.	La información sobre los países es reunida al realizar consultas SPARQL sobre DBpedia.
Vista detallada.	Al seleccionar un marcador de algún país se muestra información detallada del mismo al realizar una consulta sobre Linkedgeodata dataset y dando formato al resultado.

Fuente: (Hendrik & Hendratmo, 2015)

Elaboración: El Autor.

2.2.5. DBpedia Mobile Explorer.

Este trabajo presenta un framework que permite la visualización de datos enlazados para los ambientes móviles, escondiendo la complejidad de RDF y SPARQL a los usuarios de tal manera que pueda ser utilizado por usuarios que no tengan conocimiento técnico. (Vagliano et al., 2015).

La Tabla 7 muestra las fases que realiza el framework DBpedia Mobile explorer para el consumo de datos enlazados.

Tabla 7 Funcionamiento del framework DBpedia Mobile Explorer

Fases	Descripción
Selección de recurso.	Un usuario ingresa el recurso que quiere visualizar, se ejecuta una consulta describe sobre dbpedia y entrega los resultados.
Categorías del recurso.	Se despliega la información correspondiente a todas las características del usuario.
Selección de categoría	Al seleccionar una categoría del recurso se ejecuta una consulta select sobre DBpedia que extrae la información necesaria y la despliega al usuario.

Fuente:(Vagliano et al., 2015)

Elaboración: El Autor.

2.2.6. Mobile Advertising using Location Based Services.

En este trabajo se propone un sistema basado en las características principales de los dispositivos móviles como ubicuidad e interactividad, facilitando la comunicación entre suscriptores y publicistas. Haciendo uso de la localización del suscriptor se puede enviar anuncios personalizados a través de un post HTTP y recibiendo un mensaje en su teléfono con el anuncio.(Shoaibi & Rassan, 2012)

La Tabla 8 detalla el funcionamiento de esta aplicación:

Tabla 8. Funcionamiento de la aplicación móvil para publicidad basada en la geo-localización.

Fases	Descripción
Publicar anuncio.	Un publicista a través de un navegador web publica su anuncio el cuál debe tener alguna referencia geográfica.
Suscribirse al servicio.	El suscriptor descarga la aplicación para poder hacer uso del servicio, registrando su información así como su número de teléfono
Algoritmo de ranqueo.	El servidor envía el anuncio al suscriptor en función de la cercanía que tiene el usuario a la posición registrada del anuncio.
Recepción de anuncio.	Un SMS es enviado al teléfono del suscriptor.

Fuente:(Shoaibi & Rassan, 2012)

Elaboración: El Autor.

2.2.7. DBpedia Viewer - An Integrative Interface for DBpedia Leveraging the DBpedia Service Eco System.

DBpedia Viewer es una nueva interfaz de usuario, que permite presentar información desde DBpedia, integrando sus servicios existentes, así como herramientas externas de visualización para hacerla amigable al usuario, mostrando información desde los datasets de DBpedia en un formato comprensible y ofreciendo algunos servicios relevantes, lo que distingue a DBpedia Viewer es su framework de triple acción, que permite dinámicamente asociar tipos de acciones con tripletas.(Lukovnikov, Stadler, Kontokostas, Hellmann, & Lehmann, 2014).

La Tabla 9 muestra en mayor detalle la funcionalidad de DBpedia Viewer.

Tabla 9. Funcionamiento de DBpedia Viewer.

Fases	Descripción
Búsqueda	Implementa autocompletado utilizando el servicio DBpedia lookup, permitiendo búsquedas de cadenas de texto y ofreciendo filtros adicionales como lenguaje.
Filtro de tripletas	Las tripletas pueden ser filtradas haciendo uso de sus propiedades o valores.
Vistas	Al poner el ratón sobre entidades muestra una imagen de las mismas, y sobre propiedades muestra una breve descripción de la misma.
Mapas	Se usa openStreetMap para visualizar aquellas entidades que posean latitud y longitud.

Fuente: (Lukovnikov et al., 2014)

Elaboración: El Autor.

2.2.8. Mobile Location-Driven Associative Search in DBpedia with Tag Clouds.

En este trabajo se expone una alternativa para consumir información de la DBpedia basada en la localización y utilizando conjuntos de nubes de etiquetas basadas en los recursos o entidades de DBpedia para ofrecer al usuario una fácil interacción con la información y la relevancia de los resultados variando el tamaño del texto de las etiquetas. El tamaño de las etiquetas es calculado en función de la cantidad de veces que aparece un resultado en la consulta. (Tessem, Johansen, & Veres, n.d.).

La Tabla 10 muestra las fases que realiza la aplicación de búsqueda asociativa para obtener la información y crear las tag clouds⁷.

Tabla 10. Funcionamiento de la aplicación para búsqueda asociativa utilizando Tag clouds.

Fases	Descripción
Posición del usuario	Determina las coordenadas del usuario.
Búsqueda primaria.	Obtener todos los recursos primarios dentro del radio de la posición.
Búsqueda secundaria.	Obtener todos los artículos secundarios relacionados a los artículos primarios.
Frecuencias de palabra	Busca la frecuencia de una palabra en los abstractos del artículo.
Determinar peso de palabra	Determina el peso de la palabra con una escala de 1.0 cercano al usuario y 0.0 dentro del radio.
Crear Tag clouds.	Crea las Tag clouds con los valores más altos en la pantalla del dispositivo.

Fuente: (Tessem et al., n.d.)

Elaboración: El Autor.

⁷ Tag Cloud: Nubes de etiquetas.

2.2.9. MapXplore: Linked Data on the appstore

Propone una aplicación nativa de iPad amigable que intenta introducir a los usuarios al mundo de Linked Data, haciendo transparente todo lo referente a SPARQL, utiliza la localización actual del usuario para obtener recursos cercanos a su posición desde DBpedia, implementa adicionalmente Google Analytics para llevar un seguimiento de las funciones que no sean utilizadas.(Veres, 2012),

La Tabla 11 muestra en mayor detalle el funcionamiento de la aplicación MapXplore.

Tabla 11. Funcionamiento de MapXPlore

Fases	Descripción
Determinar posición del usuario.	Determina la posición del usuario utilizando el GPS del dispositivo.
Puntos de interés	Realiza una consulta sobre DBpedia para obtener todos los puntos de interés cercanos a la localización del usuario y despliega marcadores sobre estos.
Visualización	Usuario puede seleccionar ver la imagen o la información correspondiente al recurso.
Imágenes	Utiliza servicios basados en localización (Google Street View, Google Maps, flickr) o semánticos (OOKaboo), para obtener imágenes relevantes al recurso.
Información	Consulta información correspondiente al recurso en diversos DataSets como: DBpedia, SIndex y FreeBase.

Fuente: (Veres, 2012)

Elaboración: El Autor.

2.2.10. HDTourist: Exploring Urban Data on Android Elena

HDTourist es una aplicación para Android que busca ayudar a los turistas que visiten una ciudad desplegando información sobre la misma, manteniendo datos de consultas semánticas sobre una URI correspondiente a la ciudad y almacenando los resultados en el teléfono para ser consumidas sin necesidad de internet.(Hervalejo, Martínez-Prieto, Fernández, & Corcho, 2014).

La Tabla 12 muestra los procesos fundamentales que realiza HDtourist para la búsqueda y despliegue de la información.

Tabla 12. Funcionamiento de HDTourist.

Fase	Descripción
Obtener datos urbanos desde DBpedia	Se crea una consulta en DBpedia que obtiene todas las tripletas que describen la ciudad, o que se relacionan con el cliente.
Buscar entre información urbana	La información se clasifica en categorías: Descripción, Demografía, Geografía, atracciones y otros.

Fuente: (Hervalejo et al., 2014)

Elaboración: El Autor.

2.2.11. Comparación de trabajos relacionados.

En la Tabla 13 se realiza una comparación de trabajos relacionados al trabajo de fin de titulación; con el fin contrastar las características de cada trabajo relacionado y elegir las características más útiles para el presente trabajo de fin de titulación.

Tabla 13. Comparación de trabajos relacionados al trabajo de titulación.

Trabajos relacionados	Geo localización	Arquitectura	Plataforma	Formato de respuesta	Interfaz
Semantic driven location services in the Web of Data	Google Latitude	Cliente-servidor	Web	JSON	Mapas
Investigación y análisis de aplicaciones, para proponer un mejor consumo y visualización de datos enlazados.	Ninguna	3 capas	Web	RDF	Páginas web
Combinando Linked Data y dispositivos móviles para mejorar el acceso a OCW.	Sin definir	Sin definir	Móvil (Android)	RDF	Mapas
Geographic Mobile learning Application.	Ninguna	Capas	Móvil (Android)	JSON	Mapas
DBpedia Mobile Explorer.	Ninguna	Sin definir	Móvil (Android)	RDF	Vistas
Mobile Advertising using Location Based Services	GPS Network	Cliente-servidor	Móvil Android) Web	No usa DBpedia	SMS
DBpedia Viewer	Ninguna	Cliente-servidor	Web	RDF	Páginas web
DBpedia with Tag Clouds.	Sin definir	Cliente-servidor	Móvil (Android)	Número de resultados	Tag Clouds
MapXplore	GPS	Capas	Móvil (IOS)	RDF	Mapas
HDTourist	GPS	Capas	Móvil (Android)	RDF	Mapas
Presente Trabajo	Network	Capas Cliente-servidor	Móvil (Android) Web	JSON	Mapas Páginas web

Fuente: El Autor.

Elaboración: El Autor.

2.3. Solución Propuesta

2.3.1. Objetivo general.

Realizar el procesamiento de los datos semánticos de internet, como DBpedia y procesarlos en los dispositivos móviles Android. Con la ayuda de la geo localización mejorar la visualización de datos de los internautas que consuman datos enlazados.

2.3.2. Objetivos Específicos.

- Modelar una arquitectura que permita el consumo eficiente de datos enlazados abiertos en plataformas móvil y web.
- Diseñar una interfaz móvil para hacer fácil el consumo, visualización y almacenamiento de datos enlazados geo referenciados.
- Diseñar una interfaz web para hacer fácil el consumo, visualización y almacenamiento de datos enlazados geo referenciados.

2.3.3. Descripción de la solución.

En base a los enfoques anteriores tenemos algunos factores a considerar como: la facilidad que tiene un usuario para consumir datos enlazados y la manera en como estos se despliegan; pensando en aquello se propone que el usuario pueda consumir esta información de tal manera que las consultas sean totalmente invisibles para él, utilizando la localización de su teléfono inteligente para obtener toda la información relevante desde la DBpedia y que esta información sea desplegada a manera de marcadores en un mapa, adicionalmente se podrá consumir y visualizar la misma información desde un navegador web.

La Figura 5 muestra una representación gráfica del funcionamiento de las aplicaciones web y móvil.

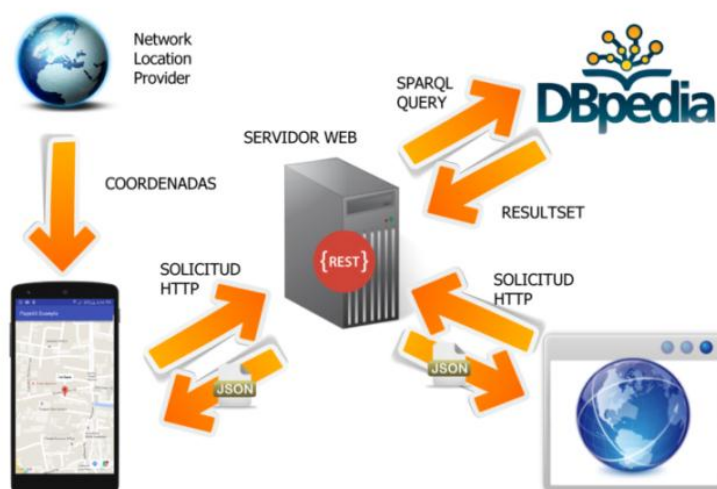


Figura 5. Gráfico de la solución propuesta

Fuente: El Autor.

Elaboración: El Autor.

La Figura 6 muestra un diagrama de secuencia para explicar el funcionamiento de la solución propuesta.

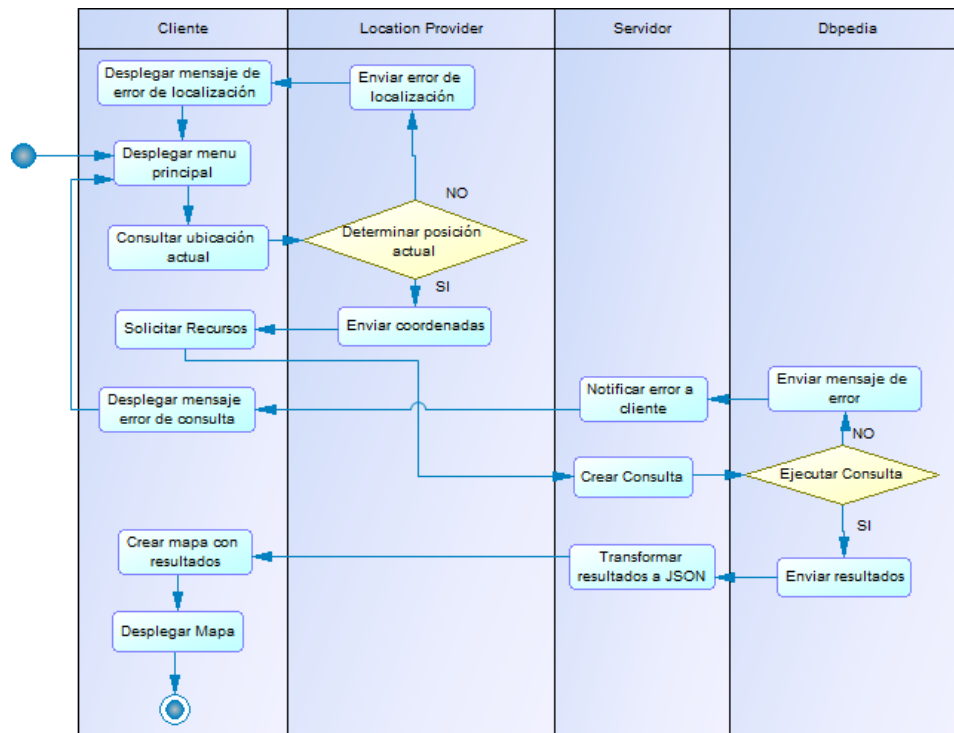


Figura 6. Diagrama de secuencia del funcionamiento de las aplicaciones.

Fuente: El Autor.

Elaboración: El Autor.

2.3.4. Metodología de desarrollo.

2.3.4.1. SCRUM

SCRUM es un framework⁸ en donde las personas pueden resolver problemas adaptativos complejos, mientras entregan productos del más alto valor posible. (Ken Schwaber & Sutherland, 2013), el enfoque de SCRUM asume que los procesos de análisis, diseño y desarrollo en cada fase de Sprint⁹ son impredecibles. Utiliza un enfoque iterativo e incremental para optimizar lo predecible y controlar el riesgo.

La Tabla 14 muestra las principales características de la metodología SCRUM.

Tabla 14. Características de SCRUM

Características de SCRUM	
La fase de Sprints es un proceso empírico	Muchos de los procesos en la fase de Sprints son indefinidos, son tratados como una caja negra que requiere de control externo.
Los Sprints son flexibles	Los Sprints son usados para construir el producto final.
El proyecto está abierto hasta la fase de cierre.	Los entregables pueden ser cambiados durante las fases de planeación y Sprints.
Entregables	Son determinados durante el proyecto basado en el entorno.

Fuente: (K. Schwaber, n.d.)

Elaboración: El Autor.

⁸ Framework: Conjunto de herramientas, módulos y estándares para agilizar el desarrollo de software.

⁹ Sprint: Es una iteración corta donde se entrega una pieza funcional del producto.

2.3.4.2. Pre-Game

2.3.4.2.1. Planeación.

La fase de planeación consiste de la conceptualización, análisis y definición del alcance del proyecto. Utilizando las “historias de usuario” se determinan las características de la aplicación y luego se estima el tiempo de desarrollo para cada característica, estas estimaciones son básicas y por lo general van a variar cuando se analice cada iteración a profundidad, los resultados de esta fase son:

- **Historias de usuario:** Es una lista de requerimientos y criterios de aceptación obtenida del cliente.
- **Product Backlog:** Es una lista de objetivos priorizada que incluyen las características de la aplicación a desarrollar.
- **Sprint Backlog:** Es una lista ordenada de los Sprints con el estado actual y el tiempo estimado en días de cada sprint.

2.3.4.3. Game

2.3.4.3.1. Desarrollo de Sprints.

Esta fase corresponde al desarrollo de una funcionalidad; con constante respeto a las variables como: tiempo, requisitos, calidad y costo. Existen múltiples Sprints iterativos que son utilizados para construir el sistema.

El proceso para el desarrollo de cada Sprint es el siguiente:

- **Desarrollar:** Definir qué se debe hacer para implementar los requerimientos del product backlog en paquetes.
- **Envolver:** Cerrar los paquetes crear una versión ejecutable y observar como implementan los requerimientos.
- **Revisar:** Se producen reuniones para presentar el estado actual y revisar el progreso, resolviendo problemas o añadiendo nuevos ítems al backlog.
- **Ajustar:** Consolidar la información reunida en los paquetes afectados.

2.3.4.4. Post-Game

2.3.4.4.1. Cierre

Corresponde a la fase final donde prepara el producto desarrollado para la entrega al cliente, para ello se realizan pruebas sobre todo el sistema a fin de asegurar que el producto tenga calidad y cumpla con las expectativas del cliente.

2.4. Comentarios finales del capítulo.

Como resultado de este capítulo se tiene una propuesta de solución la cual consta de servicios web para realizar consultas a datasets como DBpedia, transformando los resultados a JSON y ser consumidos desde las aplicaciones web o móvil. Para el desarrollo de las aplicaciones se utilizará la metodología SCRUM debido a su alta tolerancia a los cambios sumada con una estructura desagregada de tareas para estimar, controlar y medir el avance del proyecto.

CAPÍTULO 3: DESARROLLO DE LA SOLUCIÓN

Este capítulo aborda el desarrollo de la solución definida en la sección CAPÍTULO 2: PROPUESTA DE SOLUCIÓN, utilizando la metodología SCRUM, este capítulo está dividido en:

3.1. Pre-game: Corresponde a la recopilación, priorización de las historias de usuario para definir las funcionalidades de las aplicaciones desarrolladas además se definen los Sprints necesarios para desarrollar el proyecto y el tiempo estimado de cada sprint.

3.2. Game: En esta fase se procede al desarrollo de cada uno de los Sprints y los resultados de cada uno.

3.3 Comentarios finales del capítulo: Es un breve resumen de los resultados obtenidos.

3.1. Pre-Game

3.1.1. Planeación.

Durante esta fase a través de reuniones se elaboraron las historias de usuario a través de las cuales se extrajeron los requerimientos para el desarrollo del proyecto los cuáles se pueden ver en el Anexo 2. Historias de Usuario.

Partiendo de estos requerimientos se desarrollaron el Product Backlog (ver Anexo 3), en el cuál se puede ver cómo se van a desarrollar las historias de usuario y en que Sprint, por último se tiene el Sprint Backlog (ver Anexo 4. Sprint Backlog.), el cual muestra de una manera más detallada las tareas que se realizaron por cada una de las iteraciones y el tiempo que tomo realizar dichas tareas.

En base a los Sprints definidos en esta fase utilizando una estructura desagregada de tareas se agrupan las tareas necesarias para cada Sprint y se realiza una estimación del tiempo que tomará el desarrollo de cada Sprint.

Para visualizar a mayor detalle las tareas de cada Sprint revisar el Anexo 5. Estructura desagregada de Tareas en WBS Schedule Pro.

3.2. Game

En esta sección se detallan las tareas realizadas para la implementación de las historias de usuario según el Sprint Backlog y la estructura desagregada de tareas, se han clasificado las historias de usuario en función al tipo de requisito de contempla ya sean de Consumo, Visualización o Almacenamiento,

3.2.1. C001- Consumir información geo-referenciada

3.2.1.1. Diseño de consultas SPARQL

La Tabla 15 muestra la consulta base para el proyecto y una descripción de los campos obtenidos, cabe aclarar que dada la heterogeneidad de los recursos y datasets a utilizar, se escogieron aquellos campos con información más común y relevante para un usuario.

Tabla 15. Consulta base para recursos geo-referenciados.

Consulta	Etiqueta	Descripción
<pre> PREFIX dbo: <http://dbpedia.org/ontology/> PREFIX dbp: <http://dbpedia.org/property/> PREFIX dct: <http://purl.org/dc/terms/> PREFIX foaf: <http://xmlns.com/foaf/0.1/> SELECT ?name ?desc ?lat ?long ?res ?city ?country ?img ?wiki (group_concat(distinct ?p;separator=',') as ?page) (group_concat(distinct ?t;separator=',') as ?type) WHERE { ?res rdfs:label ?name ; geo:lat ?lat ; dbo:country ?countryo; geo:long ?long ; dbo:city dbo:location ?cityo ; rdfs:comment ?desc ; dct:subject ?to. ?to rdfs:label ?t. ?countryo rdfs:label ?country. ?cityo rdfs:label ?city. OPTIONAL {?res foaf:isPrimaryTopicOf ?wiki}. OPTIONAL {?res dbo:thumbnail ?img}. OPTIONAL{?res foaf:homepage ?p}. } GROUP BY ?name ?img ?city ?desc ?res ?lat ?long ?wiki ?country </pre>	name	Nombre del recurso.
	desc	Breve descripción del recurso.
	lat	Latitud donde se encuentra localizado el recurso.
	long	Longitud donde se encuentra localizado el recurso.
	res	Uri del recurso.
	city	Ciudad donde se encuentra el recurso.
	country	País al que pertenece un recurso.
	img	Imagen del recurso.
	page	Página web del recurso, si es que la tiene
	type	Categorías a las que pertenece dicho recurso.
	wiki	Enlace a la página de Wikipedia correspondiente.

Fuente: El Autor.

Elaboración: El Autor.

La Tabla 16 se muestra los filtros que se utilizarán para que la consulta retorne resultados ya sea cercanos a la ubicación o ligados a una ciudad o país específico, teniendo en cuenta que aquellos valores encerrados por corchetes, serán dados por el usuario o su dispositivo.

Tabla 16. Filtros para consulta.

Filtros		
Ubicación Actual	Por ciudad	Por País
FILTER (?long > [Longitud] - 0.05 && ?long < [Longitud] + 0.05 && ?lat > [Latitud] - 0.05 && ?lat < [Latitud] + 0.05)	FILTER (regex(?city, "[".\$ciudad."','i')).	FILTER (regex(?country, '['País'],'i')).

Fuente: El Autor.

Elaboración: El Autor.

3.2.1.2. Diseño del servicio Web de Consulta.

El servicio web de consulta es un servicio web tipo REST, en dónde al recibir los parámetros realiza una consulta sobre el dataset que se haya definido dentro de los parámetros y entregando los resultados de la consulta en un documento tipo JSON.

La Tabla 17 muestra los parámetros necesarios para ejecutar una consulta en el servicio web de consulta.

Tabla 17. Descripción de parámetros del servicio web de consulta.

Parámetro	Valor	Descripción
método	Coor	Permite realizar búsqueda por ubicación actual (Latitud,longitud).
	Ciudad	Permite realizar la búsqueda por una ciudad.
	Fav	Permite obtener los recursos favoritos de un usuario.
	Share	Permite obtener la descripción de un recurso específico para ser compartido.
	País	Permite realizar una búsqueda de recursos por país.
param	<<Latitud,longitud>>	Utilizado para realizar la búsqueda por ubicación actual.
	<<Ciudad>>	Utilizado para realizar la búsqueda por la ciudad ingresada.
	<<Google_id>>	Utilizado para buscar recursos favoritos de un usuario.
	<<Uri>>	Uri de un recurso a ser descrito.
dataset	Dbp DbpEs Latam	Data set sobre el cuál se realizará la búsqueda.
lang	es en	Lenguaje para presentación de resultados.
tipo	All <<Categoría>>	Categoría usada para filtro de resultados.

Fuente: El Autor.

Elaboración: El Autor.

Haciendo uso de la librería cURL de PHP con la consulta elaborada en base a los parámetros ingresados por el usuario y un formato de respuesta tipo JSON, se ejecuta la consulta sobre un SPARQL endpoint específico, que es obtenido a través del parámetro dataset, en el servicio web.

Para más detalles del funcionamiento del servicio web ver el Anexo 13. Manual de programador.

La Figura 7 muestra un ejemplo de consulta para obtener los recursos pertenecientes a la ciudad de Loja en DBpedia en donde se ingresaron los siguientes parámetros: método=ciudad, param=Loja, dataset=Dbp, lang=es y tipo=All.

```

{
  "head": {
    "link": [],
    "vars": [
      "name",
      "desc",
      "lat",
      "long",
      "res",
      "city",
      "img",
      "page",
      "type"
    ]
  },
  "results": {
    "distinct": false,
    "ordered": true,
    "bindings": [
      {
        "name": {
          "type": "literal",
          "xml:lang": "es",
          "value": "Universidad T\u00E9cnica Particular de Loja"
        },
        "desc": {
          "type": "literal",
          "xml:lang": "es",
          "value": "La Universidad T\u00E9cnica Particular de Loja, tambi\u00E9n conocida como la Universidad Cat\u00F3lica de Loja, es una instituci\u00F3n de educaci\u00F3n superior aut\u00F3noma, con finalidad social y p\u00fablica, pudiendo impartir ense\u00F1anza, desarrollar investigaciones con libertad cient\u00EDfica \u00F3 administrativa, y participar en los planes de desarrollo del pa\u00EDs, otorgar, reconocer y revalidar grados acad\u00E9micos y t\u00EDtulos profesionales; y en general, realizar las actividades propias para la consecuci\u00F3n de sus fines. Desde 2012 pertenece a la Red Ecuatoriana de Universidades para Investigaci\u00F3n y Postgrados."
        },
        "lat": {
          "type": "typed-literal",
          "datatype": "http://www.w3.org/2001/XMLSchema#float",
          "value": "-3.9867"
        },
        "long": {
          "type": "typed-literal",
          "datatype": "http://www.w3.org/2001/XMLSchema#float",
          "value": "-79.1989"
        },
        "res": {
          "type": "uri",
          "value": "http://dbpedia.org/resource/Universidad_T\u00E9cnica_Particular_de_Loja"
        },
        "city": {
          "type": "uri",
          "value": "http://dbpedia.org/resource/Loja,_Ecuador"
        },
        "img": {
          "type": "uri",
          "value": "http://commons.wikimedia.org/wiki/Special:FilePath/Universidad_T\u00E9cnica_Particular_de_Loja.png?width=300"
        },
        "page": {
          "type": "literal",
          "value": "http://www.utpl.edu.ec/"
        },
        "type": {
          "type": "literal",
          "value": "http://dbpedia.org/class/yago/Abstraction100002137,http://dbpedia.org/class/yago/EducationalInstitution108276342,http://dbpedia.org/class/yago/Group100031264,http://dbpedia.org/class/yago/Institution108053576,http://dbpedia.org/class/yago/Organization108008335,http://dbpedia.org/class/yago/SocialGroup107950920,http://dbpedia.org/class/yago/University108286569,http://dbpedia.org/class/yago/WikicatEducationalInstitutionsEstablishedIn1971,http://dbpedia.org/class/yago/WikicatUniversitiesInEcuador,http://dbpedia.org/class/yago/YagoLegalActor,http://dbpedia.org/class/yago/YagoLegalActorGeo,http://dbpedia.org/class/yago/YagoPermanentlyLocatedEntity,http://dbpedia.org/ontology/Agent,http://dbpedia.org/ontology/EducationalInstitution,http://dbpedia.org/ontology/Organisation,http://dbpedia.org/ontology/University,http://schema.org/CollegeOrUniversity,http://schema.org/EducationalOrganization,http://schema.org/Organization,http://umbel.org/umbel/rc/University,http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Agent,http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#SocialPerson,http://www.w3.org/2002/07/owl#Thing,http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing,http://www.wikidata.org/entity/Q3916,http://www.wikidata.org/entity/Q43229"
        }
      }
    ]
  }
}

```

Figura 7. Resultados de consulta en el servicio web para recursos de Loja en DBpedia.

Fuente: El Autor.

Elaboraci\u00F3n: El Autor.

3.2.2. A001- Almacenar recursos favoritos

3.2.2.1. Dise\u00F1o de la Base de datos.

Para el almacenamiento de los recursos favoritos del usuario se cre\u00F3 una base de datos en la nube la cual consta de 5 tablas:

- **Usuarios:** Almacena informaci\u00F3n sobre la cuenta y configuraci\u00F3n del usuario contiene: Una llave for\u00E1nea `id_dataset` para relacionar al usuario con su dataset favorito y la llave for\u00E1nea `id_lenguaje` para relacionar al usuario con el lenguaje definido.
- **Recursos:** Contiene datos acerca de los recursos almacenados, estos datos incluyen: Una llave for\u00E1nea `id_usuario` para relacionar un recurso con el usuario que la almacen\u00F3 al igual que un campo `id_dataset` para indicar desde que fuentes de datos proviene el recurso e informaci\u00F3n propia del recurso como su URI y un identificador num\u00E9rico como llave primaria.
- **Datasets:** Contiene informaci\u00F3n sobre las fuentes de datos sobre las cuales se pueden hacer consultas incluyen campos indicando: un acr\u00F3nimo para ser utilizado desde el servicio web, un nombre del dataset y una direcci\u00F3n del endpoint a d\u00F3nde las consultas ser\u00E1n enviadas y ejecutadas.
- **Lenguajes:** Es un cat\u00E1logo de aquellos lenguajes para presentar los resultados incluyen un acr\u00F3nimo para ser utilizados desde el servicio web, un nombre del lenguaje y un identificador num\u00E9rico como llave primaria.

La Figura 8 muestra de una manera gr\u00E1fica las relaciones entre las tablas y las propiedades de cada tabla.

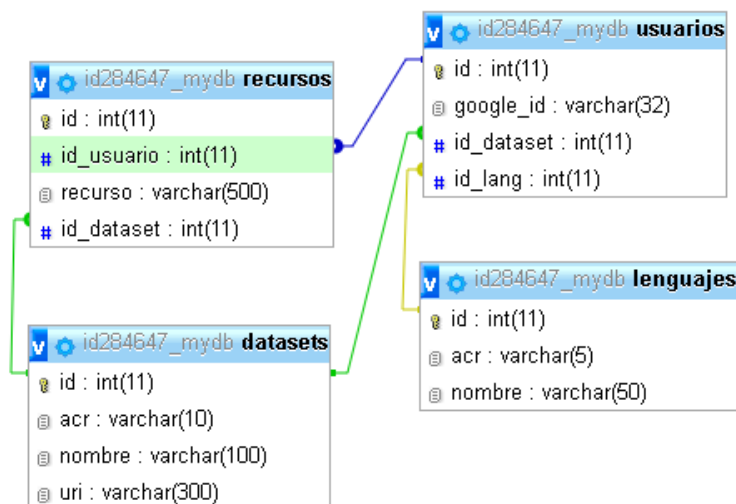


Figura 8. Modelo de base de datos
Fuente: El Autor.
Elaboración: El Autor.

Adicionalmente una vista fue creada para hacer más fácil el consumo de las configuraciones del usuario esta vista incluye el id de la cuenta de google, el acrónimo del lenguaje y dataset preferidos.

Para una descripción más detallada de cada una de las tablas y sus propiedades revisar el Anexo 1. Diccionario de datos.

3.2.2.2. Diseño del servicio web para almacenamiento.

El servicio web de almacenamiento es un servicio web tipo REST al igual que el servicio de consulta, sin embargo este realiza las operaciones de almacenamiento de recursos favoritos en la base de datos definida en el apartado anterior, además realiza tareas como extraer la información de los catálogos en la base de datos para ser mostrada en las interfaces web y móvil también se encarga de crear una configuración de usuario por defecto al momento de loguearse por primera vez en la aplicación y actualizar las configuraciones posteriormente.

La Tabla 18 muestra los campos necesarios para utilizar el servicio web de almacenamiento.

Tabla 18. Descripción de parámetros del servicio web de almacenamiento.

Parámetro	Valores	Descripción
Metodo	CREATE	Permite crear un usuario en la base de datos, además de una configuración por defecto (DBpedia-Latino América, lenguaje español).
	INSERT	Al ingresar un recurso y un dataset, este es buscado en la base de datos, si no se encuentra es almacenado como recurso favorito, caso contrario se elimina el recurso.
	UCONF	Permite actualizar la configuración actual de un usuario (Dataset y Lenguaje).
	GCATALOGS	Permite Obtener el listado de los Datasets y lenguajes disponibles.
User	<<Google_id>>	Identificador de google de un usuario.
Rec	<<Uri de recurso>>	Identificador de un recurso.
Dataset	Dbp DbpEs Latam	Nuevo valor de dataset a almacenar.
Lang	es en	Nuevo valor de lenguaje a almacenar.

Fuente: El Autor.

Elaboración: El Autor.

Para revisar en detalle el funcionamiento del servicio web de almacenamiento revisar el Anexo 13. Manual de programador.

3.2.3. C002- Realizar las consultas desde dispositivo móvil.

3.2.3.1. Diseñar arquitectura de la Aplicación Móvil.

La aplicación móvil tiene una arquitectura en 3 capas debido a que al separar la funcionalidad de las interfaces y el acceso a datos hace que sea más sencillo el mantenimiento de la aplicación además de hacer más fácil realizar cambios en alguna capa sin afectar las demás.

Las capas definidas para esta aplicación son:

- **Interfaz de usuario:** Corresponden a los elementos de la interfaz gráfica como menús, formularios, botones, notificaciones, es decir, todo con lo que el usuario interactúa directamente
- **Lógica de negocio:** Contiene códigos necesarios para ejecutar las diferentes funcionalidades de la aplicación o procesar los datos.
- **Datos:** En esta capa se realiza la comunicación de la aplicación con los servicios web REST de consulta y almacenamiento además de la comunicación con servicios para la autenticación del usuario a través de google Plus OAUTH, el API de Google Maps para generar los mapas y el Network Location Provider para obtener las coordenadas actuales del dispositivo.

En la Figura 9 se puede observar de una manera gráfica la arquitectura de la aplicación móvil.

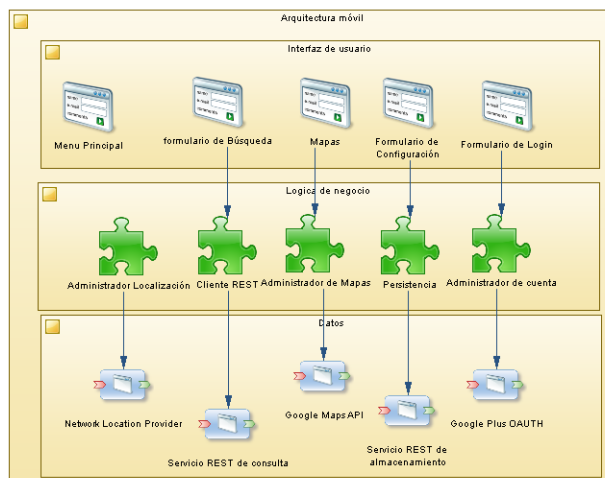


Figura 9. Arquitectura de la aplicación móvil
Fuente: El Autor.
Elaboración: El Autor.

A continuación se va a describir la funcionalidad de cada uno de los módulos de la capa de negocio.

3.2.3.1.1. Módulo de Administrador localización.

Este módulo se encarga de todo lo relacionado con la ubicación del usuario teniendo como tareas principales: la gestión de permisos de localización y determinar la posición actual.

3.2.3.1.2. Módulo de Cliente REST.

Este módulo es el encargado de la gestión de la información obtenida desde los servicios web en formato JSON, las principales tareas de este módulo son: gestionar la comunicación con el servicio web, obtener la información como un objeto JSON que se pueda interpretar y enviar la información a las clases que requieran hacer uso de estos objetos.

3.2.3.1.3. Módulo de Administrador Mapas.

Este es un módulo dedicado para la visualización de la información a través del uso de mapas, originando marcadores que indiquen la posición de cada uno de los resultados obtenidos en las búsquedas y ofrecer información sobre cada uno de ellos.

3.2.3.1.4. Módulo de persistencia

Este módulo es el encargado de almacenar toda la información correspondiente a los usuarios de la aplicación como la información de su cuenta y sus recursos favoritos, en una base de datos externa común para las versiones web y móvil.

3.2.3.1.5. Módulo de Administrador de cuentas.

Este módulo está encargado de la gestión de las cuentas de usuario, realizando las operaciones de autenticación con una cuenta de google.

3.2.3.2. Desarrollo del Módulo de Administrador de localización. (Móvil)

La Tabla 19 muestra una comparación acerca de las dos estrategias de geo-localización de un dispositivo móvil. En base a las comparaciones realizadas se ha seleccionado la localización por medio de la red debido a que esta es mucho más rápida y requiere solo de una conexión a internet para utilizarse. Dado que las consultas se realizan en rangos cercanos a la posición del dispositivo, la precisión de la geo-localización no es necesaria.

Tabla 19. Comparación de estrategias de geo-localización.

Localización por GPS	Localización por red
Localización más precisa del dispositivo.	Localización menos precisa del dispositivo.
Utiliza el satélite GPS para determinar la localización.	Determina la localización en base a los puntos de acceso WIFI.
Dependiendo de las condiciones tarda un considerable tiempo en determinar la posición.	La localización es relativamente más rápida.
Requiere tener habilitado el GPS en el teléfono.	Requiere de una conexión a internet.

Fuente: El Autor.

Elaboración: El Autor.

La Figura 10 muestra los dos métodos principales para determinar la localización de un dispositivo ANDROID: **buscarCoordenadas()** el cuál es el encargado de solicitar las coordenadas del dispositivo y **OnlocationChanged()** el cuál es el encargado de recibir las coordenadas del objeto LocationManager y enviar esa información a la clase que lo requiera para ser procesada más adelante.

```
public void BuscarCoordenadas() {  
    //Este método permite obtener las coordenadas actuales del usuario  
    locationManager = (LocationManager) main.getSystemService(Context.LOCATION_SERVICE);  
    if (ActivityCompat.checkSelfPermission(main, Manifest.permission.ACCESS_FINE_LOCATION)  
        != PackageManager.PERMISSION_GRANTED  
        && ActivityCompat.checkSelfPermission(main, Manifest.permission.ACCESS_COARSE_LOCATION)  
        != PackageManager.PERMISSION_GRANTED) {  
        return ;  
    }  
    locationManager.requestLocationUpdates(locationManager.NETWORK_PROVIDER, 1000, 1, this);  
}  
@Override  
public void onLocationChanged(Location location) {  
    //Este método al detectar una actualización en las coordenadas del usuario lo envía a la clase que lo necesite  
    locationManager.removeUpdates(this);  
    main.setLocation(location);  
}
```

Figura 10. Métodos para la geo-localización

Fuente: El Autor

Elaboración: El Autor.

En la Figura 11 muestra las coordenadas que se obtuvieron del dispositivo a través del módulo de Administrador de localización.



Figura 11. Obtención de coordenadas actuales en un dispositivo móvil ANDROID.

Fuente: El Autor.

Elaboración: El Autor.

3.2.3.3. Desarrollo del Módulo ClienteREST (Móvil)

Para la gestión de la comunicación con el servicio web se hace uso de un objeto tipo **requestQueue** de la librería Volley, siendo necesario en primer lugar importar la siguiente librería en las dependencias del archivo de configuración build.gradle a través del entorno de programación ANDROID STUDIO:

- `compile 'com.mcxiaoke.volley:library:1.0.19'`

Volley se encarga de gestionar una cola de solicitudes HTTP, haciendo uso del método **getUrl()** se realizan las peticiones al servicio web ordenando dichas peticiones en la cola del objeto **requestQueue** una vez se obtenga una respuesta desde el servicio web, se hace uso del método **OnResponse()** para enviar el objeto JSON recibido desde el servicio web a la clase que lo requiera.

En la Figura 12 se puede ver el código comentado de los métodos para realizar la comunicación con los servicios web desde un dispositivo móvil ANDROID.

```

public void getUrl(String url){
    //Este método permite consultar una URL para obtener un objeto JSON
    Log.d("MyTAG",url);
    //Iniciar Cola de solicitudes
    requestQueue= Volley.newRequestQueue(context);
    //Crear petición al servicio Web
    JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url,this, this);
    //Enviar solicitud a cola
    requestQueue.add(request);
    //Iniciar cola de solicitudes
    requestQueue.start();
}

@Override
public void onErrorResponse(VolleyError error) {...}

@Override
public void onResponse(JSONObject response) {
    try {
        //Enviar Objeto JSON a la clase que lo requiera.
        json.setJsonObject(response);
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
}

```

Figura 12. Métodos para comunicación con servicio web desde aplicación móvil.

Fuente: El Autor.

Elaboración: El Autor.

La Figura 13 muestra cómo a través del módulo cliente REST se realiza una consulta de recursos cercanos a la ubicación del dispositivo y se obtiene una respuesta en formato JSON.

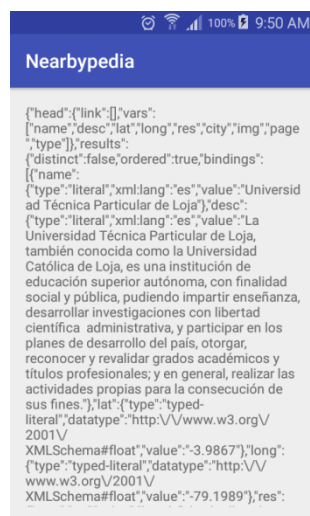


Figura 13. Resultados de consulta de recursos cercanos a la ubicación actual a través de un dispositivo móvil.

Fuente: El Autor

Elaboración: El Autor.

3.2.3.4. Diseño de la Arquitectura de la Aplicación web.

Para el diseño de la aplicación web se optó por la utilización de una arquitectura cliente-servidor debido a que la centralización del control, los archivos y bases de datos; hace que sea más fácil el mantenimiento de la aplicación y reduce la carga para los clientes ejecutando la mayor parte del procesamiento en el servidor.

La Figura 14 muestra la arquitectura de la aplicación web de una manera gráfica

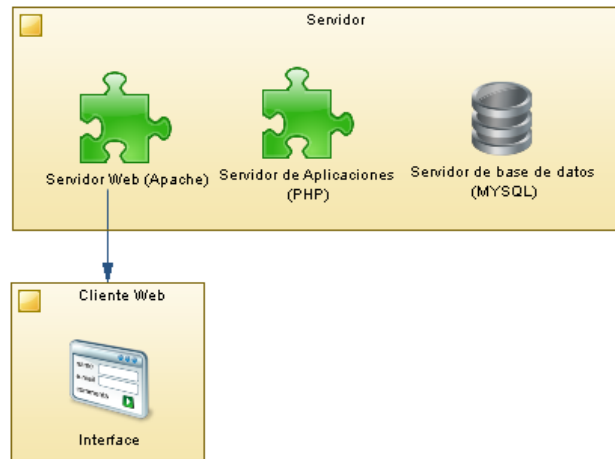


Figura 14. Arquitectura de la Aplicación Web.
Fuente: El Autor
Elaboración: El Autor.

A continuación se describen cada uno de los elementos del servidor y sus funciones.

3.2.3.4.1. Servidor web Apache.

Encargado de la transferencia de Hipertexto a través del protocolo HTTP, es utilizado para entregar los archivos que conforman una página web a un cliente, como una respuesta a una solicitud.

3.2.3.4.2. Servidor de Aplicaciones (PHP).

Proporciona los servicios de Aplicación a los clientes, en este se encuentran todos los módulos para el funcionamiento de la aplicación (Lógica del Negocio) y el acceso a datos.

3.2.3.4.3. Servidor de base de datos (MySQL).

Este servidor se encarga de organizar y almacenar toda la información generada por las aplicaciones tanto web como móvil.

3.2.3.5. Desarrollo del módulo de administrador de localización (Web).

La Figura 15 muestra como haciendo uso de las herramientas de geo localización de HTML5 es posible obtener la localización actual del usuario. Se utilizan los métodos **getLocation()** para obtener la localización actual y **callPositionForm()** para enviar las coordenadas del cliente al archivo que lo requiera.

```

//Obtener la ubicación actual del usuario
function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(callPositionform);
    } else {
        alert('Geolocalización no soportada');
    }
}

//Enviar los valores de la geolocalización para la consulta
function callPositionform(position) {
    post("<?>= base_url('index.php/controller/search')>>\"", {param: position.coords.latitude+", "+position.coords.longitude, metodo: 'coord' });
}

```

Figura 15 Métodos de geo-localización de Aplicación web.

Fuente: El Autor

Elaboración: El Autor.

La Figura 16 muestra las coordenadas actuales obtenidas por el módulo de geo-localización web, cabe recalcar que es necesario que el usuario otorgue el permiso correspondiente al hacer clic en una ventana emergente, las coordenadas se calculan en base al rango de las direcciones IP y los puntos de acceso a la red.



Figura 16. Resultados de Geo-localización por internet

Fuente: El Autor.

Elaboración: El Autor.

3.2.3.6. Desarrollo del módulo de Cliente REST (Web)

Para el consumo de los servicios web de almacenamiento y de consulta se hace uso de la librería cURL de PHP, en donde se tiene que enviar como parámetros: la url del endpoint al que se está consultando, la consulta SPARQL codificada y el formato de respuesta en este caso JSON. Este método entrega como resultado un objeto tipo JSON para ser operado más adelante.

En la Figura 17 se muestra el método para la comunicación con los servicios web desde una aplicación web utilizando cURL.


```
function request($url){
    // Verificar si CURL se encuentra instalado
    if (!function_exists('curl_init')){
        die('CURL is not installed!');
    }
    // Obtener un administrador CURL
    $ch= curl_init();
    // Establecer la URL a solicitar
    curl_setopt($ch,
        CURLOPT_URL,
        $url);
    curl_setopt($ch,
        CURLOPT_RETURNTRANSFER,
        true);
    //ejecutar consulta y obtener resultado en response
    $response = curl_exec($ch);
    //Cerrar administrador de CURL
    curl_close($ch);
    return $response;
}
```

Figura 17. Método de comunicación con el servicio web en aplicación Móvil.
Fuente: El Autor.
Elaboración: El Autor.

La Figura 18 muestra los resultados de ejecutar una consulta para obtener recursos cercanos a la ubicación del usuario desde la aplicación web.

The screenshot shows a web application interface with a dark header containing the title 'Nearbypedia' and a hamburger menu icon. The main content area displays a JSON object representing search results. The JSON includes a 'head' section with metadata and a 'results' array containing one result for 'Universidad Técnica Particular de Loja'. The result object contains detailed information such as its description, location (latitude and longitude), and various URIs for further information.

Figura 18. Recursos cercanos al usuario desde aplicación web
Fuente: El Autor.
Elaboración: El Autor.

3.2.4. V001-Visualizar información en mapas

3.2.4.1. Diseño de la interfaz (Móvil)

Para el diseño de la interfaz se realizó un wireframe, el cual es una representación visual o prototipo, donde se pueden observar la estructura y relaciones entre las interfaces así como sus características principales.

Tomando como base las directrices definidas por el equipo de ANDROID Developers para el diseño de las interfaces:

- Declarando elementos de la interfaz de usuario en xml.
- Creando instancias de los elementos del diseño en tiempo de ejecución.
- Utilizando un diseño lineal en lugar de un diseño relativo para una mejor distribución de los elementos de interfaz.

- Procurar que las interfaces sean livianas para reducir el tiempo de procesamiento y los recursos necesarios para la aplicación.

El Anexo 6. Diseño de la interfaz Móvil muestra el prototipo y la secuencia de despliegue de las interfaces.

3.2.4.2. Despliegue de mapas (Móvil)

En base al archivo JSON obtenido al consumir el servicio web de consulta, se elaboran mapas, ubicando cada uno de los recursos en forma de un marcador y a su vez al pulsar alguno de estos marcadores se obtiene una vista un poco más detallada acerca del recurso.

La Figura 19 muestra la visualización de un recurso como marcador en el mapa y su vista en detalle con sus propiedades e imagen.

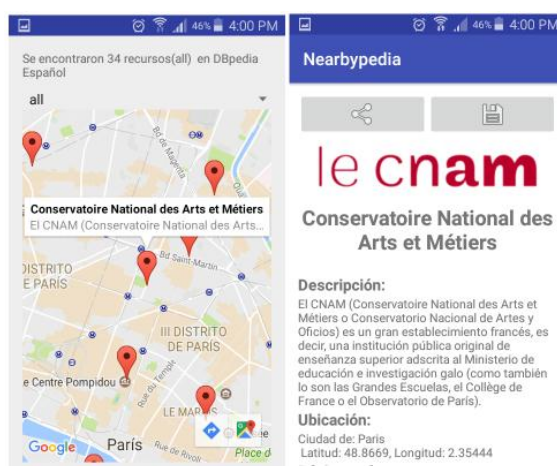


Figura 19. Visualización de recursos en dispositivos móviles

Fuente: El Autor.

Elaboración: El Autor.

3.2.4.3. Diseño de la interfaz (Web).

Al igual que con la interfaz móvil se desarrolló un prototipo de las interfaces, se utilizó un diseño en cajas con bootstrap para ordenar el contenido dentro de bloques y hacer páginas web responsivas.

En el Anexo 7. Diseño de la interfaz Web. se puede observar en detalle las interfaces, la secuencia de operación y características principales del diseño.

3.2.4.4. Despliegue de mapas (Web)

Por medio del servicio web de consulta, JavaScript y el API de google maps se generan mapas mostrando los recursos a manera de marcadores.

La Figura 20 muestra como un recurso es desplegado como marcador en un mapa y la vista en detalle del recurso al hacer clic en un marcador.



Figura 20. Despliegue de mapas en aplicación web.

Fuente: El Autor.

Elaboración: El Autor.

3.2.5. A002-Logueo al sistema utilizando cuenta de Google.

3.2.5.1. Implementar Autenticación con Google (Móvil).

Para implementar la autenticación con Google fue necesario agregar las siguientes librerías en las dependencias dentro del archivo de dependencias **build.gradle** a través del entorno de desarrollo ANDROID Studio:

- compile 'com.google.android.gms:play-services:10.0.1'
- compile 'com.google.android.gms:play-services-auth:10.0.1'
- compile 'com.google.android.gms:play-services-plus:10.0.1'

De esta manera se puede hacer uso de las clases objetos y métodos necesarios para la gestión del cliente OAUTH 2.0 para Google Plus, en la clase **PLUS** se encuentran definidos los métodos más importantes para la gestión de las cuentas, utilizando un objeto tipo **GoogleAPIClient** para el almacenamiento de la información del cliente, y el método **get_GoogleAPIClient()** para enviar la información del mismo a la clase que lo requiera.

La Figura 21 muestra los métodos y dependencias necesarias para realizar la autenticación de un cliente OAUTH 2.0 en dispositivos móviles ANDROID.

build.gradle	Clase PLUS
<pre>dependencies { compile fileTree(dir: 'libs', include: ['*.jar']) androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', { exclude group: 'com.android.support', module: 'support-annotations' }) compile 'com.android.support:appcompat-v7:24.2.1' compile 'com.mcxiaoke.volley:library:1.0.19' compile 'com.google.android.gms:play-services:10.0.1' compile 'com.android.support:design:24.2.1' compile 'com.android.support:support-v4:24.2.1' compile 'com.android.support:support-vector-drawable:24.2.1' compile 'com.google.android.gms:play-services-auth:10.0.1' compile 'com.google.android.gms:play-services-plus:10.0.1' compile 'com.android.support:multidex:1.0.0' testCompile 'junit:junit:4.12' } apply plugin: 'com.google.gms.google-services'</pre>	<pre>public class PLUS { //Clase para la gestión de la autenticación utilizando google plus private static final String TAG = "PLUS"; private static PLUS instance = null; private static GoogleApiClient mGoogleApiClient = null; protected PLUS() { } //Método para registrar una nueva instancia de cliente google. public static PLUS getInstance(GoogleApiClient aGoogleApiClient) { if(instance == null) { instance = new PLUS(); if (mGoogleApiClient == null) mGoogleApiClient = aGoogleApiClient; } return instance; } //Método para obtener el id del cliente google public GoogleApiClient get_GoogleApiClient() { return mGoogleApiClient; } //Método para expulsar cliente google public void logout() { mGoogleApiClient.clearDefaultAccountAndReconnect(); } }</pre>

Figura 21. Dependencias y Métodos necesarios para la autenticación de cliente OAUTH 2.0

Fuente: El Autor.

Elaboración: El Autor.

La Figura 22 muestra cómo se despliega la interfaz de logueo en los dispositivos móviles.

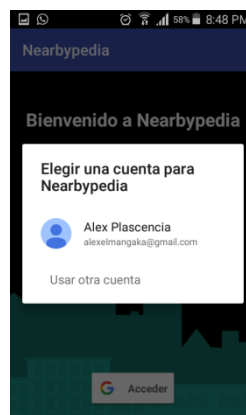


Figura 22. Autenticación con cuenta de google (Móvil)

Fuente: El Autor.

Elaboración: El Autor.

3.2.5.2. Implementar Funcionalidad Almacenamiento (Móvil).

Para el almacenamiento de un recurso en dispositivos móviles un usuario al hacer clic en el botón de guardar se comunica con el servicio web de almacenamiento enviando la URI del recurso, el dataset al que pertenece el recurso y la identificación de su cuenta de google. Si el recurso se almacena exitosamente se despliega un mensaje al usuario, en caso de que el recurso haya sido previamente almacenado como favorito se elimina el recurso de la base de datos y se notifica al usuario que se ha eliminado el recurso. En la Figura 23 se puede observar como un recurso es almacenado en la base de datos desde la interfaz móvil.



Figura 23. Almacenamiento de recursos favoritos de usuario en aplicación móvil

Fuente: El Autor.

Elaboración: El Autor.

3.2.5.3. Implementar Autenticación con Google (Web)

Una de las características definidas dentro de las historias de usuario era la necesidad de se pueda autenticar utilizando su cuenta de Google, para ello se hace uso del API de Google para autenticación OAUTH 2.0 y JavaScript para desplegar una ventana de logueo nativa de google, recuperando el identificador de la cuenta para ser utilizada dentro de la aplicación.

La Figura 24 muestra los métodos en JavaScript siendo **onSignIn()** el método para obtener el id de cuenta de google y **post()** el método para enviar la id de usuario al menú principal.

```
<script type="text/javascript">
//Obtener id de usuario y redireccionar al menú principal
function onSignIn(googleUser) {
  var profile = googleUser.getBasicProfile();
  var ID=profile.getId();
  post("<?= base_url('index.php/controller/index')>",{sID: profile.getId()})
}

//Enviar id de usuario a través del método post.
function post(path, params, method) {
  method = method || "post";
  var form = document.createElement("form");
  form.setAttribute("method", method);
  form.setAttribute("action", path);
  for(var key in params) {
    if(params.hasOwnProperty(key)) {
      var hiddenField = document.createElement("input");
      hiddenField.setAttribute("type", "hidden");
      hiddenField.setAttribute("name", key);
      hiddenField.setAttribute("value", params[key]);
      form.appendChild(hiddenField);
    }
  }
  document.body.appendChild(form);
  form.submit();
}
</script>
```

Figura 24. Métodos para autenticación con cliente OAUTH 2.0 en aplicación web.

Fuente: El Autor.

Elaboración: El Autor.

La Figura 25 muestra la interfaz que se despliega para que el usuario realice la autenticación utilizando su cuenta de Google.

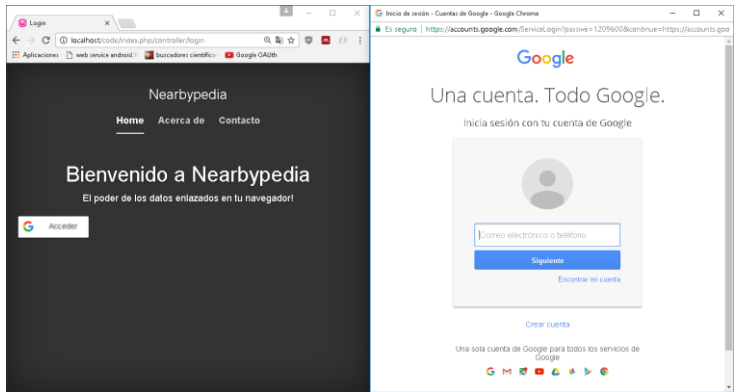


Figura 25. Autenticación con cuenta de google (Web)
Fuente: El Autor.
Elaboración: El Autor.

3.2.5.4. Implementar Funcionalidad Almacenamiento (Web)

Utilizando la librería cURL para PHP se realiza la llamada al servicio web de almacenamiento enviando los parámetros necesarios: id de cuenta de google, URI del recurso y dataset al que pertenece, esta información es almacenada dentro de la base de datos y el servicio web envía un mensaje notificando si el recurso fue agregado exitosamente para ser desplegado en la interfaz web.

La Figura 26 muestra como el recurso “Escuela Politécnica Nacional” es almacenado como un recurso favorito en la base de datos.

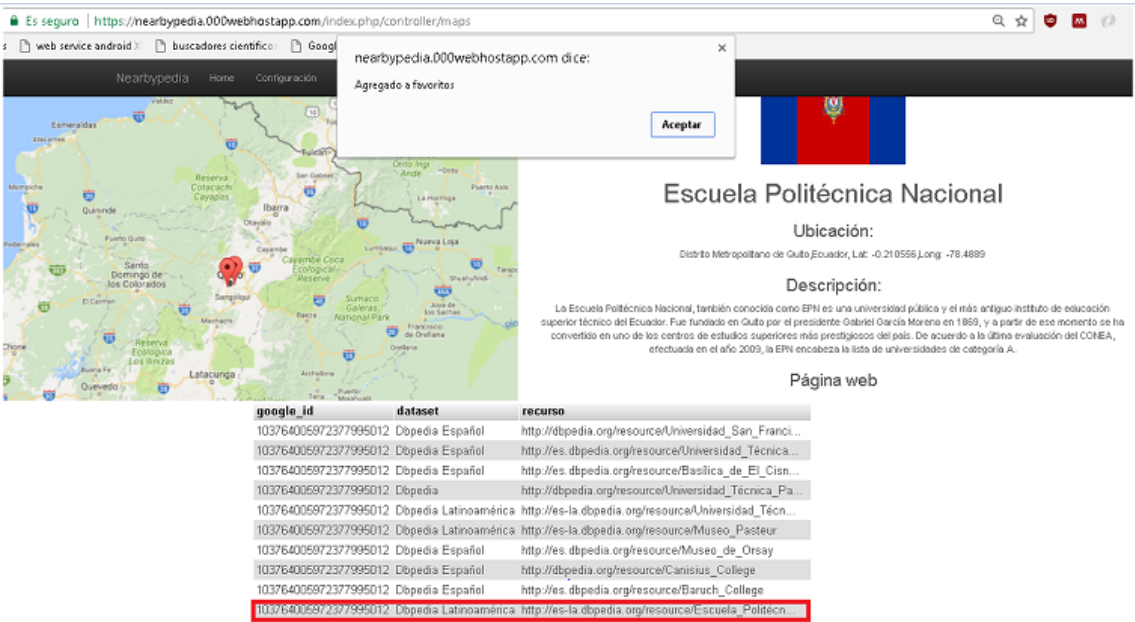


Figura 26. Almacenamiento de recursos favoritos de usuario en navegador.
Fuente: El Autor.
Elaboración: El Autor.

3.2.6. A003- Compartir recursos en las redes sociales

3.2.6.1. Implementar Funcionalidad Compartir en redes sociales (Móvil)

Haciendo uso del método “share” en el servicio web de consulta, se obtiene la información correspondiente del recurso a compartir y con esa información se elabora una página web, que muestra la información del recurso así como su localización geográfica.

Se hace uso de las ventanas nativas de ANDROID Sharing Intent, para compartir la URL del recurso en redes sociales como Facebook o Google+.

La Figura 27 muestra el método que despliega la interfaz nativa de ANDROID para compartir un recurso.

```
sharebtn.setOnClickListener(new View.OnClickListener() {  
    public void onClick(View v) {  
        Intent sharingIntent = new Intent(android.content.Intent.ACTION_SEND);  
        sharingIntent.setType("text/plain");  
        String ShareSub = "Compartir";  
        sharingIntent.putExtra(android.content.Intent.EXTRA_SUBJECT, ShareSub);  
        sharingIntent.putExtra(android.content.Intent.EXTRA_TEXT,  
            "https://nearbypedia.000webhostapp.com/index.php/controller/sharer?param=" +  
            res+"&dataset="+sharedPreferences.getString("Dataset", "Latam"));  
        startActivity(Intent.createChooser(sharingIntent, "Share via"));  
    }  
});
```

Figura 27. Método para compartir recursos utilizando las ventanas nativas de Android.

Fuente: El Autor.

Elaboración: El Autor.

La Figura 28 muestra las interfaces para compartir recursos, tanto la interfaz nativa de Android como las interfaces propias para postear un recurso en las redes sociales de Google+ y Facebook.

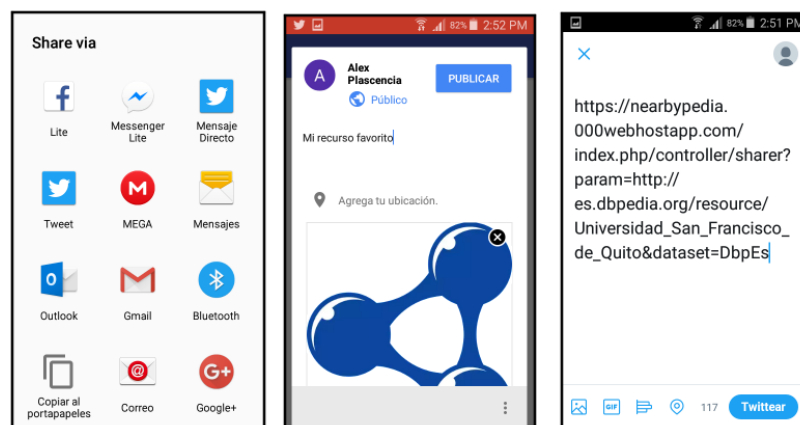


Figura 28. Compartir recursos a través de dispositivo móvil.

Fuente: El Autor

Elaboración: El Autor.

CAPÍTULO 4: PRUEBAS Y VALIDACIÓN

En este capítulo se realizan las pruebas definidas por la metodología SCRUM para validar el funcionamiento de las aplicaciones web y móvil, como aporte adicional se plantea un modelo para la publicación de datos enlazados.

Este capítulo está dividido en:

4.1 Pruebas: Se presenta un resumen de las pruebas realizadas sobre las aplicaciones web y móvil, explicando la finalidad de cada prueba además de los resultados obtenidos.

4.2 Validación: Se realiza una comparación en función de la cantidad de recursos geo-referenciados por países de Latino América y se provee de un modelo ontológico para incentivar la publicación de datos enlazados geo-referenciados.

4.3 Comentarios finales del capítulo: Corresponde a un breve resumen de los resultados obtenidos en este capítulo.

4.1. Pruebas

4.1.1. Pruebas Unitarias

De acuerdo con la metodología de SCRUM en cada uno de los Sprints se realizaron pruebas unitarias del código a fin de asegurar que se cumpla con la funcionalidad correspondiente. Estas pruebas se agruparon en función de los componentes presentes en el Product Backlog.

La Tabla 20 muestra un resumen de los resultados obtenidos por cada una de las pruebas:

Tabla 20. Resultados de pruebas Unitarias.

Componente	Entrada	Salida	Pasó/Reprobó
Consumo de información geo-referenciada (C001)	-Metodo:coor -Parametro: --Lat: -3.9943435 --Long: -79.2095252 -Dataset:Latam -Lenguaje:es -Tipo:all	JSON con 1 resultados de búsqueda.	Pasó
Almacenamiento de recursos Favoritos (A001)	-Metodo:INSERT -Usuario: 103764005972377995012 -Recurso: http://es.dbpedia.org/resource/Universidad_Técnica_Particular_de_Loja -Dataset:DbpES	Registro almacenado en base de datos.	Pasó
Visualización en mapas. (V001)	-Metodo:País -Parametro: Ecuador -Dataset: DbpES -Tipo: Universidades de Ecuador	-Mapa desplegando 10 marcadores en dispositivo móvil. -Mapa desplegando 10 marcadores en sitio web.	Pasó
Consumo de información geo-referenciada desde Smartphone.	-Metodo:País -Parametro:Ecuador -Dataset:Latam -Lenguaje:es -Tipo:all	JSON con 8 resultados de búsqueda.	Pasó

(C002)			
Consumo de información geo-referenciada desde página web. (C003)	-Metodo:Pais -Parametro:Ecuador -Dataset:Latam -Lenguaje:es -Tipo:all	JSON con 8 resultados de búsqueda.	Pasó
Logueo utilizando cuenta de google (A002)	-Cuenta de google: alexelmangaka@gmail.com -Contraseña: *****	Almacena ID de cuenta de google y crea opciones predeterminadas Lenguaje español y dataset dbpedia latinoamérica.	Pasó
Compartir recursos favoritos (A003)	Recurso: http://es-la.dbpedia.org/resource/Universidad_Técnica_Particular_de_Loja Dataset: Latam	-No se pudo compartir utilizando Facebook. -Tweet publicado en Twitter -Post compartido a través de google Plus.	No pasó Facebook empezó a bloquear los enlaces compartidos considerándolos SPAM.

Fuente: El Autor.

Elaboración: El Autor.

4.1.2. Pruebas de Aceptación de usuario.

Las pruebas de Aceptación de usuario tienen como finalidad de validar que la aplicación cumple con las expectativas del cliente en cuanto a funcionalidades y comportamiento.

Para la realización de las pruebas de aceptación de usuario y dado de que no se posee un conjunto de clientes específicos se realizó en función de los criterios de aceptación y agrupadas por funcionalidades que fueron definidas en las historias de usuario, las pruebas se realizaron tanto en la parte móvil como web, obteniendo los mismos resultados, para observar con mayor detalle las pruebas de aceptación ver el Anexo 8. Pruebas de Aceptación de usuario.

4.1.2.1. Pruebas de Aceptación para consumo de información geo-referenciada.

Esta fase de la prueba incluye los componentes de consumo de información del Product Backlog. Se plantearon escenarios de prueba relacionados con los parámetros de búsqueda de recursos para determinar que el software se comporta de la manera esperada.

La Tabla 21 muestra un resumen de las pruebas de aceptación realizadas para validar el consumo de información geo-referenciada.

Tabla 21. Pruebas de Aceptación para consumo de información geo-referenciada.

Criterio de aceptación	Pruebas realizadas	Resultados por prueba	Paso/Reprobó
Búsqueda por ubicación actual	3	1 Recurso Cercano/Prueba.	Pasó
Búsqueda por ciudad.	Loja Quito Guayaquil Cuenca	1 Recurso. 3 Recursos. 3 Recursos. 0 Recurso	Pasó
Búsqueda por país.	Ecuador Argentina Chile Brasil	8 Recursos. 98 Recursos. 26 Recursos. 17 Recursos.	Pasó
Múltiples fuentes de información	Búsqueda de recursos de Ecuador en: DBpedia Db. Español. Db. Latinoamérica	46 Recursos. 46 Recursos. 8 Recursos.	Pasó
Filtro de información	5 Categorías en Quito.	2 Red Ecuatoriana de Universidades 2 Universidades del Ecuador. 2 Universidades de Quito. 2 Universidades de Pichincha. 1 Organización fundada en 1869.	Pasó

Fuente: El Autor.

Elaboración: El Autor.

4.1.2.2. Pruebas de aceptación para almacenamiento y visualización de recursos Favoritos.

Esta fase de la prueba comprende los componentes de almacenamiento definidos en el Product Backlog, para la realización de estas pruebas se procedió a almacenar una cierta cantidad de recursos en la base de datos, evaluando su posterior recuperación y presentación.

La Tabla 22. Muestra las pruebas de aceptación para almacenamiento y visualización realizadas y sus resultados.

Tabla 22. Pruebas de aceptación para almacenamiento y visualización de recursos.

Criterio de aceptación	Pruebas realizadas	Resultados por prueba	Paso/Reprobó
Almacenamiento de Recursos.	5	5 Recursos Almacenados en la base de datos.	Pasó
Visualización en el Mapa	1	5 Marcadores Desplegados en mapa.	Pasó.

Fuente: El Autor.

Elaboración: El Autor.

4.1.3. Pruebas de Integración.

En esta fase se realizó la validación de los componentes para comprobar si trabajan bien entre sí, respondiendo de la manera esperada, para ver a mayor detalle los resultados de estas pruebas ver el Anexo 9. Pruebas de Integración.

La Tabla 23. Muestra los casos definidos para las pruebas de integración describiendo que se necesita probar y los resultados que se obtuvieron.

Tabla 23. Casos de Prueba de integración planteados.

Caso de Prueba	Componentes	Descripción de lo que se Probará	Resultados Obtenidos	Paso/Reprobó
CA001	Consumo - Visualización (Móvil)	Visualización de recursos consumidos a través del servicio web.	Resultados de consultas desplegados como marcadores en Mapas.	Pasó
CA002	Consumo - Almacenamiento (Móvil)	Almacenamiento de recursos consumidos a través del servicio web	N recursos favoritos almacenados.	Pasó
CA003	Almacenamiento - Visualización (Móvil)	Visualización de recursos favoritos almacenados en base de datos.	N recursos almacenados desplegados en mapas.	Pasó
CA004	Consumo - Visualización (Web)	Visualización de recursos consumidos a través del servicio web.	Resultados de consultas desplegados como marcadores en Mapas.	Pasó
CA005	Consumo - Almacenamiento (Web)	Almacenamiento de recursos consumidos a través del servicio web	N recursos favoritos almacenados.	Pasó
CA006	Almacenamiento - Visualización (Web)	Visualización de recursos favoritos almacenados en base de datos.	N recursos almacenados desplegados en mapas.	Pasó

Fuente: El Autor.

Elaboración: El Autor.

4.1.4. Pruebas de Caja negra.

Se sometieron análisis de código estático sobre las aplicaciones móvil, web y el servicio web en SonarQube, con una calificación que oscila entre A-C aprobando exitosamente, determinando una calidad alta del programa, las métricas analizadas para obtener estos indicadores se encuentran definidas en el Anexo 10. Métricas e indicadores analizados por SonarQube.

La Figura 30 muestra los resultados del análisis de código estático en SonarQube.

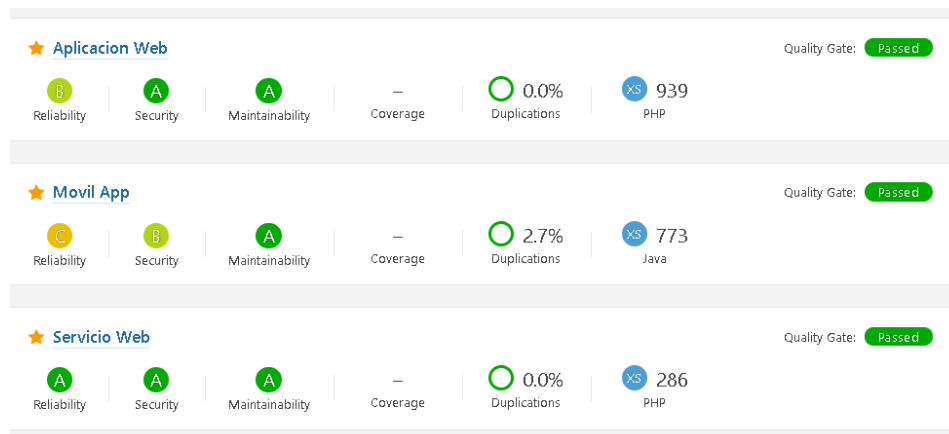


Figura 30. Resultados de análisis en SonarQube

Fuente: El Autor

Elaboración: El Autor.

4.2. Validación.

4.2.1. Comparación de resultados a nivel de Latino América

La Figura 31 muestra los 10 países con mayor cantidad de datos enlazados geo-referenciados a nivel de Latino América en DBpedia Español, ubicándose Ecuador en el 7mo puesto. En base a los resultados obtenidos se puede apreciar que dada la baja cantidad de resultados que se obtienen se vuelve muy importante promover la publicación de información geo-referenciada al proveer al usuario de herramientas, modelos y aplicaciones; para el consumo, procesamiento, visualización y publicación de datos enlazados geo-referenciados.

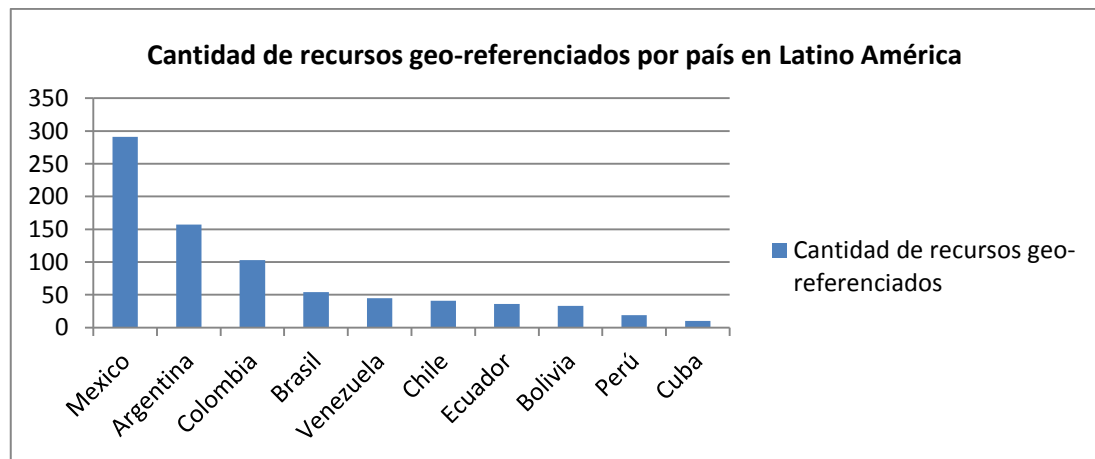


Figura 31. Cantidad de recursos geo-referenciados por País obtenidos con la aplicación.

Fuente: El Autor.

Elaboración: El Autor.

4.2.2. Modelado y publicación de datos enlazados de prueba en servidor Local.

Para contribuir a la publicación de datos enlazados geo-referenciados se construyó un modelo ontológico utilizando las clases y propiedades de vocabularios como:

- **geo:** Vocabulario para expresar propiedades de geo-referencia.
- **dbo:** Corresponde a la ontología de DBpedia

- **schema:** Vocabulario que provee esquemas para estructurar información en internet. .
- **skos:** Es un modelo de datos para compartir y enlazar conocimiento.
- **dct:** Utilizado para la definición de metadatos de un recurso.
- **rdf:** Lenguaje para definir recursos en la web.
- **rdfs:** Provee un vocabulario para modelar datos RDF.

La Tabla 24 muestra las clases que fueron necesarias de cada vocabulario en el desarrollo de un modelo ontológico para describir recursos geo-referenciados.

Tabla 24. Clases utilizadas en el modelo para la publicación de datos enlazados geo-referenciados.

Vocabulario	Clase	Uso
Dbo	Country	Definir recursos como países.
	City	Definir recursos como ciudades.
Skos	Concept	Definir conceptos.
Schema	Place	Definir recursos como lugares.

Fuente: El Autor.

Elaboración: El Autor.

La Tabla 25 muestra las propiedades utilizadas de cada vocabulario en la construcción del modelo ontológico para la publicación de datos enlazados geo-referenciados.

Tabla 25. Propiedades utilizadas en el modelo para la publicación de datos enlazados geo-referenciados.

Vocabulario	Propiedad	Uso
Dbo	country	Enlaza recursos con el país al que pertenecen.
	city	Enlaza recursos con la ciudad a la que pertenece.
	isPrimaryTopicOf	Enlazar el recurso con su descripción en Wikipedia.
Dct	Subject	Incluir recursos en una categoría.
Rdf	Type	Definir el tipo de recurso.
Rdfs	Label	Agrega una etiqueta un recurso.
	Comment	Agregar una descripción a un recurso.
Foaf	Thumbnail	Incluir una imagen para el recurso.
	Homepage	Incluir el sitio web de un recurso.

Fuente: El Autor.

Elaboración: El Autor.

La Figura 32 muestra el modelo ontológico para la publicación de datos enlazados geo-referenciados.

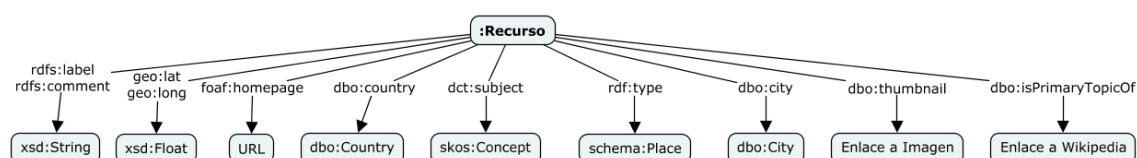


Figura 32. Modelo ontológico para la publicación de datos enlazados geo-referenciados.

Fuente: El Autor.

Elaboración: El Autor.

A través del sistema nacional de información y el ministerio de educación, se recopilaban cerca de 160 datos correspondientes a instituciones educativas geo-referenciadas de nivel secundario de la provincia de Loja, consolidando un archivo separado por comas (csv); para a través de un programa escrito en JAVA utilizando el framework Jenna (ver Anexo 11. Programa generador de tripletas.),estructurar la información en un formato RDF según el modelo ontológico propuesto; las tripletas generadas fueron publicadas en un servidor Local utilizando OpenLink virtuoso como triplestore y consumidos por la aplicación web en el mismo servidor.

La Figura 33 muestra la visualización de los datos enlazados correspondientes a las instituciones Educativas de nivel secundario de la provincia de Loja dentro de la aplicación web ubicada en el servidor Local.

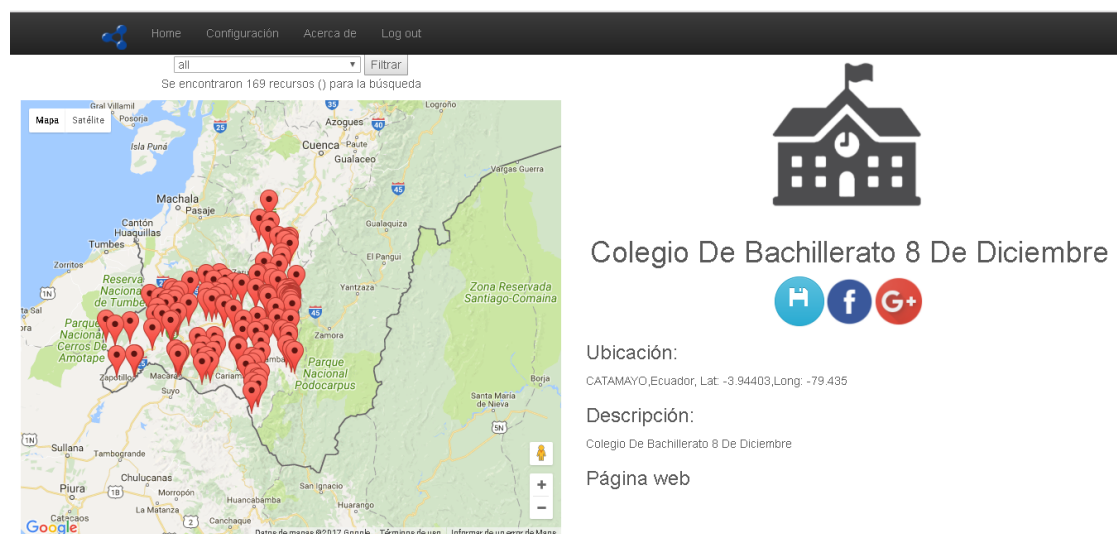


Figura 33. Visualización de instituciones educativas de nivel secundario en Loja Ecuador en aplicación web.

Fuente: El Autor.

Elaboración: El Autor.

4.3. Comentarios finales del capítulo.

Luego de haberse realizado las pruebas definidas por la metodología SCRUM se pudo determinar que las aplicaciones se comportan de la manera esperada y cumplen las expectativas definidas por el cliente en las “Historias de Usuario”.

Al observar la baja cantidad de recursos recuperados al realizar consultas de países Latino Americanos con el servicio web de consulta, se propuso un modelo ontológico para incentivar la publicación de datos enlazados geo-referenciados; el modelo propuesto fue validado con la creación, publicación y consumo de tripletas en un servidor local

CONCLUSIONES

Una vez realizado el proyecto se han llegado a las siguientes conclusiones:

- Se creó un servicio web para facilitar el consumo de información geo-referenciada, con métodos que permiten la búsqueda utilizando coordenadas o el nombre de una ciudad o país.
- Se implementó un servicio web para el almacenamiento de recursos favoritos de un usuario en una base de datos en la nube.
- Fue creada una aplicación móvil para sistemas operativos Android que consume los servicios web, ofreciendo una interfaz para realizar el consumo, visualización y almacenamiento de los recursos de una manera sencilla; sin necesidad que el usuario conozca la sintaxis de SPARQL.
- Fue desarrollada una aplicación web que hace uso de los servicios web de consumo y almacenamiento creados, ofreciendo al usuario una interfaz para un consumo, almacenamiento y visualización sencilla; haciendo las consultas SPARQL invisibles para el usuario.
- En los dispositivos móviles, obtener las coordenadas usando GPS es más preciso que obtenerlas utilizando la red; sin embargo, esta puede tomar más tiempo en determinar la posición.
- Dada la basta naturaleza de los datos enlazados publicados, fue necesario crear consultas que puedan abarcar la mayor cantidad de resultados utilizando propiedades básicas de RDF/RDFS y ontologías como: dbpedia-owl y geo.
- Para mejorar la relevancia de los resultados se crearon filtros que utilizan las coordenadas de un recurso, la ciudad o país a la que pertenece y las categorías en donde está incluido el recurso.
- Fuentes de datos como DBpedia, DBpedia Español y DBpedia Latinoamérica poseen una cantidad relativamente pequeña de información geo-referenciada sobre Ecuador.
- Se planteó un modelo para la publicación de datos enlazados geo-referenciados, validando el modelado con datos de prueba provenientes de un archivo .csv que contenía datos sobre instituciones Educativas de nivel secundario del Ecuador.

RECOMENDACIONES

Se recomienda:

- Implementar métodos que permitan una búsquedas en múltiples datasets tales como: DBpedia, DBpedia Español y DBpedia Latinoamérica.
- Realizar una planificación de las actividades del proyecto utilizando herramientas como WBS schedule pro.
- Construir consultas SPARQL con las propiedades lat y long del vocabulario Geo para consumir recursos geo referenciados.
- Utilizar los servicios web incluidos en los endpoints de los datasets de DBpedia para facilitar el consumo de información.
- Utilizar el API de Google Maps para una visualización de información geo-referenciada, siendo los recursos geo-referenciados marcadores en un mapa interactivo.
- Involucrar a los interesados en el proyecto para obtener una mejor definición de las características del software y su evolución.
- Realizar análisis de código estático con aplicaciones como Sonar Qube para obtener métricas que permitan evaluar la calidad del software desarrollado.
- Mejorar la seguridad de las aplicaciones implementando mecanismos de autenticación por medio de cuentas de correo como Google+ o Facebook.
- Utilizar el entorno de desarrollo ANDROID Studio para la construcción de aplicaciones móviles ANDROID debido a que facilita el desarrollo y ofrece herramientas como un gestor de dependencias, diseñador de interfaces y simuladores de diferentes versiones del sistema operativo ANDROID para realizar pruebas.

BIBLIOGRAFÍA

- Android Developers. (2016). Transmitting Network Data Using Volley | Android Developers. Retrieved from <https://developer.android.com/training/volley/index.html>
- Battle, R., & Benson, E. (2008). Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST): Semantic Web and Web 2.0. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(1), 61–69. <https://doi.org/10.1016/j.websem.2007.11.002>
- Bizer, C., Lehmann, J., Isele, R., & Jakob, M. (2012). DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia, 1, 29.
- Bray, T. (2014). The JavaScript Object Notation (JSON) Data Interchange Format, 1–16.
- EllisLab. (2017). CodeIgniter. Retrieved June 20, 2017, from <https://ellislab.com/codeigniter>
- Esther, M., Educational, E., Domingue, J., Haase, P., Maleshkova, M., Mikroyannidis, A., ... Vidal, M. E. (2014). EUCLID | EdUcational Curriculum for the usage of Linked Data. Retrieved from <http://www.euclid-project.eu/>
- Flanagan, D. (2007). *JavaScript*. Anaya Multimedia. Retrieved from <https://dialnet.unirioja.es/servlet/libro?codigo=316551>
- Gongolidis, V., Kommata, M., Papantoniou, A., & Loumos, V. (2012). Semantic driven location services in the web of data. In *Proceedings of the 2012 16th Panhellenic Conference on Informatics, PCI 2012* (pp. 7–12). IEEE. <https://doi.org/10.1109/PCI.2012.17>
- Google Developers. (2016). Introducción a la Google Maps Android API | Google Maps Android API | Google Developers. Retrieved from <https://developers.google.com/maps/documentation/android-API/intro?hl=es-419>
- Gosling, James; Joy, Bill; Steele, Guy; Bracha, Gilad; Buckley, A. (2014). The JAVA® Language Specification - jls8.pdf. *Addison-Wesley*, 688. Retrieved from <https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf>
- Hendrik, & Hendratmo, A. R. T. (2015). Linked data-driven geographic mobile learning application. *Proceedings 2015 International Conference on Science and Technology, TICST 2015*, 254–259. <https://doi.org/10.1109/TICST.2015.7369367>
- Hervalejo, E., Martínez-Prieto, M. A., Fernández, J. D., & Corcho, O. (2014). HDTourist: Exploring urban data on android. *CEUR Workshop Proceedings*, 1272, 65–68.
- International Data Corporation. (2016). IDC: Smartphone OS Market Share 2016, 2015. Retrieved January 30, 2017, from <http://www.idc.com/promo/smartphone-market-share/os>
- Lukovnikov, D., Stadler, C., Kontokostas, D., Hellmann, S., & Lehmann, J. (2014). DBpedia viewer-An integrative interface for DBpedia leveraging the DBpedia service eco system. *CEUR Workshop Proceedings*, 1184.
- Piedra, N., Chicaiza, J., Lopez, J., Tovar, E., & Martinez-Bonastre, O. (2012). Combining Linked Data and mobiles devices to improve access to OCW. In

- Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1–7). IEEE. <https://doi.org/10.1109/EDUCON.2012.6201202>
- Quezada, E. (2016). *Investigación y análisis de aplicaciones, para proponer un mejor consumo y visualización de datos*. Universidad Técnica Particular de Loja.
- Saha, a. (2008). A Developer's First Look At Android. *Linux For You*, (January 2008), 48–50.
- Schwaber, ken, & Sutherland, J. (2013). The Scrum Guide™ The Definitive Guide to Scrum: The Rules of the Game. Retrieved from <http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf>
- Schwaber, K. (n.d.). SCRUM Development Process. Retrieved from <http://www.jeffsutherland.org/oopsla/schwapub.pdf>
- Shoaibi, D. A. Al, & Rassan, I. A. Al. (2012). Mobile Advertising Using Location Based Services. *2012 IEEE First International Conference on Internet Operating Systems*, 13–16. <https://doi.org/10.1109/ICIOS.2012.15>
- Tessem, B., Johansen, B., & Veres, C. (n.d.). Mobile Location-Driven Associative Search in DBpedia.pdf, 6–10.
- The PHP Group. (2001). PHP: ¿Qué es PHP? - Manual. Retrieved November 16, 2016, from <http://PHP.net/manual/es/intro-what-is.PHP>
- Vagliano, I., Marengo, M., & Morisio, M. (2015). DBpedia Mobile Explorer. *2015 IEEE 1st International Forum on Research and Technologies for Society and Industry, RTSI 2015 - Proceedings*, 181–185. <https://doi.org/10.1109/RTSI.2015.7325095>
- Veres, C. (2012). MapXplore: Linked data in the app store. *CEUR Workshop Proceedings*, 905.
- World Wide Web Consortium (W3c) España. (2016). SOAP Version 1.2 Parte 0: Fundamentos. Retrieved November 30, 2016, from <http://www.w3c.es/Traducciones/es/TR/2003/REC-soap12-part0-20030624/#L1153>
- World Wide Consortium. (2016). Guía Breve de Linked Data. Retrieved from <http://www.w3c.es/Divulgacion/GuiasBreves/LinkedData>
- World Wide Web Consortium. (2016). Sparql Lenguaje de Consulta para RDF.
- World Wide Web Consortium (W3c) España. (2016). Guía Breve de Servicios Web. Retrieved November 30, 2016, from <http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>

ANEXOS

Anexo 1. Diccionario de datos

Este anexo contiene una descripción detallada de las tablas, campos y características de los elementos de la base de datos con el fin de dar a entender el contexto de la información con la que interactúan las aplicaciones Web y Móvil.

Datasets: Es un catálogo de información que contiene todas las fuentes de datos a partir de las cuales se pueden ejecutar consultas.

Columna	Tipo de dato	Null	Llave	Descripción
id (<i>Primary</i>)	int(11)	No	PK	Identificador del dataset.
acr	varchar(10)	No		Acrónimo del dataset.
nombre	varchar(100)	No		Nombre del dataset.
uri	varchar(300)	No		Dirección del dataset.

Lenguajes: Es un catálogo de lenguajes en los cuales la información puede ser consultada y desplegada.

Columna	Tipo de dato	Null	Llave	Comentario
id (<i>Primary</i>)	int(11)	No	PK	Identificador de lenguaje.
acr	varchar(5)	No		Acrónimo del lenguaje.
nombre	varchar(50)	No		Nombre del lenguaje.

Recursos: Almacena información correspondiente a los recursos favoritos de un usuario.

Columna	Tipo de dato	Null	Llave	Descripción
id (<i>Primary</i>)	int(11)	No	PK	Identificador de recurso.
id_usuario	int(11)	No	FK	Identificador de usuario al que pertenece este recurso.
recurso	varchar(500)	No		Identificador de recurso único del recurso.
id_dataset	int(11)	No	FK	Identificador del dataset al que pertenece.

Usuarios: Almacena información sobre la cuenta de usuario y lo enlaza con las configuraciones definidas por el mismo.

Column	Type	Null	Llave	Descripción
id (<i>Primary</i>)	int(11)	No	PK	Identificador de recurso
google_id	varchar(32)	No		Identificador de la cuenta de google de un usuario.
id_dataset	int(11)	No	FK	Identificador del dataset elegido por este usuario.
id_lang	int(11)	No	FK	Identificador del lenguaje elegido por este usuario.

UserView: Es una vista creada para hacer más fácil la consulta de las configuraciones del usuario.

Column	Type	Null	Llave	Descripción
google_id	varchar(32)			Identificador de la cuenta de google de un usuario.
lenguaje	varchar(5)			Lenguaje preferido por el usuario.
dataset	varchar(10)			Dataset preferido por el usuario.

Anexo 2. Historias de Usuario.

Este anexo incluye las necesidades recopiladas del usuario para definir las características de la aplicación y los criterios para validar el funcionamiento de una determinada característica.

ID de la historia	Enunciado de la historia				Criterios de aceptación			
	Rol	Característica / Funcionalidad	Razón / Resultado	N. Escenario	Criterio de aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
00-000-0001	Como un usuario	Necesito consumir información geo-referenciada.	Con la finalidad de conocer mi entorno.	1	Ubicación Actual	En caso que se requiera conocer los recursos relacionados a mi ubicación.	Cuando se haga clic en un botón.	El sistema debe realizar una consulta a un dataset, utilizando mi ubicación actual.
				2	Buscar en una ciudad	En caso que se requiera conocer los recursos relacionados a una ciudad.	Cuando se ingrese el nombre de la ciudad.	El sistema debe realizar una consulta a un dataset, utilizando el nombre de la ciudad.
				3	Buscar en un país	En caso que se requiera conocer los recursos relacionados a un país.	Cuando se ingrese el nombre de un país.	El sistema debe realizar una consulta a un dataset, utilizando el nombre del país.
				4	Filtrar información	En caso que se requiera consultar tipos específicos de recursos	Cuando se seleccione una categoría	El sistema debe realizar consultas de tipos de recursos específicos a un dataset.
				5	Múltiples fuentes de información	En caso que se desee realizar una consulta en un conjunto de datos diferente.	Cuando se realice una consulta.	El sistema permita seleccionar entre conjuntos de datos diferentes como: DBpedia, DBpedia Español y DBpedia Latinoamérica.

Enunciado de la historia					Criterios de aceptación			
ID de la historia	Rol	Característica / Funcionalidad	Razón / Resultado	N. Escenario	Criterio de aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
00-000-0002	Como un usuario	Necesito almacenar mis recursos favoritos.	Con la finalidad de visualizarlos posteriormente.	1	Almacenamiento	En caso que se desee guardar un recurso como favorito	Cuando se haga clic en un botón.	El sistema debe almacenar dicho recurso en una base de datos.
				2	Visualización posterior	En caso que se desee consultar recursos favoritos.	Cuando se realice una consulta de favoritos	El sistema debe mostrar los recursos favoritos del usuario.
00-000-0003	Como un usuario	Necesito visualizar la información en mapas	Con la finalidad de entender la información.	1	Visualización en mapas	En caso de realizar una consulta.	Cuando se muestren los resultados	El sistema debe desplegar un mapa con los recursos en forma de marcadores.
				2	Descripción del recurso	En caso de requerir ver en detalle un recurso.	Cuando se haga clic en un marcador	El sistema debe mostrar una ventana con la información del recurso como: imagen, descripción, ubicación, página web y las categorías a las que pertenece.
00-000-0004	Como un usuario	Necesito realizar las consultas desde mi Smartphone	Con la finalidad de consumir la información desde cualquier lugar.	1	Sistema Operativo Android	En caso que se requiera consultar recursos desde un Smartphone Android.	Para realizar las consultas desde un Smartphone	El sistema debe contar con una aplicación móvil compatible con sistemas operativos Android 4.4 kitkat o superior.
00-000-0005	Como un usuario	Necesito realizar las consultas desde una página web.	Con la finalidad de consumir la información desde un navegador.	1	Página Web	En caso que quiera consultar los recursos desde un navegador web.	Para realizar consultas desde un navegador web	El sistema debe disponer de una interfaz web.

Enunciado de la historia					Criterios de aceptación			
ID de la historia	Rol	Característica / Funcionalidad	Razón / Resultado	N. Escenario	Criterio de aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
00-000-0006	Como un usuario	Necesito loguearme al sistema utilizando mi cuenta de Google.	Con la finalidad de evitar crear cuentas adicionales.	1	Sistema Operativo Android	En caso que se requiera consultar recursos desde un Smartphone Android.	Para utilizar la aplicación móvil.	Desplegar una interfaz de logueo con cuenta de Google.
				2	Página Web	En caso que se requiera consultar los recursos desde un navegador web.	Para utilizar la aplicación web.	Desplegar una interfaz de logueo con cuenta de Google.
00-000-0007	Como un usuario	Necesito compartir recursos en las redes sociales.	Con la finalidad de dar a conocer un recursos favorito a contactos de una red social.	1	Compartir en redes sociales	En caso que se requiera compartir un recurso	Cuando se haga click en el botón compartir	El sistema permita compartir un recurso en diferentes redes sociales como (facebook,Google+,twitter).

Anexo 3. Product Backlog.

Este anexo corresponde a una lista priorizada de las características de la aplicación obtenidas a partir de las “Historias de Usuario”.

Identificador (ID) de la Historia	Enunciado de la Historia	Alias	Estado	Prioridad
00-000-0001	Como un usuario, necesito consumir información geo-referenciada, con la finalidad de conocer mi entorno	C001	HECHO	ALTA
00-000-0002	Como un usuario, necesito almacenar mis recursos favoritos, con la finalidad de visualizarlos posteriormente.	A001	HECHO	MEDIA
00-000-0003	Como un usuario, necesito visualizar información en mapas, con la finalidad de entender la información	V001	HECHO	ALTA
00-000-0004	Como un usuario, necesito realizar las consultas desde mi Smartphone, con la finalidad de consumir la información desde cualquier lugar.	C002	HECHO	ALTA
00-000-0005	Como un usuario, necesito realizar las consultas desde una página web, con la finalidad de consumir la información desde un navegador.	C003	HECHO	MEDIA
00-000-0006	Como un usuario, loguearme al sistema utilizando mi cuenta de google, con la finalidad de evitar crear cuentas adicionales.	A002	HECHO	MEDIA
00-000-0007	Como un usuario, necesito compartir recursos en las redes sociales, con la finalidad de compartir mis recursos favoritos	A003	HECHO	MEDIA

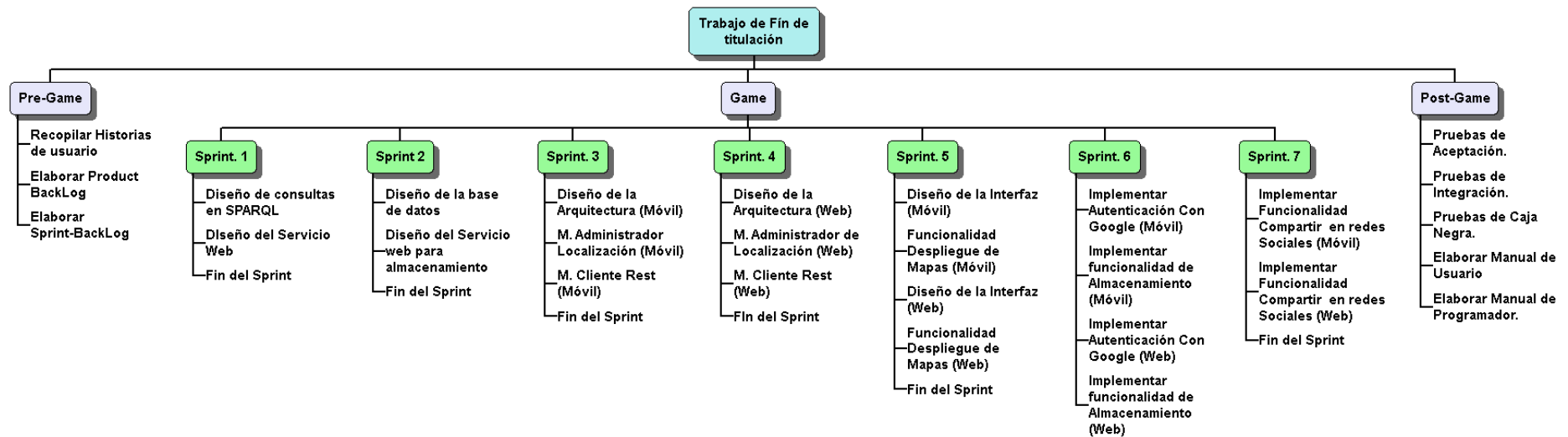
Anexo 4. Sprint Backlog.

Este anexo incluye una lista de las tareas necesarias para el desarrollo de las funcionalidades definidas en el artefacto “Product Backlog” además de una estimación en días de cada una de las tareas.

ID de item de Product Backlog	Tarea	Estatus	Días estimados totales
C001	Diseño de consultas en SPARQL.	HECHO	5
	Diseño del servicio web.	HECHO	5
A001	Diseño de la base de datos.	HECHO	1
	Diseño de servicio web para almacenamiento.	HECHO	2
C002	Diseño de la arquitectura.(MÓVIL)	HECHO	5
	Desarrollo M. AdministradorLocalización (MÓVIL)	HECHO	5
	Desarrollo M. Cliente REST (MÓVIL)	HECHO	5
C003	Diseño de la arquitectura.(WEB)	HECHO	5
	Desarrollo M. AdministradorLocalización (WEB)	HECHO	5
	Desarrollo M. Cliente REST (WEB)	HECHO	5
v001	Diseño de la interfaz (MÓVIL)	HECHO	5
	Despliegue de mapas. (MÓVIL)	HECHO	5
	Diseño de la interfaz (WEB)	HECHO	5
	Despliegue de mapas. (WEB)	HECHO	5
A002	Implementar autenticación con google (MÓVIL)	HECHO	5
	Implementar funcionalidad almacenamiento (MÓVIL)	HECHO	5
	Implementar autenticación con google (WEB)	HECHO	5
	Implementar funcionalidad almacenamiento (WEB)	HECHO	5
A003	Implementar funcionalidad compartir con redes sociales.(MÓVIL)	HECHO	5
	Implementar funcionalidad compartir con redes sociales.(WEB)	HECHO	5

Anexo 5. Estructura desagregada de Tareas en WBS Schedule Pro.

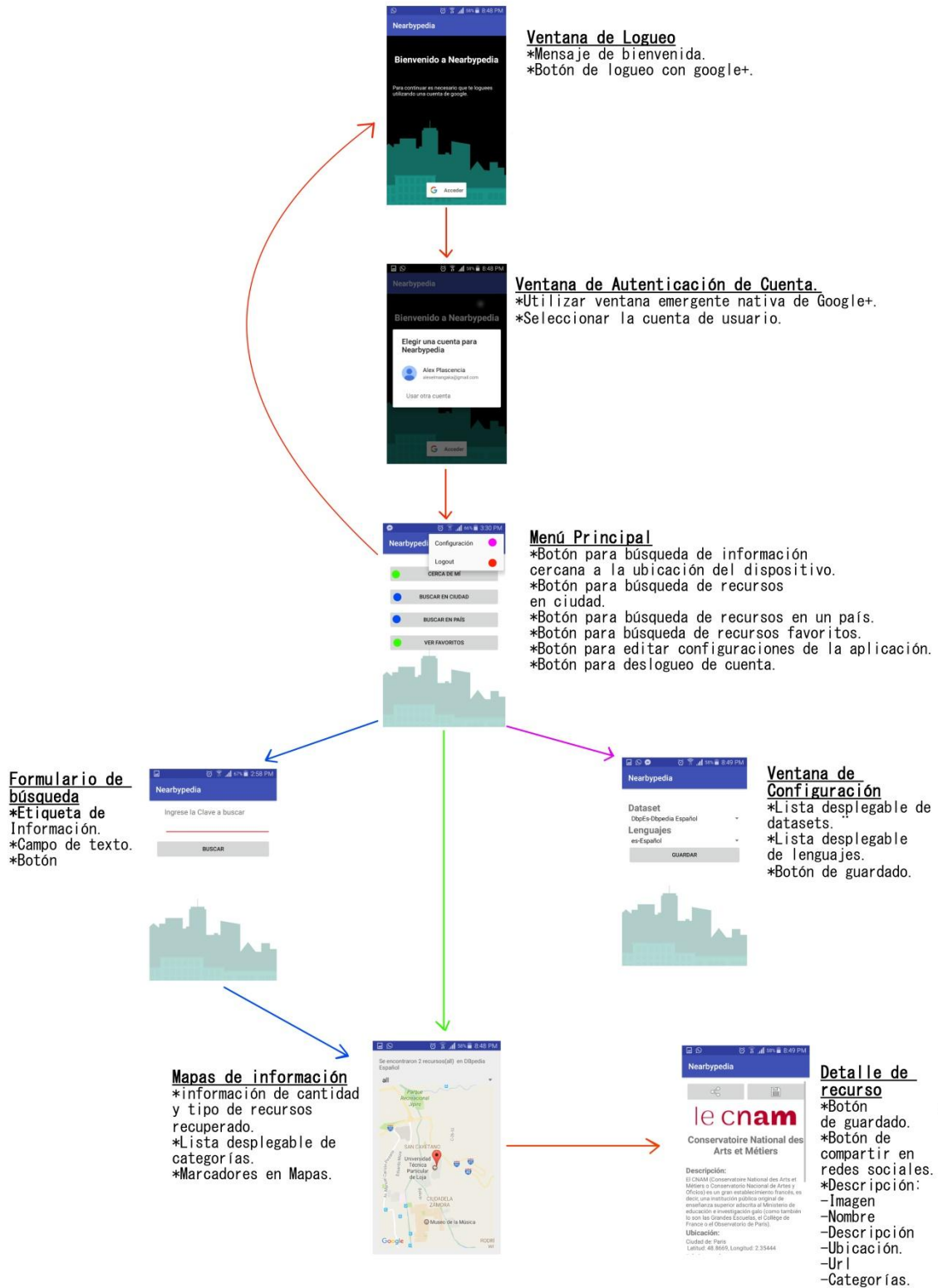
Este anexo incluye una estructura desagregada de tareas construida con el software WBS Schedule Pro, ubicando cada una de las fases de SCRUM y ordenando las actividades definidas en el artefacto “Sprint Backlog” en cada uno de los Sprints; este anexo incluye también una planificación más detallada acerca del tiempo que fue necesario para el desarrollo del trabajo de fin de titulación.



WBS	Nombre	Duración	Fecha Inicio	Fecha Fin
1	Trabajo de Fín de titulación	117d	30/12/2016	12/6/2017
1.1	Pre-Game	4d	30/12/2016	4/1/2017
1.1.1	Recopilar Historias de usuario	2d	30/12/2016	2/1/2017
1.1.2	Elaborar Product Backlog	1d	3/1/2017	3/1/2017
1.1.3	Elaborar Sprint-BackLog	1d	4/1/2017	4/1/2017
1.2	Game	100d	5/1/2017	24/5/2017
1.2.1	Sprint. 1	8d	5/1/2017	16/1/2017
1.2.1.1	Diseño de consultas en SPARQL	5d	5/1/2017	11/1/2017
1.2.1.2	Diseño del Servicio Web	5d	10/1/2017	16/1/2017
1.2.1.3	Fin del Sprint	0d	16/1/2017	16/1/2017
1.2.2	Sprint 2	3d	17/1/2017	19/1/2017
1.2.2.1	Diseño de la base de datos	1d	17/1/2017	17/1/2017
1.2.2.2	Diseño del Servicio web para almacenamiento	2d	18/1/2017	19/1/2017
1.2.2.3	Fin del Sprint	0d	19/1/2017	19/1/2017
1.2.3	Sprint. 3	24d	20/1/2017	22/2/2017
1.2.3.1	Diseño de la Arquitectura (Móvil)	5d	20/1/2017	26/1/2017
1.2.3.2	M. Administrador Localización (Móvil)	5d	2/2/2017	8/2/2017
1.2.3.3	M. Cliente REST (Móvil)	5d	16/2/2017	22/2/2017
1.2.3.4	Fin del Sprint	0d	22/2/2017	22/2/2017
1.2.4	Sprint. 4	15d	23/2/2017	15/3/2017
1.2.4.1	Diseño de la Arquitectura (Web)	5d	23/2/2017	1/3/2017
1.2.4.2	M. Administrador de Localización (Web)	5d	2/3/2017	8/3/2017
1.2.4.3	M. Cliente REST (Web)	5d	9/3/2017	15/3/2017
1.2.4.4	Fin del Sprint	0d	15/3/2017	15/3/2017
1.2.5	Sprint. 5	20d	16/3/2017	12/4/2017
1.2.5.1	Diseño de la Interfaz (Móvil)	5d	16/3/2017	22/3/2017
1.2.5.2	Funcionalidad Despliegue de Mapas (Móvil)	5d	23/3/2017	29/3/2017
1.2.5.3	Diseño de la Interfaz (Web)	5d	30/3/2017	5/4/2017
1.2.5.4	Funcionalidad Despliegue de Mapas (Web)	5d	6/4/2017	12/4/2017
1.2.5.5	Fin del Sprint	0d	12/4/2017	12/4/2017
1.2.6	Sprint. 6	20d	13/4/2017	10/5/2017
1.2.6.1	Implementar Autenticación Con Google (Móvil)	5d	13/4/2017	19/4/2017
1.2.6.2	Implementar funcionalidad de Almacenamiento (Móvil)	5d	20/4/2017	26/4/2017
1.2.6.3	Implementar Autenticación Con Google (Web)	5d	27/4/2017	3/5/2017
1.2.6.4	Implementar funcionalidad de Almacenamiento (Web)	5d	4/5/2017	10/5/2017
1.2.7	Sprint. 7	10d	11/5/2017	24/5/2017
1.2.7.1	Implementar Funcionalidad Compartir en redes Sociales (Móvil)	5d	11/5/2017	17/5/2017
1.2.7.2	Implementar Funcionalidad Compartir en redes Sociales (Web)	5d	18/5/2017	24/5/2017
1.2.7.3	Fin del Sprint	0d	24/5/2017	24/5/2017
1.3	Post-Game	13d	25/5/2017	12/6/2017
1.3.1	Pruebas de Aceptación.	3d	25/5/2017	29/5/2017
1.3.2	Pruebas de Integración.	3d	30/5/2017	1/6/2017
1.3.3	Pruebas de Caja Negra.	3d	2/6/2017	6/6/2017
1.3.4	Elaborar Manual de Usuario	2d	7/6/2017	8/6/2017
1.3.5	Elaborar Manual de Programador.	2d	9/6/2017	12/6/2017

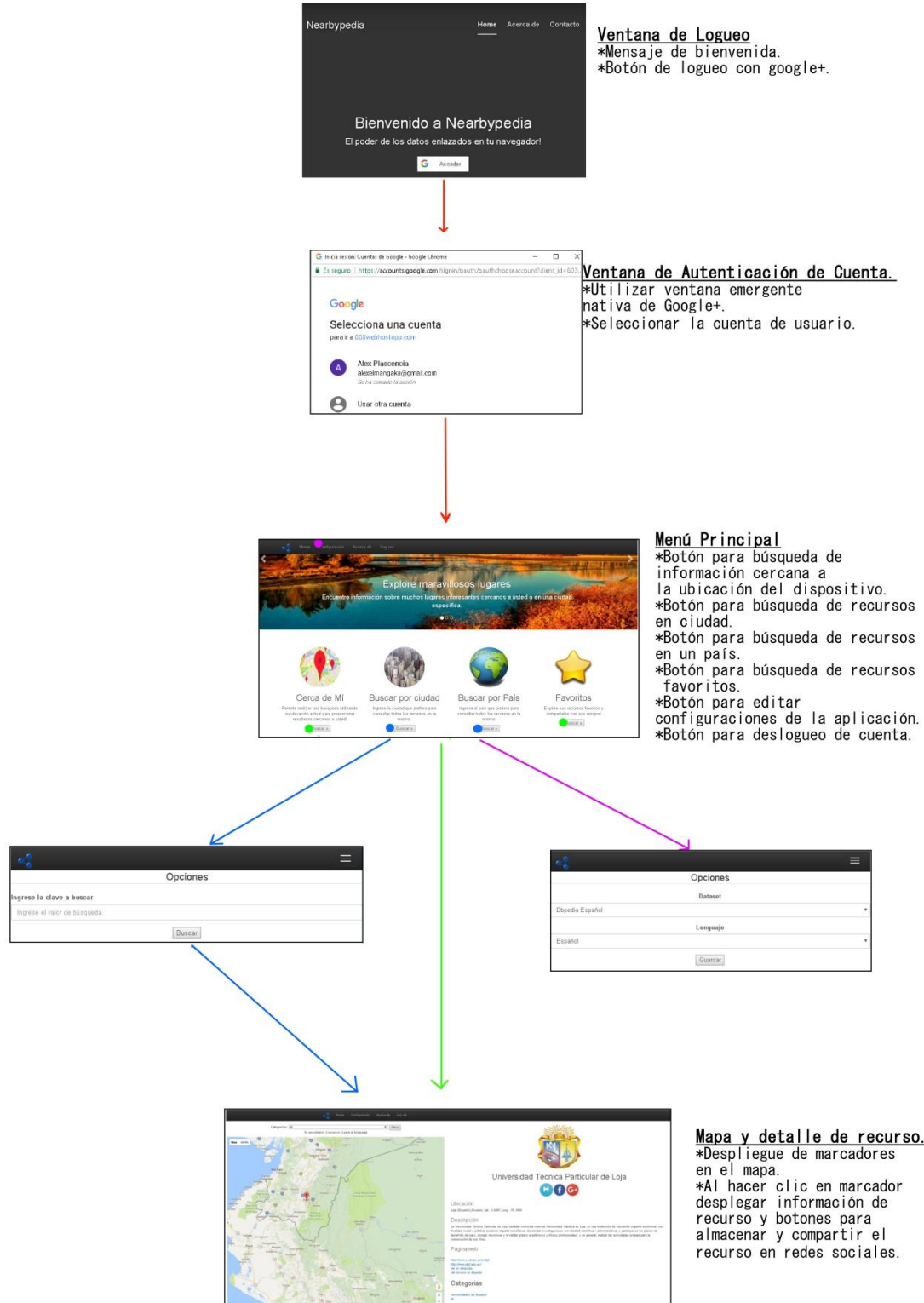
Anexo 6. Diseño de la interfaz Móvil.

En este anexo se describen los elementos de cada una de las interfaces y los flujos entre las interfaces están representados por colores.



Anexo 7. Diseño de la interfaz Web.

Este anexo muestra los elementos y flujos de las interfaces con las que interactúa el usuario para el consumo, visualización y almacenamiento de datos enlazados geo-referenciados.



Anexo 8. Pruebas de Aceptación de usuario.

Este anexo incluye las pruebas a través de las cuales se busca validar el funcionamiento de las características de la aplicación con los criterios de aceptación definidos en las “Historias de Usuario”.

Funcionalidad:	Criterios de Aceptación	Fase de pruebas	Pruebas previstas	Resultados esperados	Resultados Obtenidos	Paso/Reprobó
Consumir información geo-referenciada	Búsqueda por ubicación Actual	Búsqueda por coordenadas en dispositivo móvil.	3	N recursos cercanos.	1 recurso cercano/prueba	Pasó
		Búsqueda por coordenadas en navegador.	3	N recursos cercanos.	1 recurso cercano/prueba	Pasó
	Búsqueda por ciudad	Búsqueda en ciudad por dispositivo móvil.	Loja Quito Guayaquil Cuenca	N recursos cercanos.	1 recurso. 3 recursos. 3 recursos. 0 recursos	Pasó
		Búsqueda en ciudad por navegador.	Loja Quito Guayaquil Cuenca	N recursos cercanos.	1 recurso. 3 recursos. 3 recursos. 0 recursos.	Pasó
	Búsqueda por país	Búsqueda por país en dispositivo móvil	Ecuador Argentina Chile Brasil	N recursos/ prueba.	8 Recursos. 98 Recursos. 26 Recursos. 17 Recursos.	Pasó
		Búsqueda en país por navegador.	Ecuador Argentina Chile Brasil	N recursos/prueba	8 Recursos. 98 Recursos. 26 Recursos. 17 Recursos.	Pasó

Funcionalidad:	Criterios de Aceptación	Fase de pruebas	Pruebas previstas	Resultados esperados	Resultados Obtenidos	Paso/Reprobó
Consumir información geo-referenciada	Múltiples fuentes de información	Búsqueda de recursos en web y móvil.	Búsqueda de recursos cercanos a ubicación: DBpedia DBpedia Español DBpedia latinoamérica	N recursos cercanos.	1 recurso Web/Móvil 1 recurso Web/Móvil 1 recurso Web/Móvil	Pasó
	Filtro de información	Filtro por categorías	5 categorías en Quito	N recursos filtrados por tipo.	2 Red Ecuatoriana de Universidades 2 Universidades del Ecuador. 2 Universidades de Quito. 2 Universidades de Pichincha. 1 Organización fundada en 1869.	Pasó
Almacenar Recursos Favoritos	Almacenamiento	Almacenar n recursos desde aplicación móvil	5	N recursos en base de datos.	5 recursos Almacenados	Pasó
		Almacenar n recursos desde navegador web	5	N recursos en base de datos.	5 recursos Almacenados	Pasó
	Visualización Posterior	Desplegar recursos favoritos de usuario en dispositivo móvil	1	N recursos favoritos desplegados	10 recursos Desplegados	Pasó
		Desplegar recursos favoritos de usuario en navegador web	1	N recursos favoritos desplegados	10 recursos Desplegados	Pasó
Visualizar Información en mapas	Visualización del mapa	Despliegue de mapa en dispositivo móvil	1	Despliegue correcto	Correcto	Pasó
		Despliegue de mapa en navegador web	1	Despliegue correcto	Correcto	Pasó

Funcionalidad:	Criterios de Aceptación	Fase de pruebas	Pruebas previstas	Resultados esperados	Resultados Obtenidos	Paso/Reprobó
Visualizar Información en mapas	Descripción de recursos	Descripción de recurso en dispositivo móvil	5	Nombre Ubicación Descripción Página web Enlace a Wikipedia Categorías a las que pertenece	Todos los recursos consultados disponen de toda o parte de la información, en caso de no disponer de algún dato este es reemplazado con un link predeterminado de no data	Pasó
		Descripción de recurso en navegador web	5	Nombre Ubicación Descripción Página web Enlace a Wikipedia Categorías a las que pertenece	Todos los recursos consultados disponen de toda o parte de la información, en caso de no disponer de algún dato este es reemplazado con un link predeterminado de no data	Pasó
Compartir recursos en redes sociales	Compartir a través de dispositivo móvil	Compartir recurso en twitter.	3	Tweet creado.	3 tweets publicados.	Pasó
		Compartir recurso en facebook	3	Post Facebook	Post publicados pero enlaces bloqueados posteriormente.	Reprobó
		Compartir recurso en google plus	3	Post Google+	3 Posts Publicados.	Pasó
	Compartir a través de navegador web	Compartir recurso en twitter.	3	Tweet creado.	3 tweets publicados.	Pasó
		Compartir recurso en facebook	3	Post Facebook	Post publicados pero enlaces bloqueados posteriormente.	Reprobó
		Compartir recurso en Google plus	3	Post Google+	3 Posts Publicados.	Pasó

Anexo 9. Pruebas de Integración.

Este anexo incluye pruebas para determinar si el comportamiento entre los módulos de la aplicación son los correctos, para ello se plantean casos de prueba involucrando a varios módulos a la vez.

Número del Caso de Prueba	Componente	Descripción de lo que se Probará	Prerrequisitos
CA001	Consumo - Visualización (Móvil)	Visualización de recursos consumidos a través del servicio web.	Módulos terminados
CA002	Consumo - Almacenamiento (Móvil)	Almacenamiento de recursos consumidos a través del servicio web	Módulos terminados
CA003	Almacenamiento - Visualización (Móvil)	Visualización de recursos favoritos almacenados en base de datos.	Módulos terminados y N recursos Favoritos Almacenados
CA004	Consumo - Visualización (Web)	Visualización de recursos consumidos a través del servicio web.	Módulos terminados
CA005	Consumo - Almacenamiento (Web)	Almacenamiento de recursos consumidos a través del servicio web	Módulos terminados
CA006	Almacenamiento - Visualización (Web)	Visualización de recursos favoritos almacenados en base de datos.	Módulos terminados y N recursos Favoritos Almacenados

CA001					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Realizar búsqueda con ubicación Actual, desde dispositivo móvil.	Latitud, Longitud, Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
2	Realizar búsqueda ingresando una ciudad, desde dispositivo móvil.	Ciudad,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	

CA002					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Realizar búsqueda con ubicación actual desde dispositivo móvil.	Latitud, Longitud, Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
2	Almacenar recurso favorito de usuario.	Click en la interfaz	Recurso al macenado en base de datos	SI	
3	Realizar búsqueda ingresando una ciudad desde dispositivo móvil.	Ciudad,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
4	Almacenar recurso favorito de usuario.	Click en la interfaz	Recurso al macenado en base de datos	SI	

CA003					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Almacenar N recursos favoritos para usuario desde dispositivo móvil	id_google,uri del recurso, dataset	N filas creadas en base de datos.	SI	
2	Realizar búsqueda de favoritos, desde dispositivo móvil.	id_google,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	

CA004					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Realizar búsqueda con ubicación Actual, desde navegador web.	Latitud, Longitud, Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
2	Realizar búsqueda ingresando una ciudad, desde navegador web.	Ciudad,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	

CA005					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Realizar búsqueda con ubicación actual desde navegador web.	Latitud, Longitud, Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
2	Almacenar recurso favorito de usuario.	Click en la interfaz	Recurso al macenado en base de datos	SI	
3	Realizar búsqueda ingresando una ciudad desde navegador web.	Ciudad,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	
4	Almacenar recurso favorito de usuario.	Click en la interfaz	Recurso al macenado en base de datos	SI	

CA006					
Paso	Descripción de pasos a seguir	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Almacenar N recursos favoritos para usuario desde navegador web.	id_google,uri del recurso, dataset	N filas creadas en base de datos.	SI	
2	Realizar búsqueda de favoritos, desde navegador web.	id_google,Dataset,Lenguaje,Tipo	N recursos desplegados en mapa.	SI	

Anexo 10. Métricas e indicadores analizados por SonarQube.

En este anexo contiene una descripción detallada de que métricas e indicadores que son evaluados por SonarQube para determinar la calidad del software.

Medida	Significado	Métricas	Indicadores
Reliability	Determina la confiabilidad de un sistema.	Número de Bugs	0 Bugs. Al menos 1 bug menor. Al menos 1 bug mayor. Al menos 1 bug Crítico. Al menos 1 bug Bloqueador.
Security	Determina que tan seguro es un software.	Número de vulnerabilidades.	0 Vulnerabilidades. Al menos 1 Vulnerabilidad menor. Al menos 1 Vulnerabilidad mayor. Al menos 1 Vulnerabilidad Crítica. Al menos 1 vulnerabilidad Bloqueadora.
Maintainability	Determina que tan mantenible es un software.	Code Smells: Indican deficiencias en el diseño que puede ralentizar el desarrollo o aumentan el riesgo de errores o fallos en el futuro. Deuda Técnica: Costo por desarrollo de línea de código.	La calificación es dada en función de la deuda técnica, el costo para corregir errores del tiempo total de desarrollo: >=5% del tiempo total. 6% al 10% 11% al 20% 21% a 50% 51% en adelante.
Duplications	Determina la cantidad de código duplicado presente en el proyecto.	Bloques duplicados: Cada 10 líneas de código sucesivo duplicado. Líneas duplicadas: Cada línea de código duplicada	$\text{densidad de duplicación} = \frac{\text{Lineas de código duplicado}}{\text{Lineas de código Totales}} * 100$
Code coverage	Cuánto código ha sido cubierto por pruebas unitarias automatizadas.	No Implementado.	

Anexo 11. Programa generador de tripletas.

Este anexo incluye un programa para generar tripletas geo referenciadas en RDF con un formato "N-TRIPLES" a partir de un archivo separados por comas (.csv).

```
package clase.jena.pkg1;
import JAVA.io.BufferedReader;
import org.apache.jena.rdf.model.Model;
import org.apache.jena.rdf.model.ModelFactory;
import org.apache.jena.rdf.model.Resource;
import org.apache.jena.vocabulary.RDF;
import org.apache.jena.vocabulary.DCTerms;
import org.apache.jena.vocabulary.RDFS;
import org.apache.jena.rdf.model.RDFWriter;
import JAVA.io.File;
import JAVA.io.FileInputStream;
import JAVA.io.FileNotFoundException;
import JAVA.io.FileOutputStream;
import JAVA.io.IOException;
import JAVA.io.InputStreamReader;
import org.apache.jena.datatypes.xsd.XSDDatatype;
import org.apache.jena.rdf.model.ResourceFactory;
import org.apache.jena.vocabulary.SKOS;
/**
 *
 * @author Alex Plascencia
 */
public class ClaseJena1 {
    public static void main(String[] args) throws FileNotFoundException {
        //Creación de un modelo vacio
        Model model = ModelFactory.createDefaultModel();
        File f= new File ("OUT/data.rdf"); //Fijar ruta donde se creará el archivo RDF
        FileOutputStream os = new FileOutputStream(f);

        //Fijar Prefijo para URI base de datos a crear
        String dataPrefix = "http://localhost:8890/resource/";
        model.setNsPrefix("myData",dataPrefix);

        //Fijar prefijos de vocabularios incorporados en Jena
        //DCTerms Usado para definir las categorias de los recursos.
        String dcterms = "http://purl.org/dc/terms/";
        model.setNsPrefix("dcterms",dcterms);

        //Fijar Prefijo para otros vocabularios como DBpedia que no están directamente incorporados en Jena.
        //Incorporar Ontología de DBpedia.
        String dbo = "http://dbpedia.org/ontology/";
        model.setNsPrefix("dbo", dbo);

        //Incorporar geo utilizado para la geo-referenciación.
        String geo = "http://www.w3.org/2003/01/geo/wgs84_pos#";
        model.setNsPrefix("geo", geo);
    }
}
```

```
// en el caso de vocabularios externos (no incorporados en Jena) se debe crear un modelo
Model dboModel = ModelFactory.createDefaultModel(); // modelo para la ontología de dbpedia
dboModel.read(dbo);
```

```
Model geoModel = ModelFactory.createDefaultModel(); // modelo para la ontología geo
geoModel.read(geo,"RDFXML");
```

```
//Crear una categoría general para todos los recursos.
```

```
Resource categoria=model.createResource(dataPrefix+"Colegios_de_la_provincia_de_Loja")
    .addProperty(RDF.type, SKOS.Concept)
    .addProperty(RDFS.label, "Colegios de la provincia de Loja");
```

```
BufferedReader br = null;
```

```
try {
```

```
    //Leer Archivo csv fuente
```

```
    br = new BufferedReader(
        new InputStreamReader(new FileInputStream("IN/fuente.csv"), "UTF-8"));
```

```
    String line;
```

```
    br.readLine();
```

```
    while ((line = br.readLine()) != null) {
```

```
        String[] data=line.split(";");
```

```
        String uriname=data[0];
```

```
    //Crear un recurso
```

```
    Resource colegio01
```

```
    = model.createResource(dataPrefix+uriname)
```

```
        .addProperty(RDF.type, dboModel.getResource(dbo+"EducationalInstitution"))
```

```
        .addProperty(RDFS.label, model.createLiteral(data[2], "es"))
```

```
        .addProperty(RDFS.comment, model.createLiteral(data[2], "es"))
```

```
.addProperty(geoModel.getProperty(geo+"lat"),ResourceFactory.createTypedLiteral(data[3],XSDDatatype.X
SDfloat))
```

```
.addProperty(geoModel.getProperty(geo+"long"),ResourceFactory.createTypedLiteral(data[4],XSDDatatype
.XSDfloat))
```

```
        .addProperty(dboModel.getProperty(dbo+"city"),
```

```
            model.createResource(dataPrefix+data[6])
```

```
            .addProperty(RDFS.label, model.createLiteral(data[6], "es"))
```

```
            .addProperty(RDF.type, dboModel.getResource(dbo+"City"))
```

```
        )
```

```
        .addProperty(dboModel.getProperty(dbo+"country"),
```

```
dboModel.getResource("http://dbpedia.org/resource/Ecuador"))
```

```
        .addProperty(DCTerms.subject, categoria)
```

```
        .addProperty(dboModel.getProperty(dbo+"thumbnail"), data[7])
```

```
        .addProperty(DCTerms.subject,
```

```
            model.createResource(dataPrefix+"Colegios_" +data[8])
```

```
            .addProperty(RDF.type, SKOS.Concept)
```

```
            .addProperty(RDFS.label, "Colegios " +data[9]));
```

```
    }
```

```
} catch (IOException e) {
```

```
    e.printStackTrace();
```

```
} finally {
```

```
    try {
```

```
        if (br != null) {
```

```

        br.close();
    }
} catch (IOException ex) {
    ex.printStackTrace();
}
}

// Escribir modelo RDF
System.out.println("MODELO RDF-----");
//Seleccionar formato de Salida
model.write(System.out, "NTRIPLES");

// Almacenar en archivo.
RDFWriter writer = model.getWriter("NTRIPLES"); //RDF/XML
writer.write(model,os, "");

//Cerrar modelos
dboModel.close();
geoModel.close();
model.close();
}
}

```

Nearbypedia Manual de Usuario

Autor: Alex Daniel Plascencia Arévalo

Versión: 1

Fecha: 22/05/201

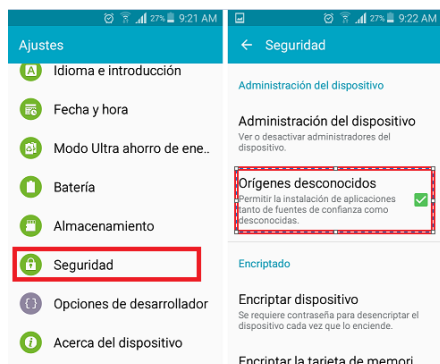
1. Introducción.

Este manual le permitirá aprender a utilizar todas las funcionalidades básicas de Nearbypedia tanto de la aplicación móvil, así como del portal en internet.

2. Aplicación Móvil

2.1. Descargar la Aplicación.

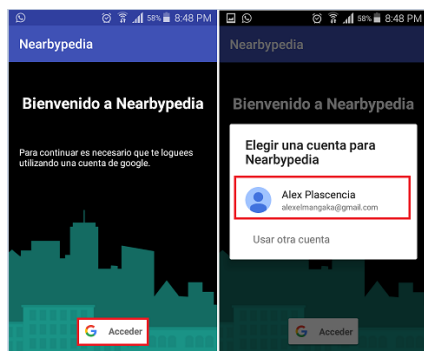
Para hacer uso de la aplicación es necesario descargarla en su dispositivo móvil desde el siguiente enlace https://mega.nz/#!u48FySxC!NEEV2PRJb5tsLg_LxHRYa6X5j3M-0wXycT3MZX-04K0.



Una vez descargado el archivo .apk, deberá activar la instalación desde orígenes desconocidos para ello debe ir a los **ajustes** del sistema, luego a **seguridad** y por último activar la pestaña **Orígenes desconocidos**.

Posteriormente solo se tiene que abrir el archivo .apk y seleccionar instalar.

2.2. Logeo al Sistema.



Ahora que ya tiene instalada la aplicación para acceder a la funcionalidad es necesario que usted pulse el botón **Acceder** en la ventana, a través del cual puede utilizar su cuenta de Google.

2.3. Consultar Recursos.

Una vez autenticado se desplegará el menú principal de la misma en el diagrama mostrado a continuación se encuentran definidos cada uno de los menús y opciones a los que se puede acceder.



A. Configuración: Permite entrar a la configuración de la aplicación, permite ajustar el conjunto de datos a utilizar y el idioma.

B. Logout: Permite cerrar su sesión actual y volver a la pantalla de inicio.

C. Cerca de mí: permite la búsqueda de recursos utilizando la localización actual (coordenadas) de su dispositivo.

D. Buscar en Ciudad: Permite la búsqueda de recursos, ingresando una ciudad.

E. Buscar en País: Permite la búsqueda de recursos, ingresando un país.

F. Ver Favoritos: Permite la visualización de todos sus recursos favoritos.

2.3.1. Consultar recursos utilizando coordenadas actuales.

Al pulsar el botón cerca de mí del menú principal, la aplicación procede a obtener las coordenadas actuales de su dispositivo por lo que es necesario que se tenga activa los servicios de ubicación, en caso de que no esté activada se despliega una ventana para activar los servicios de ubicación en donde solo es necesario pulsar el botón de activar.

Los resultados se desplegarán en un mapa (ver apartado **Mapa de recursos.**)

2.3.2. Consultar recursos utilizando una ciudad.

Al pulsar el botón buscar en ciudad del menú principal, se despliega una ventana en dónde se puede ingresar el nombre de una ciudad y luego pulsando el botón buscar, realiza la búsqueda de los recursos pertenecientes a esa ciudad.

Los resultados se despliegan en un mapa (ver apartado **Mapa de recursos.**)

2.3.3. Consultar recursos utilizando un país.

Al pulsar el botón buscar en país del menú principal, se despliega una ventana en dónde se puede ingresar el nombre de un país y luego pulsando el botón buscar, realiza la búsqueda correspondiente para dicho país.

Los resultados se despliegan en un mapa (ver apartado **Mapa de recursos.**)

2.3.4. Consultar recursos favoritos.

Con esta opción se visualizan todos los recursos favoritos del usuario haciendo una consulta utilizando su cuenta y recuperando todos los recursos almacenados como favoritos, para almacenar un recurso como favorito (ver apartado **Almacenar recursos Favoritos.**).



2.4. Mapa de recursos.



Una vez realizada una consulta se despliega los recursos como marcadores en un mapa, adicionalmente se muestra una lista desplegable con todas las categorías encontradas en la búsqueda, al seleccionar una de estas categorías se visualizan todos los recursos pertenecientes a dicha categoría, dejando en **all** se visualizan todos los recursos de todas las categorías.

Al pulsar sobre un marcador se despliega algo de información del mismo en una ventana de información.

Al pulsar la ventana de información se despliega una ventana para visualizar el recurso a detalle incluyendo:


- Menú de opciones del recurso.
 - Compartir recurso.  (Ver apartado **Compartir recursos en redes sociales.**).
 - Almacenar recurso.  (Ver apartado **Almacenar**)



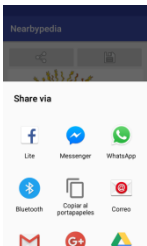
recursos Favoritos.).


- Imagen del recurso.
- Descripción del recurso.
- Localización: Ciudad, País y coordenadas.
- Enlaces a páginas web del recurso.
 - Propio del recurso.
 - Descripción en Wikipedia.
 - Descripción de recurso en dataset.
- Categorías a las que pertenece el recurso.

2.5. Almacenar recursos Favoritos.

Para almacenar un recurso como favorito cuando esté en la vista en detalle de un recurso, se debe presionar el botón .

2.6. Compartir recursos en redes sociales.



En la vista de detalle de un recurso pulsando el botón , se despliega una ventana en donde se puede seleccionar como se desea compartir el recurso utilizando redes sociales, mensajería, etc.

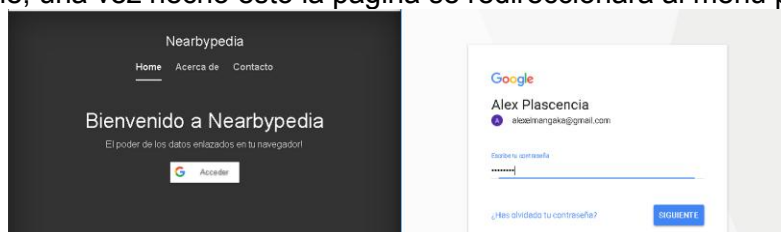
3. Aplicación Web.

3.1. Acceder al portal web.

Desde su navegador web tiene que ir a la siguiente dirección web:
<https://nearbypedia.000webhostapp.com>.

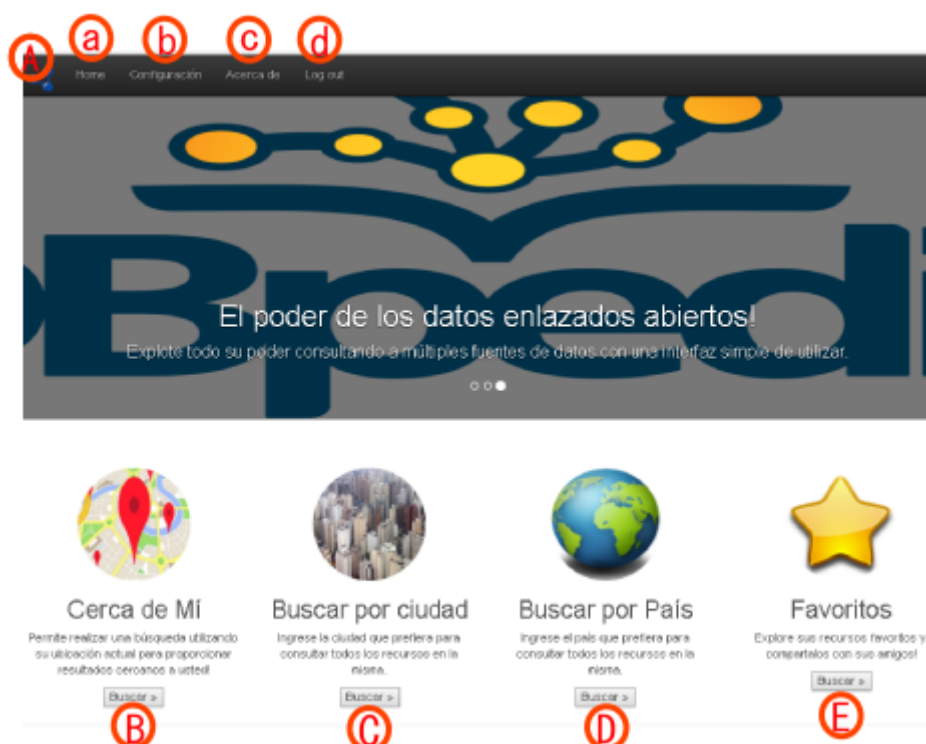
3.2. Logueo al Sistema.

Una vez en el sitio web en la ventana principal debe hacer click en el botón acceder y seleccionar su cuenta de google, una vez hecho esto la página se redireccionará al menú principal.



3.3. Consultar Recursos.

El menú principal se encuentra descrito en el siguiente diagrama.



- A. Menú de Opciones.
 - a. Home. Permite regresar al menú principal.
 - b. Configuración: Permite entrar a la configuración de la aplicación, permite ajustar el conjunto de datos a utilizar y el idioma.
 - c. Acerca de: Permite ingresar a una página en donde se indica una descripción de la aplicación, propósito, etc enlaces de descarga de aplicación móvil y manual de usuario.
 - d. Logout: Permite cerrar su sesión actual y volver a la pantalla de inicio.
- B. Cerca de mí: permite la búsqueda de recursos utilizando su localización actual (coordenadas)..
- C. Buscar en Ciudad: Permite la búsqueda de recursos, ingresando una ciudad.
- D. Buscar por país: Permite la búsqueda de recursos, ingresando un país.
- E. Ver Favoritos: Permite la visualización de todos sus recursos favoritos.

3.3.1. Consultar recursos utilizando coordenadas actuales.

Al pulsar el botón cerca de mí del menú principal, la aplicación procede a obtener las coordenadas actuales por lo que se mostrará un mensaje para permitir el uso de la ubicación es necesario dar clic en permitir.

Los resultados se desplegarán en un mapa (ver apartado **Mapa de recursos.**)

3.3.2. Consultar recursos utilizando una ciudad.

Al pulsar el botón buscar en ciudad del menú principal, se despliega una ventana en dónde se puede ingresar el nombre de una ciudad y luego pulsando el botón buscar, realiza la búsqueda de los recursos pertenecientes a esa ciudad.

Los resultados se despliegan en un mapa (ver apartado **Mapa de recursos.**)

3.3.3. Consultar recursos utilizando un país.

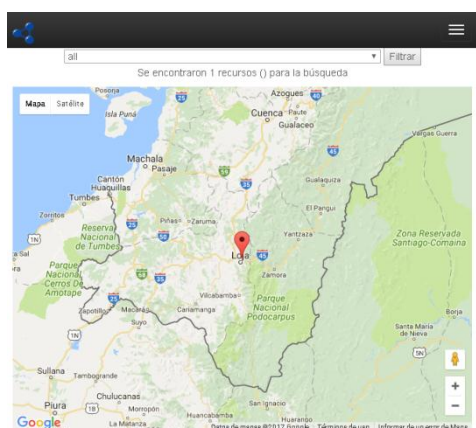
Al pulsar el botón buscar en país del menú principal, se despliega una ventana en dónde se puede ingresar el país y luego pulsando el botón buscar, realiza la búsqueda de los recursos pertenecientes a ese país.

Los resultados se despliegan en un mapa (ver apartado **Mapa de recursos.**)

3.3.4. Consultar recursos favoritos.

Con esta opción se visualizan todos los recursos favoritos del usuario haciendo una consulta utilizando su cuenta y recuperando todos los recursos almacenados como favoritos, para almacenar un recurso como favorito (ver apartado),

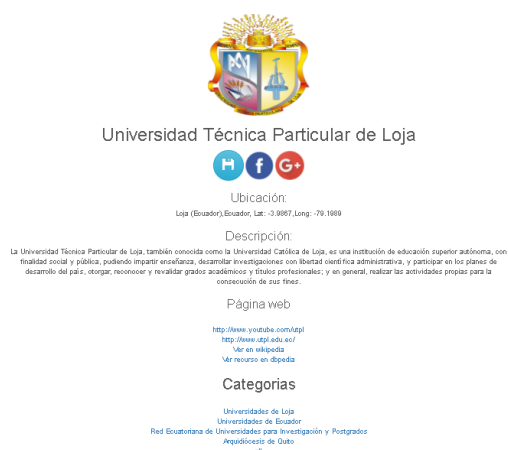
3.4. Mapa de recursos.




Una vez realizada una consulta se despliega los recursos como marcadores en un mapa, adicionalmente se muestra una lista desplegable con todas las categorías encontradas en la búsqueda, al seleccionar una de estas categorías se visualizan todos los recursos pertenecientes a dicha categoría, dejando en **all** se visualizan todos los recursos de todas las categorías.

Al pulsar sobre un marcador se despliega una ventana para visualizar el recurso a detalle incluyendo:



- Imagen del recurso.
- Nombre del recurso.
- Almacenar recurso como favorito.
- Compartir en Facebook.
- Compartir en Google+.
- Localización: Ciudad, País y coordenadas.
- Enlaces a páginas web del recurso.
 - Propio del recurso.
 - Descripción en Wikipedia.
 - Descripción de recurso en dataset.
- Categorías a las que pertenece el recurso.



3.5. Almacenar recursos Favoritos.

Para almacenar un recurso como favorito cuando esté en la vista en detalle de un recurso, se debe presionar el botón .

3.6. Compartir recursos en redes sociales.

En la vista de detalle de un recurso pulsando el botón  o , se despliegan las interfaces propias de Facebook o Google+ para compartir contenido.

4. Preguntas frecuentes.

4.1. Aplicación móvil

4.1.1. ¿No se puede instalar la aplicación?

Recuerde activar instalar desde orígenes desconocidos desde la pestaña seguridad en el menú de opciones del dispositivo.

4.1.2. ¿No se puede autenticar a la aplicación móvil?

Recuerde comprobar que su dispositivo se encuentra conectado a internet.

4.1.3. ¿Por qué una consulta no entrega resultados?

Algunas veces ciertas consultas no entregan resultados esto se debe a que no se encuentren disponibles recursos para la consulta como tal, para resolver eso puede cambiar el dataset y/o el idioma para la búsqueda desde el menú configuración de la aplicación, adicionalmente puede utilizar buscar por ciudad desde el menú principal; asegúrese de que la ciudad esté escrita de manera correcta con las tildes necesarias en el caso que lo requiera.

4.1.4. ¿Por qué se despliega un mensaje de error al realizar una consulta?

Compruebe si su dispositivo se encuentra conectado a internet, muchas veces una consulta puede tardar más de lo normal por la congestión y desplegar estos errores, entonces también puede realizar su consulta nuevamente unos minutos después.

4.1.5. ¿Por qué no se despliegan todos los recursos favoritos?

Recuerde que cada recurso favorito está ligado a un dataset específico, es decir que si almacena un recurso A en un dataset D1 y cambia el dataset a D2 se despliegan sus favoritos solo para el dataset D2.

4.2. Aplicación Web

4.2.1. No se puede acceder al sitio Web.

Compruebe su conexión a internet, si el problema continúa el servidor puede que este caído, en ese caso espere unas horas y el sistema estará disponible una vez más.

4.2.2. No se puede realizar consultas con coordenadas.

Recuerde haber pulsado permitir geo-localización en el mensaje que aparece en la parte superior del navegador en caso de que haya presionado no permitir no será capaz de utilizar dicha funcionalidad.

4.2.3. ¿Por qué aparece un mapa vacío al realizar una búsqueda?

Si aparece un mapa vacío es porque no se encontraron recursos relativos a la consulta realizada, adicional al mapa vacío se muestra un mensaje en pantalla indicando que no existen recursos disponible.

4.2.4. ¿Por qué no se puede ver recursos favoritos de un usuario?

Consulte el apartado 4.1.5.

Nearbypedia

Manual de Programador

Autor: Alex Daniel Plascencia Arévalo

Versión: 1

Fecha: 22/05/201

1. Introducción.

Este manual tiene el propósito de guiar a un nuevo desarrollador en el proceso de familiarización del software, incluyendo requerimientos de entorno, herramientas.

2. Servicio Web.

2.1. Requerimientos de Entorno:

Servidor Web	Apache
Base de datos:	MYSQL
Versión PHP	5.5 o Superior

El sitio web se encuentra alojado en la nube en www.nearbypedia.000webhostapp.com, para acceder a la configuración del mismo se lo puede realizar a través del siguiente enlace <https://www.000webhost.com/cpanel-login>

2.2. Herramientas Utilizadas:

Editor de Texto	Sublime Text 3
Cliente FTP	Filezilla

2.2.1. Configuración de Cliente FTP.

Servidor	files.000webhostapp.com
Usuario	Nearbypedia
Contraseña	*****
Puerto	21

2.3. Servicio web de consulta:

Para hacer uso de este servicio web es necesario ingresar un valor para cada uno de los campos definidos, estos valores pueden ser constantes o variables <<>> que se obtienen del usuario, este servicio se encuentra localizado en el archivo /REST/query.PHP

Campo	Valores	Descripción
Método	Coor	Permite realizar búsqueda por ubicación actual (Latitud,longitud).
	Ciudad	Permite realizar la búsqueda por una ciudad.
	Fav	Permite obtener los recursos favoritos de un usuario.
	Share	Permite obtener la descripción de un recurso específico para ser compartido.
	Pais	Permite la bpusqueda de recursos por país.
Param	<<Latitud,longitud>>	Utilizado para realizar la búsqueda por ubicación actual.
	<<Ciudad>>	Utilizado para realizar la búsqueda por la ciudad ingresada.
	<<Google_id>>	Utilizado para buscar recursos favoritos de un usuario.
	<<Uri>>	Uri de un recurso a ser descrito.
Dataset	Dbp	Data set sobre el cuál se realizará la búsqueda.

	DbpEs Latam	
Lang	es en	Lenguaje para presentación de resultados.
Tipo	All <<Categoría>>	Categoría usada para filtro de resultados.

2.3.1. Cambiar parámetros de consulta.

En el caso de que se necesite obtener información diferente acerca de los recursos es necesario modificar la variable \$query con la nueva consulta en SPARQL en el servicio web.

```

query.php
<?php
//=====
//SERVICIO WEB DE CONSULTA DE RECURSOS
//=====

//Establecer el formato de salida del servicio.
header('content-type: application/json;');
//Importar configuraciones para la base de datos.
require('config.php');
//Definición de Consulta Base
$query="
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?desc ?lat ?long ?res ?city ?country ?img ?wiki
(group_concat(distinct ?p;separator=',') as ?page)
(group_concat(distinct ?t;separator=',') as ?type)
WHERE {
  ?res rdfs:label ?name ;
    geo:lat ?lat ;
    dbo:country ?countryo;
    geo:long ?long ;
    dbo:city ?cityo ;
    rdfs:comment ?desc ;
    dct:subject ?to.
  ?to rdfs:label ?t.
  ?countryo rdfs:label ?country.
  ?cityo rdfs:label ?city.
  OPTIONAL { ?res foaf:isPrimaryTopicOf ?wiki}.
  OPTIONAL { ?res dbo:thumbnail ?img}.
  OPTIONAL { ?res foaf:homepage ?p}.";
```

2.3.2. Agregar filtro de consulta.

Un filtro de consulta permite filtrar los resultados de una consulta realizada por el servicio web, los filtros definidos actualmente son:

Tipo de filtro	Método	Uso
Lenguaje	langfilter	Permite filtrar el resultado de las consultas por idioma.
Tipo	typefilter	Permite filtrar el resultado de las consultas por categorías.

Para agregar un nuevo filtro se tiene que crear un nuevo método en el servicio web e incluir dentro de él la lógica que aplica dicho filtro sobre la consulta base en lenguaje SPARQL.

2.3.3. Agregar método de consulta.

Una consulta está compuesta por:

Parte	Variable o Método	Explicación
Consulta base	\$query	Esta consulta base define los atributos obtenidos para cada recurso consultado.

Filtro Específico	querylocation()	Permite filtrar los recursos obtenidos cercanos una coordenada.
	querybycity()	Permite filtrar los recursos obtenidos que pertenezcan a una ciudad específica.
	sharer()	Permite buscar un recurso específico con su uri.
	querybyfavourite()	Recupera de la base de datos las uris de los recursos favoritos de un usuario y realiza una búsqueda con esas uris.
Filtros Generales	langfilter()	Filtro de resultados por lenguaje.
	Typefilter()	Filtro de resultados por tipo de recurso.
Agrupación de términos	\$group	Corresponde a la parte final de la consulta en este se definen las agrupaciones de los resultados y el límite de resultados que se puede obtener por consulta.

Entonces para agregar un nuevo método de consulta, basta con crear un método con otro filtro específico y llamar a dicho método desde la estructura switch, utilizando el parámetro método con un nombre específico para la nueva consulta y recibiendo los parámetros necesarios para armar dicha consulta.

```
//Obtener los parámetros de consulta
switch ($_GET['metodo']) {
    case 'coord':
        //Crear la url de consulta
        $parametro=explode(",", $_GET['param']);
        $requestURL =querylocation($parametro[0],$parametro[1],$_GET['dataset'],$_GET['lang'],$_GET['tipo']);
        break;
    case 'fav':
        //Crear la url de consulta
        $requestURL =querybyfavorite($_GET['param'],$_GET['dataset'],$_GET['lang'],$_GET['tipo']);
        break;
    case 'ciudad':
        //Crear la url de consulta
        $requestURL =querybycity($_GET['param'],$_GET['dataset'],$_GET['lang'],$_GET['tipo']);
        break;
    case 'share':
        //Crear la url de consulta
        $requestURL =sharer($_GET['param'],$_GET['dataset'],'es');
        break;
    default:
        header($_SERVER["SERVER_PROTOCOL"]." 404 Not Found", true, 404);
        break;
}
```

Cabe destacar que cada uno de los métodos genera una url de consulta apuntando a una uri de un dataset almacenado en la base de datos, con la consulta creada a partir de la consulta base aplicando los filtros necesarios y definiendo un formato de respuesta el cuál por defecto es JSON, ya que es más fácil operar con este formato de datos en ambas aplicaciones.

```
function querylocation($lat,$long,$dataset,$Lang,$tipo){
    $myquery=$GLOBALS['query']."
    FILTER(?long > ".$long." - 0.05 && ?long < ".$long." + 0.05 &&
    ?lat > ".$lat." - 0.05 && ?lat < ".$lat." + 0.05)
    ".langfilter($lang)."".typefilter($type).
    "}".$GLOBALS['group'];
    //Armar url de Consulta
    $searchUrl = get_dataset_uri($dataset)
    .'query='.urlencode($myquery)
    .'&format=json';
    return $searchUrl;
}
```

2.4. Servicio web de almacenamiento:

En el siguiente cuadro se mencionan los campos y valores que utiliza el servicio web, de igual manera pueden ser valores constantes o variables <<>>, este servicio se encuentra localizado en el archivo **/REST/persistence.PHP**

Campo	Valores	Descripción
Metodo	CREATE	Permite crear un usuario en la base de datos, además de una configuración por defecto (DBpedia-latinoamerica, Español).
	INSERT	Al ingresar un recurso y un dataset, lo busca en la base de datos, si no lo encuentra lo almacena como recurso favorito, caso contrario elimina el recurso.
	UCONF	Permite actualizar la configuración actual de un usuario (Dataset y Lenguaje).
	GCATALOGS	Permite Obtener el listado de los Datasets y lenguajes disponibles.
User	<<Google_id>>	Identificador de google de un usuario.
Rec	<<Uri de recurso>>	Identificador de un recurso.
Dataset	Dbp DbpEs Latam	Valor de dataset a almacenar para configuración de usuario.
Lang	es en	Valor de lenguaje a almacenar para configuración de usuario.

2.4.1. Crear nuevo método.

Para crear un nuevo método es necesario crear las funciones que realicen las operaciones a la base de datos necesarias, y llamar estas funciones desde la estructura switch con el parámetro método, enviando los parámetros necesarios para ejecutar una operación en la base de datos, entregando una respuesta en formato JSON:

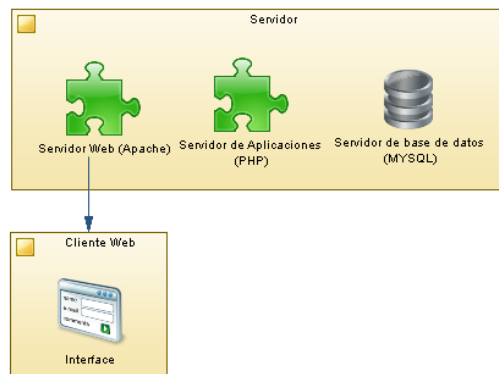
```

persistence.php
<?php
//SERVICIO WEB DE ALMACENAMIENTO DE RECURSOS
//=====
require('config.php');
//Establecer el formato de salida del servicio
header('Content-Type: application/json; charset=utf-8');
//Obtener los parametros de consulta
$method = $_GET['metodo'];
switch ($method) {
    case 'CREATE':
        $arr=create_user($_GET['user']);
        break;
    case 'INSERT':
        $arr=insert($_GET['user'],$_GET['rec'],$_GET['dataset']);
        break;
    case 'UCONF':
        $arr=updateconfiguration($_GET['user'],$_GET['dataset'],$_GET['lang']);
        break;
    case 'GCATALOGS':
        $arr = array();
        $arr['datasets']=get_dataset_catalog();
        $arr['lenguajes']=get_lang_catalog();
        break;
    default:
        header("Status: 404 Not Found");
        break;
}
}

```

3. Aplicación Web.

3.1. Arquitectura de Aplicación web.



3.1.1. Servidor web Apache.

Encargado de la transferencia de Hipertexto a través del protocolo HTTP, es utilizado para entregar los archivos que conforman una página web a un cliente, como una respuesta a una solicitud.

3.1.2. Servidor de Aplicaciones (PHP).

Proporciona los servicios de Aplicación a los clientes, en este se encuentran todos los módulos para el funcionamiento de la aplicación (Lógica del Negocio) y el acceso a datos.

3.1.3. Servidor de base de datos (MYSQL).

Este servidor se encarga de organizar y almacenar toda la información generada por las aplicaciones tanto web como móvil.

3.2. Requerimientos de Entorno:

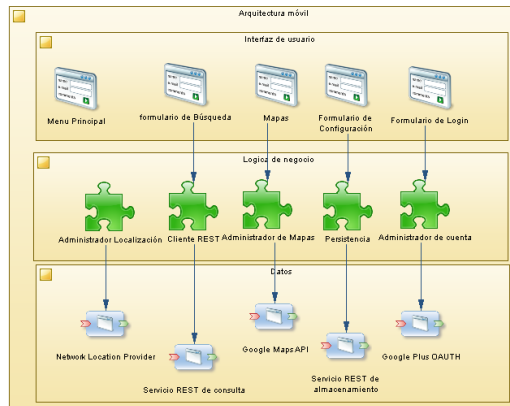
Servidor Web	Apache
Base de datos:	MYSQL
Versión PHP	5.5 o Superior

3.3. Herramientas Utilizadas:

Editor de Texto	Sublime Text 3
Cliente FTP	Filezilla
Servidor Local	Wamp

4. Aplicación Móvil.

4.1. Arquitectura de la Aplicación Móvil.



4.1.1.1. Módulo de Administrador localización.

Este módulo se encargará de todo lo relacionado con la ubicación del usuario teniendo como tareas principales como: la gestión de permisos de localización y determinar la posición actual.

4.1.1.2. Módulo de Cliente REST.

Este módulo es el encargado de la gestión de la información obtenida desde el web service en formato JSON, las principales tareas de este módulo son: gestionar la comunicación con el servicio web, obtener la información como un objeto JSON que se pueda interpretar y enviar la información a las clases que requieran hacer uso de estos objetos.

4.1.1.3. Módulo de Administrador Mapas.

Este es un módulo dedicado para la visualización de la información a través del uso de mapas, originando marcadores que indiquen la posición de cada uno de los resultados obtenidos en las búsquedas y ofrecer información sobre cada uno de ellos.

4.1.1.4. Módulo de persistencia

Este módulo es el encargado de almacenar toda la información correspondiente a los usuarios de la aplicación como la información de su cuenta y sus recursos favoritos, en una base de datos externa común para las versiones web y móvil.

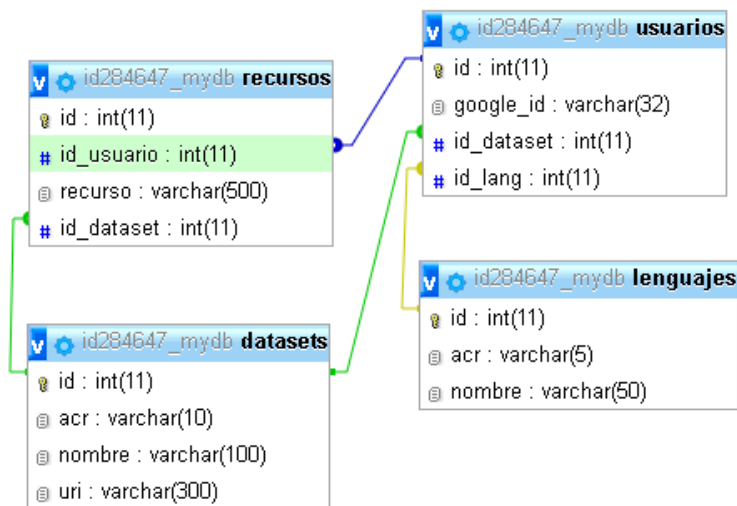
4.1.1.5. Módulo de Administrador de cuentas.

Este módulo está encargado de la gestión de las cuentas de usuario, realizando las operaciones de autenticación con una cuenta de Google, para hacer uso de esta aplicación.

5. Base de datos:

5.1. Estructura de la Base de datos.

La estructura de la base de datos se encuentra definida en la siguiente imagen:



5.2. Diccionario de datos.

Datasets: Es un catálogo de información que contiene todas las fuentes de datos a partir de las cuales se pueden ejecutar consultas.

Columna	Tipo de dato	Null	Llave	Comentario
id (<i>Primary</i>)	int(11)	No	PK	Identificador del dataset.
acr	varchar(10)	No		Acrónimo del dataset.
nombre	varchar(100)	No		Nombre del dataset.
uri	varchar(300)	No		Dirección del dataset.

Lenguajes: Es un catálogo de lenguajes en los cuales la información puede ser consultada y desplegada.

Columna	Tipo de dato	Null	Llave	Comentario
id (<i>Primary</i>)	int(11)	No	PK	Identificador de lenguaje.
acr	varchar(5)	No		Acrónimo del lenguaje.
nombre	varchar(50)	No		Nombre del lenguaje.

Recursos: Almacena información correspondiente a los recursos favoritos de un usuario.

Columna	Tipo de dato	Null	Llave	Comentario
id (<i>Primary</i>)	int(11)	No	PK	Identificador de recurso.
id_usuario	int(11)	No	FK	Identificador de usuario al que pertenece este recurso.
recurso	varchar(500)	No		Identificador de recurso único del recurso.
id_dataset	int(11)	No	FK	Identificador del dataset al que pertenece.

Usuarios: Almacena información sobre la cuenta de usuario y lo enlaza con las configuraciones definidas por el mismo.

Column	Type	Null	Default	Comments
id (<i>Primary</i>)	int(11)	No		Identificador de recurso
google_id	varchar(32)	No		Identificador de la cuenta de google de un usuario.
id_dataset	int(11)	No		Identificador del dataset elegido por este usuario.
id_lang	int(11)	No		Identificador del lenguaje elegido por este usuario.

UserView: Es una vista creada para hacer más fácil la consulta de las configuraciones del usuario

Column	Type	Null	Llave	Descripción
google_id	varchar(32)			Identificador de la cuenta de google de un usuario.
lenguaje	varchar(5)			Lenguaje preferido por el usuario.
dataset	varchar(10)			Dataset preferido por el usuario.