



UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA
La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN INFORMÁTICA

**Consumo y visualización de archivos RDF de la DBpedia en dispositivos
móviles de Android**

TRABAJO DE TITULACIÓN

AUTOR: Arciniegas Ayala, Cristian Vinicio

DIRECTOR: Ramírez Coronel, Ramiro Leonardo, Mgtr.

CENTRO UNIVERSITARIO QUITO - CARCELÉN

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>

Abril, 2015

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Magister.

Ramiro Leonardo Ramírez Coronel

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación “Consumo y visualización de archivos RDF de la DBpedia en dispositivos móviles de Android” realizado por Arciniegas Ayala Cristian Vinicio, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, julio de 2017

f.....

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

“Yo Arciniegas Ayala Cristian Vinicio, declaro ser autor del presente trabajo de titulación: Consumo y visualización de archivos RDF de la DBpedia en dispositivos móviles de Android, de la Titulación de Ingeniería en Informática siendo Mgtr. Ramírez Coronel Ramiro Leonardo director del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

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

f.....

Autor. Arciniegas Ayala Cristian Vinicio

Cedula. 1719654574

DEDICATORIA

Dedico este trabajo en primer lugar a Dios por ser siempre mi principal soporte, y por darme la inspiración y las fuerzas para nunca perder la fe a pesar de las adversidades y así llegar a culminar esta carrera.

A mi amada esposa e hija por ser siempre mi mayor motivación y por tenerme cariño y paciencia a pesar de que fueron muchas las veces que no pude dedicarles el tiempo que les correspondía.

A mis queridos padres que supieron poner en mí los valores necesarios para luchar en esta vida, por sus sacrificios realizados para darme lo necesario y porque siempre estuvieron dándome palabras de ánimo en momentos difíciles.

A mi hermana, familiares y amigos que gracias a su amistad y sus consejos, he podido apreciar lo valiosos que ustedes son en mi vida.

CRISTIAN ARCINIEGAS

AGRADECIMIENTO

A mi Dios, por permitirme llegar a este momento

A mi familia, por su paciencia y comprensión

A mis padres, por siempre creer mí

A mi tutor de tesis Mgtr. Ramiro Ramírez, por su ayuda y dirección

A mis profesores y a todo el personal de la UTP

A todos quienes de alguna forma hicieron posible la culminación de este trabajo.

CRISTIAN ARCINIEGAS

ÍNDICE DE CONTENIDO

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN.....	ii
DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS.....	iii
DEDICATORIA	iv
AGRADECIMIENTO	v
ÍNDICE DE CONTENIDO	vi
ÍNDICE DE FIGURAS.....	x
ÍNDICE DE TABLAS.....	xii
ÍNDICE DE ANEXOS.....	xiii
RESUMEN.....	14
ABSTRACT	15
INTRODUCCIÓN.....	16
CAPITULO I: Estado del Arte.....	18
1.1. Introducción.....	19
1.2. La web y su evolución	19
1.2.1. Creación de la web.....	19
1.2.2. Etapas evolutivas de la web	19
1.2.3. Diferencias principales entre la Web 2.0 y la Web 3.0	22
1.3. La Web 3.0 o Web Semántica	24
1.3.1. Introducción a la Web Semántica	24
1.3.2. Componentes de la Web Semántica.....	24
1.3.3. Linked Open Data y la Web Semántica: los principios de Linked Data	25
1.3.4. RDF (Resource Description Framework)	26
1.3.5. Serializaciones RDF	27
1.3.6. SPARQL (SPARQL Protocol and RDF Query Language)	31
1.3.7. Formas de Consultas SPARQL	32
1.3.8. DBpedia.....	34

1.3.9.	DBpedia: framework de extracción del conocimiento.....	35
1.3.10.	Acceso a la base de conocimientos de DBpedia	37
1.3.11.	Ventajas de la DBpedia.....	38
1.4.	Tecnología Móvil	39
1.4.1.	Sistema Operativo Android	39
1.4.2.	Arquitectura Android	39
1.4.3.	Desarrollo de Aplicaciones en Android	40
1.4.4.	Características de una Aplicación Móvil.....	41
1.4.5.	Tecnologías y herramientas complementarias.....	42
1.5.	Trabajos Relacionados	45
1.5.1.	Metodología de búsqueda	45
1.5.2.	Mobile Location-Driven Associative Search in DBpedia with Tag Clouds.....	45
1.5.3.	Mobile Facets Application.....	45
1.5.4.	A Self-medication web application built on Linked Open Data	46
1.5.5.	VOX System: a semantic embodied conversational agent exploiting Linked data	46
1.5.6.	Smart places: Multi-agent based Smart mobile virtual community management system	46
1.5.7.	DBpedia Viewer: An Integrative interface for DBpedia Leveraging the DBpedia Service Eco System.....	46
1.5.8.	Linked tag: image annotation using semantic relationship between image tags	47
1.5.9.	A framework to support educational decision making in mobile learning.....	47
1.5.10.	ONLI: an ontology-based system for querying DBpedia using natural language paradigm.	47
1.5.11.	Knowledge Structuring and Reuse System Using RDF for Supporting Scenario Generation	47
1.5.12.	Memo Graph: an ontology visualization tool for everyone	48
1.5.13.	Mobile services discovery framework using DBpedia and non-monotonic rules	48

1.5.14.	QueryGen: Semantic interpretation of keyword queries over heterogeneous information systems	48
1.5.15.	ReDyAI: A Dynamic Recommendation Algorithm based on Linked Data	48
1.5.16.	Short Query Expansion for Microblog Retrieval	49
CAPITULO II: Propuesta de la Solución.....		51
2.1.	Introducción	52
2.2.	Problemática.....	52
2.3.	Soluciones planteadas en trabajos relacionados	52
2.4.	Propuesta de solución	53
2.5.	Arquitectura de la solución.....	54
2.6.	Objetivos de la propuesta	55
2.6.1.	Objetivo General.....	55
2.6.2.	Objetivos Específicos	55
2.7.	Metodología de desarrollo	55
CAPITULO III: Desarrollo de la Solución.....		57
3.1.	Introducción	58
3.2.	Product Backlog	58
3.3.	Backlog.....	58
3.4.	Planificación y desarrollo de Sprint	60
3.4.1.	Sprint 1	61
3.4.2.	Sprint 2.....	64
3.4.3.	Sprint 3.....	68
3.4.4.	Sprint 4.....	71
3.4.5.	Sprint 5.....	74
3.4.6.	Sprint 6.....	77
3.4.7.	Sprint 7.....	81
CAPITULO IV: Pruebas e Implementación		86
4.1.	Introducción	87
4.2.	Sprint Review – Sprint Retrospective.....	87

4.3. Pruebas de aceptación	88
4.3.1. Prueba de aceptación 1	89
4.3.2. Prueba de aceptación 2	92
4.3.3. Prueba de aceptación 3	95
4.3.4. Prueba de aceptación 4	98
4.4. Pruebas de Carga.....	101
4.5. Pruebas de usabilidad y accesibilidad	105
4.6. Implementación	108
CONCLUSIONES	110
RECOMENDACIONES.....	111
BIBLIOGRAFÍA.....	112
ANEXOS.....	116

ÍNDICE DE FIGURAS

Figura 1. Evolución de la web.....	20
Figura 2. Arquitectura de la web semántica	25
Figura 3. Grafo RDF	27
Figura 4. Serialización RDF/XML	28
Figura 5. Serialización N-triples.....	28
Figura 6. Serialización N3.....	29
Figura 7. Serialización RDF/JSON	29
Figura 8. Serialización Turtle	30
Figura 9. Serialización RDFa	30
Figura 10. Consulta SPARQL	31
Figura 11. Framework de extracción del conocimiento DBpedia	36
Figura 12. Acceso a la DBpedia mediante la web	37
Figura 13. Arquitectura del SO Android	40
Figura 14. Estructura de un símbolo QR	42
Figura 15. Arquitectura de la propuesta	54
Figura 16. Burndown Chart - Sprint 1.....	61
Figura 17. Prototipos de las interfaces de usuario de la aplicación móvil	63
Figura 18. Burndown Chart - Sprint 2.....	64
Figura 19. Activity_index.....	65
Figura 20. Activity_about	66
Figura 21. Activity_keywords	66
Figura 22. Activity_queries_sparql	67
Figura 23. Burndown Chart - Sprint 3.....	68
Figura 24. Activity_qr_code.....	69
Figura 25. Archivo strings.xml.....	70
Figura 26. Fragmento del archivo styles.xml.....	70
Figura 27. Burndown Chart - Sprint 4.....	71
Figura 28. Implementación del servicio Lookup DBpedia.....	72
Figura 29. Recuperación de datos desde el archivo JSON	72
Figura 30. Implementación del visualizador Lookup DBpedia	73
Figura 31. Burndown Chart - Sprint 5.....	74
Figura 32. Implementación del servicio SPARQL Endpoint.....	75
Figura 33. Recuperación de datos en formato HTML.....	75
Figura 34. Implementación del visualizador SPARQL Endpoint	76

Figura 35. Burndown Chart - Sprint 6.....	78
Figura 36. Implementación de lector de códigos QR.....	79
Figura 37. Lectura de código QR y obtención de la consulta SPARQL.....	79
Figura 38. Implementación del servicio SPARQL Endpoint.....	79
Figura 39. Recuperación de datos desde el archivo RDF/XML.....	80
Figura 40. Implementación del visualizador SPARQL Endpoint QR.....	80
Figura 41. Burndown Chart - Sprint 7.....	82
Figura 42. Página index.html.....	83
Figura 43. Página qrcode.html.....	83
Figura 44. Página contactos.html.....	84
Figura 45. Fragmento de la función QRCode en JavaScript.....	84
Figura 46. Fragmento del archivo estilos.ccs.....	85
Figura 47. Implementación del generador de códigos QR en el archivo qrcode.html.....	85
Figura 48. Resultados de la prueba de aceptación 1.....	91
Figura 49. Resultados de la prueba de aceptación 2.....	94
Figura 50. Resultados de la prueba de aceptación 3.....	97
Figura 51. Resultados de la prueba de aceptación 4.....	100
Figura 52. Test de rendimiento - Nexus 5 (Android 5.0).....	101
Figura 53. Test de rendimiento - Galaxy S7 Edge (Android 6.0).....	102
Figura 54. Test de rendimiento - Galaxy S7 (Android 7.0).....	102
Figura 55. Resultados de carga del sitio web.....	103
Figura 56. Resultados de la carga de objetos del sitio web.....	104
Figura 57. Mapa de actividad y navegación de la aplicación móvil.....	105
Figura 58. Lista de dispositivos Android usados para prueba.....	106
Figura 59. Resultados de optimización del sitio web.....	107
Figura 60. Implementación de la aplicación móvil en Google Play.....	108
Figura 61. Implementación del sitio web en hosting gratuito.....	109

ÍNDICE DE TABLAS

Tabla 1. Comparación entre Web 2.0 y Web 3.0	23
Tabla 2. Tripletas RDF correspondiente al Usuario	26
Tabla 3. Consulta SPARQL-SELECT	32
Tabla 4. Consulta SPARQL-CONSTRUCT	32
Tabla 5. Consulta SPARQL-ASK	33
Tabla 6. Consulta SPARQL-DESCRIBE	33
Tabla 7. Datasets DBpedia	35
Tabla 8. Clases comunes de DBpedia	36
Tabla 9. Trabajos Relacionados	49
Tabla 10. Product Backlog	58
Tabla 11. Backlog	58
Tabla 12. Planificación de Sprint	60
Tabla 13. Avance diario del Sprint 1	61
Tabla 14. Avance diario del Sprint 2	64
Tabla 15. Avance diario del Sprint 3	68
Tabla 16. Avance diario del Sprint 4	71
Tabla 17. Avance diario del Sprint 5	74
Tabla 18. Avance diario del Sprint 6	77
Tabla 19. Avance diario del Sprint 7	81
Tabla 20. Actividades de Sprint Review	87
Tabla 21. Actividades de Sprint Retrospective	88
Tabla 22. Prueba de aceptación 1	89
Tabla 23. Prueba de aceptación 2	92
Tabla 24. Prueba de aceptación 3	95
Tabla 25. Prueba de aceptación 4	98
Tabla 26. Resultados de usabilidad - Aplicación móvil	106
Tabla 27. Lista de navegadores usados para prueba	108
Tabla 28. Planteamiento del problema	120
Tabla 29. Resumen de Stakeholders	120
Tabla 30. Perfil de los stakeholders: Director del Proyecto	121
Tabla 31. Perfil de los stakeholders: Administrador del Proyecto	121
Tabla 32. Perfil de los stakeholders: Usuarios técnicos y usuarios finales	122
Tabla 33. Resumen de características del producto	123
Tabla 34. Herramientas para ejecución del plan de pruebas	127

ÍNDICE DE ANEXOS

Anexo 1. Documento de Visión.....	117
Anexo 2. Plan de Pruebas	124

RESUMEN

Este trabajo se basa en el estudio realizado en el campo de la web semántica, sus tecnologías complementarias y especialmente la DBpedia, cuya base investigativa facilitó la construcción de una aplicación para dispositivos móviles de Android que permite el consumo (consulta) de información al repositorio semántico de la DBpedia y la visualización de los datos obtenidos.

La aplicación móvil desarrollada permite al usuario realizar consultas a la DBpedia utilizando diferentes mecanismos tales como: palabras claves y consultas SPARQL; consultas que pueden ser digitadas por el usuario o ejecutadas directamente a través de consultas SPARQL codificadas en códigos QR. La codificación de las consultas SPARQL en códigos QR se realiza mediante una aplicación web construida para dicha finalidad.

Al desarrollar una aplicación de este tipo, lo que se busca es brindar al usuario una herramienta que le permita tener acceso de manera sencilla a todo el contenido de conocimientos que ofrece la DBpedia, haciendo uso de un dispositivo móvil de Android.

PALABRAS CLAVES: Web Semántica, Web 3.0, RDF, SPARQL Endpoint, DBpedia extracción, DBpedia móvil, Android.

ABSTRACT

This work is based on the study done in the field of semantic web, its complementary technologies and especially the DBpedia, whose research base facilitated the construction of an application for Android mobile devices that allows the consumption (query) of information to the semantic repository of the DBpedia and the visualization of the obtained data.

The developed mobile application allows the user to query the DBpedia using different mechanisms such as: SPARQL keywords and queries; queries that can be entered by the user or executed directly through SPARQL queries encoded in QR codes. The coding of SPARQL queries in QR codes is done through a web application built for this purpose.

In developing an application of this type, what is sought is to provide the user with a tool that allows them to have easy access to all the knowledge content offered by DBpedia, making use of an Android mobile device.

KEYWORDS: Semantic Web, Web 3.0, RDF, SPARQL Endpoint, DBpedia extraction, DBpedia mobile, Android.

INTRODUCCIÓN

Actualmente la Web tradicional está compuesta por cantidades gigantescas de información de todo tipo y en diferentes formatos; se encuentra almacenada en plataformas muy variadas y ésta no se encuentra organizada o codificada para que las máquinas o dispositivos puedan usarla de forma fácil y directa.

Sin embargo, el avance de las tecnologías web ha permitido que estas grandes cantidades de información puedan ser organizadas utilizando esquemas de representación que las máquinas logran interpretar y entender. De ahí que surge la Web Semántica, la misma que tiene precisamente como objetivo organizar y representar la información para que las máquinas la puedan procesar y entender. La Web Semántica codifica los datos de tal forma que las búsquedas de información son más precisas y exactas, intentando de esa manera mejorar el retorno de resultados de una búsqueda en relación a los buscadores actuales, como por ejemplo Google. Pero aunque son muchas las ventajas que proporciona la Web Semántica, aún son pocos los mecanismos que existen para el consumo y visualización de datos, específicamente, sobre repositorios como DBpedia y en un ambiente móvil.

El presente trabajo tiene como finalidad, desarrollar una aplicación para dispositivos móviles de Android la cual se la denominada SevirQR (Search & Visualization RDF with QR), que mediante el uso de palabras claves y consultas SPARQL permita consumir y visualizar información de la DBpedia. Se implementaron los mecanismos y servicios más adecuados que permitieron conectar esta aplicación móvil con los recursos que ofrece la DBpedia para poder extraer y visualizar su información. Adicionalmente, se implementó un sitio web que proporciona información referente a este trabajo, y de manera especial, ofrece un generador de códigos QR el cual permite codificar consultas SPARQL y cuyos códigos serán leídos por la aplicación móvil para ejecutar consultas SPARQL automáticas a la DBpedia. Esta aplicación permite a los usuarios realizar consultas en los repositorios de la DBpedia de una forma sencilla utilizando su propio dispositivo móvil. Por este motivo, cabe destacar la importancia que tiene el desarrollo de este trabajo, pues con la construcción de esta aplicación móvil se estará generando una solución novedosa en un ambiente móvil y que implica el consumo y visualización de información sobre un repositorio semántico.

A continuación se describen los capítulos que componen este trabajo: el capítulo 1 detalla toda la base teórica e investigativa donde se abordan los principales temas como son la web semántica, Linked data, DBpedia, tecnología móvil y el análisis de varios trabajos relacionados; el capítulo 2 describe la propuesta de la solución del trabajo donde se realiza un resumen de los trabajos relacionados, se especifican los objetivos que se desean alcanzar y los componentes a desarrollar, se indica la arquitectura a utilizar y finalmente la metodología

de desarrollo que se va a utilizar; el capítulo 3 detalla el proceso de desarrollo de la aplicación móvil y el sitio web según la metodología SCRUM. Aquí es importante indicar que se adoptó esta metodología de desarrollo, ya que la misma permite un desarrollo flexible permitiendo incorporar nuevos requerimientos y además se adapta fácilmente cuando el equipo de desarrollo es pequeño y el tiempo de desarrollo del proyecto es relativamente corto. El capítulo 4 documenta las pruebas de funcionamiento realizadas y la implementación de la aplicación móvil y el sitio web. Finalmente, se describen las respectivas conclusiones y recomendaciones.

En el desarrollo de este trabajo se encontraron ciertos inconvenientes, principalmente con lo relacionado a los mecanismos de conexión con la DBpedia los cuales permiten realizar el consumo de la información, y que además involucra la lectura de los archivos de respuesta serializados en formato RDF y JSON que contienen la información producto de las consultas SPARQL realizadas. Otra de las dificultades presentadas, fue también la manera en cómo se debían construir los visualizadores para lograr mostrar los resultados obtenidos en cada consulta SPARQL y que estos se acoplen al dispositivo móvil. Sin embargo, gran parte de estos inconvenientes fueron solucionados gracias a la investigación realizada en este trabajo y al análisis efectuado a varios trabajos relacionados.

**CAPITULO I:
Estado del Arte**

1.1. Introducción

Este capítulo presenta los principales conceptos y definiciones, tales como: web semántica, Linked data, DBpedia, tecnología móvil, los mismos que aportan la base investigativa para poder sustentar y desarrollar este trabajo. Finalmente, se hace un estudio de varios trabajos relacionados al consumo y la visualización de datos usando la DBpedia, lo que permitirá tener una visión más amplia y enfocada para elaborar una propuesta de solución.

1.2. La web y su evolución

La World Wide Web desde sus inicios en 1989 con Tim Berners-Lee,¹ ha tenido que ir evolucionando para permitir que las personas puedan desarrollar tecnologías digitales y medios efectivos de comunicación en tiempo real. Pero por sobre todas estas innovaciones, el descubrimiento de la información y del conocimiento, es un tema donde aún la web sigue en constante evolución hasta el día de hoy.

1.2.1. Creación de la web

La idea de World Wide Web o conocida simplemente como “la web”, fue introducida por primera en 1989 por Tim Berners-Lee, pero fue necesario esperar hasta la aparición del primer navegador (Mosaic, 1993), para que ésta fuese de fácil acceso (Zapater, J., & Calderaro., 2013). Al hablar de la web también se hace referencia a un conjunto de documentos que se encuentran disponibles en Internet, pues es necesario entender que la web e Internet no se refieren al mismo asunto. La web es un servicio más de tantos que dispone y ofrece Internet. (E-mail, FTP, voz sobre IP, etc.)

1.2.2. Etapas evolutivas de la web

La web ha dado grandes pasos en su desarrollo y evolución en las últimas décadas. Esta evolución ha traído consigo nuevas funcionalidades y mejoras aplicadas para el beneficio de los usuarios y para una mejor utilización de los datos y recursos electrónicos.

¹ Científico británico considerado como el padre de la web. Disponible en: https://es.wikipedia.org/wiki/Tim_Berners-Lee



Figura 1. Evolución de la web.

Fuente: ("Historia de la Web 2.0 - Historia de la Informática", 2010)

La Figura 1, muestra las diferentes fases por las cuales ha transitado la web desde su origen. Esta evolución consta de cuatro generaciones y cada una de ellas apunta a un objetivo específico. A continuación, se caracteriza lo más sobresaliente de cada una de estas etapas, poniendo especial interés en lo que se refiere al proceso de búsqueda de información (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012).

1.2.2.1. WEB 1.0 – Centrada en la información

La Web 1.0 está enfocada principalmente en los datos y en la manera en que éstos pueden ser de utilidad a las personas. A continuación, se presentan sus principales características (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012):

- Se inicia en 1989 con la propuesta de Tim Berners-Lee.
- Considerada como la “web sólo de lectura”, pues el contenido que presenta está basado en un HTML estático y casi no necesita de la interactividad del usuario.
- La actualización de la información no se da con frecuencia.
- Existe muy poca interacción entre los sitios web.
- Sus protocolos esenciales son: URI, HTTP, HTML.
- Aparece el primer motor de búsqueda (Worm, 1993) que maneja los índices de títulos y encabezados. Más adelante, aparecen nuevos motores que realizan la búsqueda sobre el contenido de sitios web y usando el lenguaje natural juntamente con operadores booleanos. (WebCrawler, 1994) y (AltaVista, 1995) respectivamente.
- Los motores de búsqueda de esta generación ignoran gran parte de la información relevante o simplemente la confunden, por lo que los resultados de búsqueda que se obtienen presentan muy poca precisión.

1.2.2.2. WEB 2.0 – Centrada en las personas

Lo que busca la Web 2.0 es que las personas puedan tener la posibilidad de interactuar con los datos, algo que en la Web 1.0 prácticamente era imposible. Sus características más importantes son (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012):

- La Web 2.0 se define oficialmente en 2004 por Dale Dougherty, cuyo objetivo busca que esta web permita la creación de aplicaciones interactivas, y aprovechando el impacto que ya poseen las redes, que más personas puedan aprovechar estas aplicaciones de mejor forma.
- Es considerada como la “web de lectura – escritura”, “web de la sabiduría” o “web participativa”, pues es una web dirigida a los usuarios y donde estos ahora sí tiene la posibilidad y capacidad de interactuar.
- Apoya la colaboración de los usuarios mediante las llamadas “redes sociales”, y genera un nuevo concepto denominado “inteligencia colectiva” (Inteligencia que se origina de la participación y colaboración que realizan muchos individuos de su misma especie).
- Las principales tecnologías y servicios de esta web son: blogs, wikis, servicios web, etc.
- Aparece el motor de búsqueda más famoso y utilizado en la historia (Google).
- A pesar de que los mecanismos para realizar la búsqueda de información por la web se mejoran considerablemente, el gran inconveniente que ahora se presenta es la exuberante cantidad de información existente, y por tanto, es mucha también la cantidad de resultados que se obtienen para un determinado tema, dificultando así el análisis del mismo. En la mayoría de los casos, la búsqueda trae consigo resultados incongruentes o que no tienen relación alguna.

1.2.2.3. WEB 3.0 – Centrada en las máquinas

La Web 3.0 da prioridad a la información estructurada, característica que permitirá al usuario realizar búsquedas más rápidas y eficaces. Lo más destacable de esta etapa se describe a continuación (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012):

- Es más conocida como la “Web Semántica” donde una vez más Tim Berners-Lee es quien está al frente de este concepto.
- La idea básica de este modelo es poder definir una estructura de datos que los vincule, y así poder descubrir, automatizar, integrar y reusar los mismos a través de varias aplicaciones.

- La Web 3.0 trata de vincular, integrar y analizar la información de varias fuentes de datos, sin importar su origen o formato, con la finalidad de obtener una nueva fuente de información.
- El principal y más importante objetivo de la web semántica es hacer que el contenido de los sitios web sea legible para las máquinas tanto como lo es para los seres humanos.
- Aunque los mecanismos de búsqueda de información requieran de un conocimiento más profundo (búsquedas semánticas) y sobre conjunto de datos semánticos más especializados (Datasets), el resultado que se obtiene para un determinado tema es de mucho más valor y exactitud.

1.2.2.4. WEB 4.0 – Centrada en los agentes

Podríamos decir que aún no hemos ni siquiera entrado en la era de la Web 4.0, ya que esta web estaría enfocada en una interacción mucho más compleja y sofisticada entre el usuario y los agentes software. Sus principales características son (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012):

- La Web 4.0 aún está en gestación y a ciencia cierta no se la puede definir con exactitud. Sin embargo, se podría decir que es un novedoso modelo de interacción con el usuario, mucho más completo y personalizado.
- Es llamada también “web simbiótica”, y el sueño que ésta persigue es que en algún momento sea posible que la mente humana y el de las máquinas puedan interactuar en “simbiosis”.
- Lo más interesante de este modelo de Web es que sería posible hacer consultas y peticiones en lenguaje natural, por ejemplo: “Necesito el servicio de taxi, urgente” y entonces el teléfono móvil se comunicaría automáticamente con la compañía de taxis más cercana, sin intervención directa del usuario.

1.2.3. Diferencias principales entre la Web 2.0 y la Web 3.0

Las diferencias generales entre estas dos generaciones de web son:

- La Web 2.0 da mayor importancia a interacción de las personas, mientras que la Web 3.0 prioriza la estructuración de la información.
- La Web 2.0 apunta a las comunicaciones y redes sociales, mientras que la Web 3.0 trabaja sobre conjuntos de datos vinculados.
- Finalmente, en la Web 2.0 el usuario tiene cierto poder y control sobre la información, en tanto que en la Web 3.0, son las máquinas las que tienen el poder de generar e interpretar la información automáticamente.

La Tabla 1, muestra aspectos específicos que hacen significativas las diferencias existentes entre la web 2.0 y la web 3.0. Por ejemplo, la web 2.0 es una web de lectura y escritura, dirigida a crear comunidades de usuarios y que permite compartir contenido; los blogs son la herramienta más común de esta web junto con la tecnología *AJAX*, la cual permite el desarrollo de aplicaciones interactivas en la web 2.0. Finalmente, el *tagging* no es más que un sistema de etiquetas que poseen las aplicaciones de la web 2.0 para agrupar artículos o información que tiene relación entre sí. De igual forma, de la web 3.0 se dice que es una web personal y portable, pues está dirigida a proporcionar la información que requiere cada individuo. La web 3.0 busca consolidar la mayor cantidad de información en un mismo formato que sea entendible y de fácil proceso para las máquinas a través del uso de tecnologías como RDF, que es un modelo de datos que permite describir y estructurar recursos semánticos. En la sección 1.2.4 se hace una amplificación de este tema. Por último, la web 3.0 busca que la información pueda estar enlazada, y que al usar esos vínculos, sea posible explorar el conocimiento de la web. Un ejemplo de ello es la DBpedia, tema que se abordará en la sección 1.2.8.

Tabla 1. Comparación entre Web 2.0 y Web 3.0

WEB 2.0	WEB 3.0
Read/Write Web	Portable Personal Web
Communities	Individuals
Sharing Content	Consolidating Dynamic Content
Blogs	Lifestream
AJAX	RDF
Wikipedia, google	DBpedia, igoogole
Tagging	User engagement

Fuente: (Aghaei, S., Nematbakhsh, M., & Farsani, H., 2012)

1.3. La Web 3.0 o Web Semántica

Se mencionó que la Web Semántica trata de mejorar la estructura y definición de la información, para de esa forma, dotarla de semántica lo que permite darle un mayor significado. Es por eso que se dice que la Web Semántica es una web con significado, puesto que permitirá a cualquier usuario realizar consultas de una manera más rápida y obtener resultados exactos.

1.3.1. Introducción a la Web Semántica

La idea fundamental de la web semántica es permitir la interrelación entre un usuario y un agente de software (Codina, L., & Rovira, C., 2006), donde el usuario sea quién dictamine las reglas de los resultados que desea obtener en una búsqueda de datos, y entonces, el agente de software tenga la capacidad de diseñar un plan de búsqueda que le permita responder efectivamente y con rapidez a las demandas del usuario.

La web semántica no es una web separada o independiente de la web actual, sino más bien es una web que permite dar a la información un significado bien definido; esto permite, como se mencionó anteriormente, crear una cooperación usuario/máquina (Berners-Lee, T., Hendler, J., Lassila, O., & others., 2001). Aún hoy, la web tradicional es un medio de documentos para las personas, donde sólo estas tienen la capacidad de entender el contenido de dichos documentos. Sin embargo, la web semántica pretende compensar esto haciendo que las máquinas y agentes software también tengan la capacidad de comprender a la web tradicional y así permitir que su información pueda ser procesada automáticamente.

En resumen, la web semántica intenta solventar las carencias que tiene la web actual, transformando la información común en “fuentes o bases de conocimiento”.

1.3.2. Componentes de la Web Semántica

La web semántica es presentada por Tim Berners-Lee, quien es considerado como el padre de la web, como una pirámide de siete capas denominada “Web Semántica Pila” o también conocida como “Web Semántica Layer Cake”, la misma que a su vez, consta de nueve componentes que describen tres tipos de información: datos, metadatos y clases de datos (Castro, V., 2014).

La Figura 2, permite apreciar la estructura y los componentes de la web semántica.

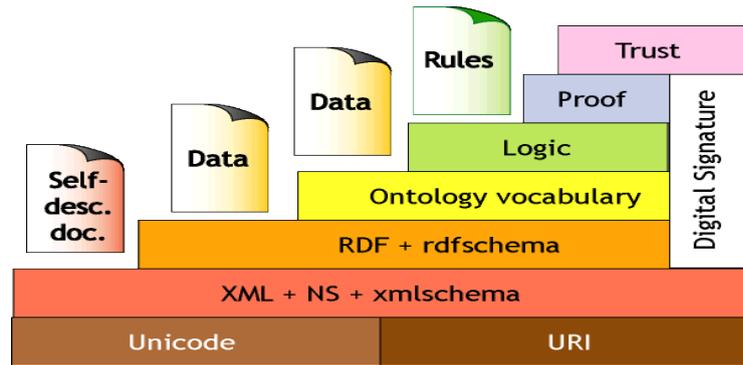


Figura 2. Arquitectura de la web semántica

Fuente: ("Semantic Web - XML2000 - Slide "Architecture"", 2000)

Haciendo una breve descripción de su arquitectura, y comenzando desde las capas inferiores se encuentran tecnologías conocidas y consolidadas, las mismas que representan la base de la web semántica. El estándar URI (*Uniform Resource Identifier*), permite identificar los recursos y representar los caracteres Unicode (estándar que proporciona un número único para cada carácter). En el siguiente nivel se encuentra XML (*eXtensible Markup Language*), el cual define tipos de datos, componentes y restricciones, e intenta dar un poco de información sobre la semántica de los objetos.

Sin embargo, XML no es una tecnología que permite definir completamente la estructura y el intercambio de la información en todos los ámbitos. Por este motivo, en el siguiente nivel de la arquitectura de la web semántica aparece la tecnología RDF (*Resource Description Framework*), que es un modelo lógico que permite representar las relaciones entre los recursos. RDF es el elemento clave de toda la arquitectura. Más adelante, se analizará su estructura y las funciones que desempeña.

Subiendo al nivel siguiente, encontramos a *Ontology Vocabulary*, la misma que expresa el significado que se desea dar a un determinado conjunto de términos para cierto dominio del conocimiento. Un ejemplo de dominio del conocimiento sería, "la web semántica". En la siguiente capa encontramos a *Logic* o nivel lógico, encargada de analizar las reglas formales para interpretar si un razonamiento es válido o no. Más adelante, se encuentra *Proof* o nivel de prueba, el cual establece demostraciones con las inferencias generados por los razonamientos válidos. Finalmente, la capa *Trust + Digital Signature* o nivel de confianza, determina la confianza para garantizar la fiabilidad y certificar la procedencia de las ontologías y datos.

1.3.3. Linked Open Data y la Web Semántica: los principios de Linked Data

La W3C (*World Wide Web Consortium*) es la encargada de promover las tecnologías y mejoras de la web, y en el caso de "*Linked Open Data*" no es la excepción. Linked data permite

la compartición de datos estructurados en la web, bajo el principio de que el valor y la utilidad de los datos se incrementan mientras mayor sea el número de enlaces que exista entre ellos. De esa manera, Linked data hace uso de la web para crear enlaces entre datos de fuentes diferentes (Lupiani, E., et al., 2010).

En 2009, Tim Berners-Lee redefinió los principios de *Linked Open Data* mediante las siguientes tres reglas (Hallo, M., et al., 2012):

- Se asignará a todo objeto conceptual nombres que comiencen con http.
- Se obtendrá información importante a partir de estos nombres.
- La información obtenida deberá contener relaciones a otros datos.

Así, Linked data utiliza modelos RDF (*Resource Description Framework*) para representar los datos estructurados de la web, a la vez que permite vincular otros datos para poder enriquecer y mejorar las formas de exploración de los datos para las personas y las máquinas. Esto ha contribuido, de gran manera, el proceso de publicación de datos en la Web Semántica brindando recursos abiertos enlazados (Piedra, N., Chicaiza, J., Cadme, E., & Guaya, R., 2016).

Una vez que los datos semánticos han sido publicados, entonces, la gran interrogante es: ¿cómo poder recuperar y consultar esta información? El problema principal radica en que el usuario debería manejar los siguientes tres elementos: 1) la sintaxis de los modelos RDF, 2) un lenguaje formal que permita realizar consultas semánticas (SPARQL), 3) la estructura del repositorio destino donde se aplicarán las consultas (DBpedia).

1.3.4. RDF (Resource Description Framework)

RDF consiste en un “marco de descripción de recursos”, cuyo principal objetivo es brindar un modelo de datos que proporcione una organización lógica y definida de los datos. El modelo de datos de RDF es bastante simple, pero muy expresivo. El mismo está conformado por su unidad básica de información denominada “tripleta”, la misma que está compuesta de: un sujeto (s), un predicado (p) y un objeto (o) (Curé, O., & Blin, G., 2015).

Tabla 2. Tripletas RDF correspondiente al Usuario

Subject	Predicate	Object
JD	firstName	Joe
JD	lastName	Doe
JD	Gender	Male
MS	firstName	Mary
MS	Gender	Female

Fuente: (Curé, O., & Blin, G., 2015)

La Tabla 2, permite observar un ejemplo bastante claro de tripletas RDF. Cada tripleta representa un hecho sobre algo que se está intentando describir. Al sujeto también se lo conoce como “recurso”, al predicado como una “propiedad” y al objeto como un “valor”.

RDF es una manera de representar datos y metadatos. Su modelo de datos se basa en un grafo dirigido cuyos nodos son el sujeto y el objeto, y sus arcos corresponden a sus propiedades – “Predicado” (Almendros-Jiménez, J., 2008).

Lo que se busca con el esquema RDF es la representación de los datos obtenidos de diversas fuentes en un solo conjunto de estas tripletas. Además, se resuelve el problema de la identificación de recursos comunes entre las múltiples fuentes, puesto que RDF hace uso de URIs para enlazar a los datos. Una URI representa a un identificador global común de cualquier recurso en la web, (por ejemplo, “http://recursos.com/blog#user”).

La Figura 3, muestra el grafo RDF completo de un grupo semántico de tripletas.

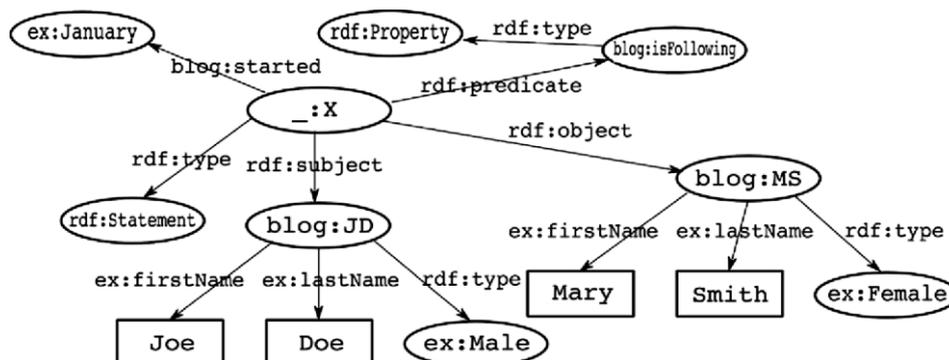


Figura 3. Grafo RDF
Fuente: (Curé, O., & Blin, G., 2015)

1.3.5. Serializaciones RDF

A pesar de que RDF permite la representación de datos de manera simple, este modelo en sí mismo no indica exactamente cómo se deben representar las tripletas en un formato entendible para la máquina. Es ahí donde aparece la serialización, la misma que permite describir una representación textual de los datos o de la información. Existen serializaciones como RDF/XML, N-triples, N3, JSON, Turtle, las mismas que definen una sintaxis especial para declaraciones RDF (Powell, J., & Hopkins, M., 2015).

1.3.5.1. RDF/XML

Es una recomendación de la W3C que permite expresar tripletas RDF utilizando el formato XML. Se dice que es el formato estándar de intercambio de datos de la web semántica. Sin embargo, es un formato muy complejo de leer. La Figura 4 es un ejemplo de este formato (*Resource Description Framework (RDF) Model and Syntax Specification, 1999*):

```

<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:s="http://description.org/schema">
  <rdf:Description about="http://www.w3.org/Home/Lassila">
    <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>

```

Figura 4. Serialización RDF/XML

Fuente: (Resource Description Framework (RDF) Model and Syntax Specification, 1999)

1.3.5.2. N-triples

Es la serialización más simple para la representación textual de datos RDF, pero es quizá la forma más difícil de representación visual. Esto, debido a que este esquema no permite la abreviación de URIs. A continuación, un ejemplo de N-triples (Curé, O., & Blin, G., 2015):

```

<http://example.com/Cat#Science      http://www.w3.org/1999/02/22-rdf-
syntax-ns#type http://example.com/Cat#Category> .
<http://example.com/Cat#Science      http://example.com/Cat#title "Science"> .

```

Figura 5. Serialización N-triples

Fuente: (Curé, O., & Blin, G., 2015)

1.3.5.3. N3 (Notation 3)

Este esquema fue propuesto por Tim Berners-Lee como una forma de juntar la simplicidad de la N-triples y la expresividad de los RDF/XML. Su representación es muy semejante a la N-triples, sin embargo, es posible abreviar las URIs mediante nombres de prefijos (Curé, O., & Blin, G., 2015). A continuación, se presenta un ejemplo:

```

@prefix blog: <http://example.com/Blog#> .
@prefix cat: <http://example.com/Cat#> .
@prefix ex: <http://example.com/terms#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
<cat:Sport> a cat:Category ;
    cat:title "Sport" .
<cat:Science> a cat:Category ;
    cat:title "Science" .
<cat:Tech> a cat:Category ;
    cat:title "Technology" .
<http://blogs.com/blog1> a blog:Blog ;
    blog:category <cat:Tech> ;
    blog:content "Today..." ;
    blog:owner <http://blogs.com/JD> ;
    ex:writtenOn "10/13/2013"^^<xsd:date> .

```

Figura 6. Serialización N3

Fuente: (Curé, O., & Blin, G., 2015)

1.3.5.4. RDF/JSON (Javascript Object Notation)

Es un estándar que facilita a las máquinas el intercambio de datos semánticos. A comparación con RDF/XML, es un esquema fácil de entender. JSON define un formato de par “clave/valor” muy flexible que permite representar y agrupar varias estructuras de datos (Powell, J., & Hopkins, M., 2015). En seguida, se muestra un ejemplo de este esquema (*RDF 1.1 JSON Alternate Serialization (RDF/JSON)*, 2013):

```

{
  "http://example.org/about" : {
    "http://purl.org/dc/terms/title" : [ { "value" : "Anna's Homepage",
                                          "type" : "literal",
                                          "lang" : "en" } ]
  }
}

```

Figura 7. Serialización RDF/JSON

Fuente: (*RDF 1.1 JSON Alternate Serialization (RDF/JSON)*, 2013)

1.3.5.5. Turtle (Terse RDF Triple Language)

Permite, de una forma concisa y fácil, leer las tripletas RDF (Powell, J., & Hopkins, M., 2015). Turtle es una versión simplificada de N3 que se vuelve cada vez más popular y que es recomendada por W3C. A continuación, se muestra un ejemplo de esta estructura (Curé, O., & Blin, G., 2015):

```

@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix cat:<http://example.com/Cat#> .
@prefix blog:<http://example.com/Blog#> .
@prefix ex:<http://example.com/terms#> .
...
blog:MS      a          blog:User ;
              ex:firstName  "Mary" ;
              ex:lastName   "Smith" ;
              blog:hasGender ex:Female ;
              blog:isFollowing (blog:JD _:blogger4) .
_:blogger4   a          blog:User .
...

```

Figura 8. Serialización Turtle

Fuente: (Curé, O., & Blin, G., 2015)

1.3.5.6. Otras serializaciones

RDFa define una gran cantidad de atributos HTML que se utilizan para codificar datos RDF en varios contextos, especialmente, páginas web. El siguiente, es un ejemplo de este esquema (Powell, J., & Hopkins, M., 2015):

```

<dl itemscope itemid="http://dbpedia.org/resource/Kona_District,_Hawaii">
<dt>Subject Item</dt><dd>n2:_Hawaii</dd>
<dt>rdf:type</dt><dd>
<a itemprop="http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
href="http://dbpedia.org/ontology/Place">dbpedia-owl:Place</a>
<a itemprop="http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
href="http://schema.org/Place">n9:Place</a>
<a itemprop="http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
href="http://dbpedia.org/ontology/PopulatedPlace">dbpedia-owl:PopulatedPlace</a>
<a itemprop="http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
href="http://www.ontologydesignpatterns.org/ont/d0.owl#Location">d0:Location</a>
<a itemprop="http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
href="http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing">geo:SpatialThing</a>
</dd>
<dt>dcterms:subject</dt><dd>
<a itemprop="http://purl.org/dc/terms/subject" href="http://dbpedia.org/resource/Category:
Geography_of_Hawaii_(island)">n28:</a>
</dd>
<dt>grs:point</dt><dd>
<span itemprop="http://www.georss.org/georss/point">19.6996 -155.99</span>
</dd>

```

Figura 9. Serialización RDFa

Fuente: (Powell, J., & Hopkins, M., 2015)

Por tanto, la serialización permite tener varios formatos específicos para representar y recuperar información, lo cual no altera el significado original de las declaraciones RDF. Además, la serialización permite que la representación RDF tenga un formato más

comprensible; que su contenido pueda ser incorporado en páginas web o que se pueda realizar un mejor intercambio de datos RDF (Powell, J., & Hopkins, M., 2015).

1.3.6. SPARQL (SPARQL Protocol and RDF Query Language)

RDF es un modelo de datos de la web semántica, y como todo modelo de datos, éste viene incorporado con un lenguaje que soporta consultas sobre conjuntos de datos. Este lenguaje de consultas se denomina SPARQL, el cual proporciona especificaciones y protocolos para realizar consultas y manipular datos RDF. La sintaxis utilizada por SPARQL fue adoptada de SQL, (Structured Query Lenguaje), que es un lenguaje de consultas de base de datos muy popular y ampliamente aceptado (Curé, O., & Blin, G., 2015).

Es así, que Tim Berners Lee mencionó que “tratar de usar la Web Semántica sin SPARQL sería como intentar usar una base de datos relacional sin SQL”. Las búsquedas se han convertido en un proceso crucial al momento de manejar datos e información, puesto que los mayores avances tecnológicos de la información se han logrado gracias a la “búsqueda” (Powell, J., & Hopkins, M., 2015).

SPARQL permite realizar consultas sobre diversas fuentes de datos RDF. El resultado que se obtiene de dichas consultas SPARQL pueden ser conjuntos de resultados o grafos RDF (Consortium, W.W.W., & others., 2008).

Sin embargo, el proceso de ejecución de la consulta es muy diferente a SQL. Con SPARQL es necesario que las variables estén soportadas en los patrones de tripletas RDF, de forma particular, en la cláusula WHERE. Una variable se identifica mediante el signo de interrogación (?) seguido del nombre de la misma. Estas variables permiten almacenar la información resultante del problema de consulta. A continuación, se muestra una consulta simple utilizando SPARQL (Curé, O., & Blin, G., 2015):

```
PREFIX blog: <http://example.com/Blog#>  
PREFIX ex: <http://example.com/terms#>  
SELECT ?name  
WHERE {  
    ?user      ex:lastName    "Doe" ;  
              ex:firstName    ?name ;  
}
```

Figura 10. Consulta SPARQL

Fuente: (Curé, O., & Blin, G., 2015)

Es importante mencionar que SPARQL define un lenguaje de búsqueda así como una forma de comunicación con las tripletas RDF. Puesto que SPARQL está construido sobre el protocolo HTTP, una consulta de SPARQL puede ser transportada a través de la web (World

Wide Web). Así mismo, SPARQL requiere que los datos resultantes de la búsqueda se obtengan mediante un documento XML (Powell, J., & Hopkins, M., 2015).

1.3.7. Formas de Consultas SPARQL

Existen cuatro formas de realizar consultas mediante SPARQL (Consortium, W.W.W., & others., 2008). Estas formas de consulta son:

1.3.7.1. SELECT

Obtiene todo, o un subconjunto de las variables vinculadas con la concordancia del patrón de búsqueda. La Tabla 3, muestra un ejemplo de consulta SELECT:

Tabla 3. Consulta SPARQL-SELECT

Consulta SELECT	Resultado									
<pre> PREFIX foaf: <http://xmlns.com/foaf/0.1/> SELECT ?nameX ?nameY ?nickY WHERE { ?x foaf:knows ?y ; foaf:name ?nameX . ?y foaf:name ?nameY . OPTIONAL { ?y foaf:nick ?nickY } } </pre>	<table border="1"> <thead> <tr> <th>nameX</th> <th>nameY</th> <th>nickY</th> </tr> </thead> <tbody> <tr> <td>"Alice"</td> <td>"Bob"</td> <td></td> </tr> <tr> <td>"Alice"</td> <td>"Clare"</td> <td>"CT"</td> </tr> </tbody> </table>	nameX	nameY	nickY	"Alice"	"Bob"		"Alice"	"Clare"	"CT"
nameX	nameY	nickY								
"Alice"	"Bob"									
"Alice"	"Clare"	"CT"								

Fuente: (Consortium, W.W.W., & others., 2008)

1.3.7.2. CONSTRUCT

Obtiene un grafo RDF que está construido por la sustitución de variables en un conjunto de plantilla de tripleta. Un ejemplo de consulta CONSTRUCT se puede observar en la siguiente tabla:

Tabla 4. Consulta SPARQL-CONSTRUCT

Consulta CONSTRUCT	Resultado
<pre> PREFIX foaf: <http://xmlns.com/foaf/0.1/> PREFIX vcard: <http://www.w3.org/2001/vcard-rdf/3.0#> CONSTRUCT { <http://example.org/person#Alice> vcard:FN ?name } WHERE { ?x foaf:name ?name } </pre>	<pre> @prefix vcard: <http://www.w3.org/2001/vcard-rdf/3.0#> . <http://example.org/person#Alice> vcard:FN "Alice" . </pre>

Fuente: (Consortium, W.W.W., & others., 2008)

1.3.7.3. ASK

Obtiene un valor booleano que indica si se ha encontrado o no alguna concordancia con el patrón de consulta. Es una forma utilizada por las aplicaciones para comprobar que un patrón de consulta tiene solución. La Tabla 5, muestra un ejemplo de consulta ASK:

Tabla 5. Consulta SPARQL-ASK

Consulta ASK	Resultado
PREFIX foaf: <http://xmlns.com/foaf/0.1/> ASK { ?x foaf:name "Alice" }	yes

Fuente: (Consortium, W.W.W., & others., 2008)

1.3.7.4. DESCRIBE

Obtiene un grafo RDF que describe a cada recurso encontrado en el patrón de consulta. La descripción de los recursos está determinada por el servicio de consulta. Esta es una forma de consulta informativa. A continuación se muestra un ejemplo de consulta DESCRIBE:

Tabla 6. Consulta SPARQL-DESCRIBE

Consulta DESCRIBE
PREFIX ent: <http://org.example.com/employees#> DESCRIBE ?x WHERE { ?x ent:employeeid "1234" }
Resultado
@prefix foaf: <http://xmlns.com/foaf/0.1/> . @prefix vcard: <http://www.w3.org/2001/vcard-rdf/3.0> . @prefix exOrg: <http://org.example.com/employees#> . @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> . @prefix owl: <http://www.w3.org/2002/07/owl#> _:a exOrg:employeeid "1234" ; foaf:mbox_sha1sum "ABCD1234" ; vcard:N [vcard:Family "Smith" ; vcard:Given "John"] . foaf:mbox_sha1sum rdf:type owl:InverseFunctionalProperty .

Fuente: (Consortium, W.W.W., & others., 2008)

Adicionalmente, SPARQL posee otros tipos de cláusulas que permiten mayor detalle o complejidad para implementar consultas en repositorios RDF. Por ejemplo, tenemos las cláusulas: DISTINCT, REDUCED, LIMIT, FILTER, OPTIONAL, ORDER BY.

1.3.8. DBpedia

DBpedia es un proyecto que se fundamenta en la extracción de información estructurada de Wikipedia, lo que permite que ésta información sea accesible en la web. Actualmente, DBpedia posee aproximadamente 4,7 billones de piezas de información que abarca muchos dominios del conocimiento (Bizer, C., et al., 2009).

Las contribuciones que realiza el proyecto DBpedia al desarrollo de la Web de Datos son:

- Desarrolla un marco (framework) de extracción de información que permite convertir el contenido de Wikipedia en una rica base de conocimientos multi-dominio.
- Define un identificador Web para cada entidad o recurso de DBpedia, permitiendo así, la interconexión de las fuentes de datos en la Web.
- DBpedia publica vínculos RDF que apuntan a otras fuentes de datos en la Web, lo que ha dado lugar a la aparición de una Web de datos bastante amplia.

DBpedia permite realizar consultas sofisticadas sobre los conjuntos de datos o “datasets” derivados de Wikipedia y los vincula con otros conjuntos de datos de la Web. Estos Datasets pueden ser importados en aplicaciones de terceros o pueden ser accedidos en línea usando cualquier interfaz de consulta de DBpedia (Auer, S., et al., 2007). En la sección 1.3.10 se mencionan las distintas formas de acceso para el consumo de la información en DBpedia.

1.3.8.1. Conjunto de datos DBpedia

El conjunto de datos o dataset DBpedia provee información de alrededor de 1,95 millones de “cosas” tales como: 80.000 personas, 70.000 lugares, 35.000 álbumes musicales, 12.000 películas, 657.000 links a imágenes, 1.600.000 links a páginas web externas relevantes, 180.000 links externos a otros datasets RDF, 207.000 categorías Wikipedia y 75.000 categorías YAGO (Auer, S., et al., 2007).

El conjunto de datos DBpedia posee alrededor de 103 millones de tripletas RDF, y como se mencionó anteriormente, los dataset DBpedia están disponibles para su descarga como un conjunto de archivos RDF más pequeño. La Tabla 7, presenta los principales datasets que posee DBpedia.

Tabla 7. Datasets DBpedia

Tipo Dataset	Descripción
Articles Categories	Enlaza los conceptos con las categorías usando el vocabulario de SKOS. ²
Short Abstracts	Breves resúmenes de artículos de Wikipedia.
Infoboxes	Información que se ha extraído de infoboxes ³ de Wikipedia.
Images	Enlaces en miniatura de artículos de Wikipedia.
External Links	Enlaces a páginas web externas sobre un concepto.
Yago Classes	Conjunto de datos que contienen instancias de DBpedia usando el algoritmo de clasificación YAGO.
Persondata	Información acerca de personas extraídas desde Wikipedia.
Page Links	Conjunto de datos que contiene enlaces internos entre instancias de DBpedia.

Fuente: Autor.

Cada uno de los 1,97 millones de recursos descritos en el dataset DBpedia está identificado mediante una referencia URI, por ejemplo: *http://dbpedia.org/resource/Name*, donde *Name* se refiere al recurso correspondiente a un artículo de Wikipedia.

1.3.9. DBpedia: framework de extracción del conocimiento

Los artículos de Wikipedia básicamente son estructuras de texto libre con una información adicional de marcado, denominado “marcado wiki”. Esta información de marcado contiene datos relevantes relacionados a cada artículo. Lo que hace DBpedia es extraer esta información de Wikipedia y la transforma en una base de conocimientos. (Bizer, C., et al., 2009).

EL framework de extracción del conocimiento de DBpedia está compuesto de varios elementos, los mismos que permiten la extracción, almacenamiento y conversión de los datos que se obtienen en el proceso. El núcleo del framework es el gestor de extracción, el cual gestiona el proceso de obtener la información de cada artículo de Wikipedia y de entregar dicha información a su destino. La Figura 11, muestra los componentes del framework de DBpedia.

² SKOS (Simple Knowledge Organization System), permite representar esquemas conceptuales. Disponible en: <http://skos.um.es/acerca/index.php>

³ Las infoboxes presentan un resumen del tema que contiene una página en un formato organizado y que es fácil de leer. Disponible en: http://comunidad.wikia.com/wiki/Ayuda:Infoboxes_clásicas

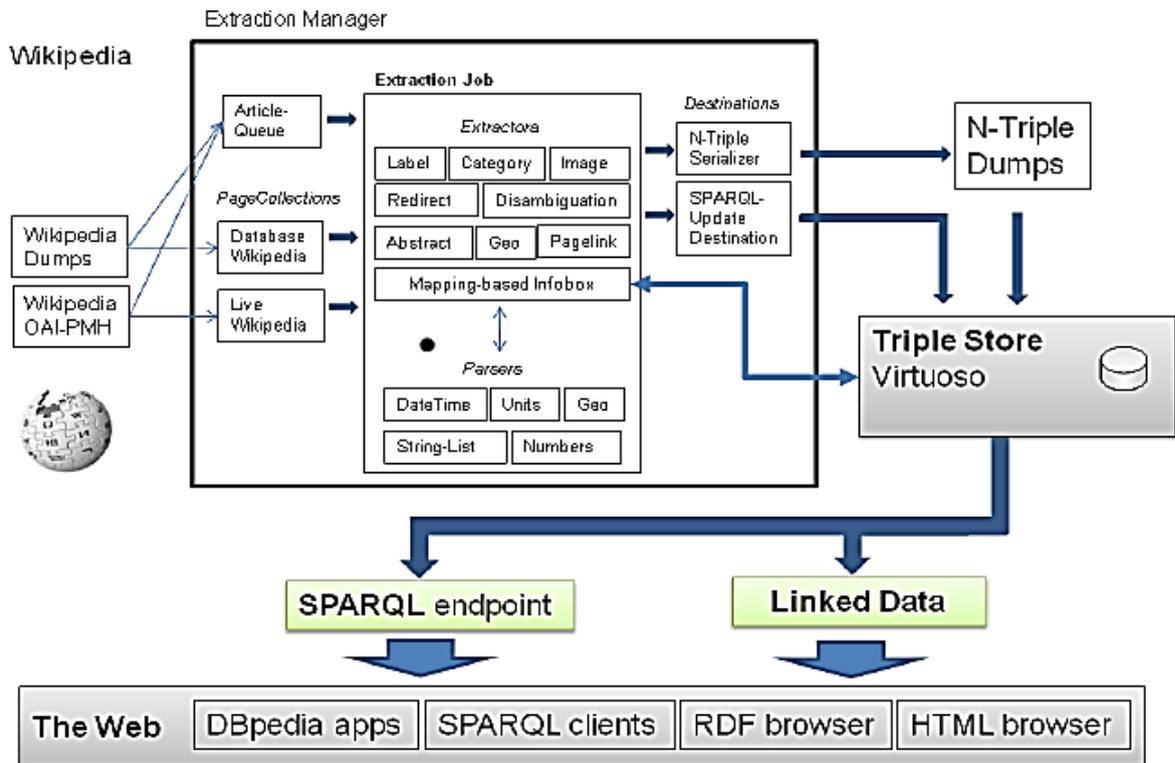


Figura 11. Framework de extracción del conocimiento DBpedia

Fuente: (Bizer, C., et al., 2009)

1.3.9.1. Base de conocimientos DBpedia

La base de conocimiento que DBpedia posee actualmente consta de alrededor de 274 millones de tripletas RDF, las mismas que han sido extraídas de todos los idiomas que soporta Wikipedia. Esta base de conocimiento describe más de 2,6 millones de entidades, con lo que se ha logrado cubrir una amplia gama de temas enciclopédicos. Además, esta identificación de entidades ha permitido que existan políticas claras para su manejo y que cada recurso pueda estar disponible en una localización web conocida como Wikipedia (Bizer, C., et al., 2009). La Tabla 8, muestra un ejemplo de entidades de DBpedia.

Tabla 8. Clases comunes de DBpedia

Ontology class	Instances	Example properties
Person	198.056	name, bithdate, birthplace, employer
Artist	54.262	activeyears, awards, occupation, genre
Actor	26.009	academyaward, goldenglobeaward
MusicianArtist	19.535	genre, instrument, label, voiceType
Athlete	74.832	currentTeam, currentPosition, currentNumber
Politician	12.874	predecessor, successor, party

Fuente: (Bizer, C., et al., 2009)

1.3.10. Acceso a la base de conocimientos de DBpedia

DBpedia se encuentra bajo los términos de la Licencia de Documentación Libre (GNU), es así que su base de conocimiento está abierta a las demandas de cualquier aplicación cliente que desee consumir su información. Por tanto, DBpedia provee cuatro mecanismos de acceso a su base de conocimiento (Bizer, C., et al., 2009):

- **Linked Data**
 - DBpedia resource identifiers(ex: <http://dbpedia.org/resource/Berlin>)

- **SPARQL Endpoint**
 - <http://dbpedia.org/sparql>



- **RDF Dumps**
 - <http://wiki.dbpedia.org/Downloads32>
- **Lookup Index**
 - <http://lookup.dbpedia.org/api/search.asmx>

Figura 12. Acceso a la DBpedia mediante la web

Fuente: (Bizer, C., et al., 2009)

1.3.10.1. LINKED DATA (Datos Vinculados)

Este es un método de publicación de datos RDF basado en URIs que sirven como identificadores de recursos y en el protocolo HTTP que permite recuperar dichos recursos. Por ejemplo, un identificador de recurso sería <http://dbpedia.org/resource/Ecuador>, el mismo que está configurado para entregar las descripciones RDF del recurso y para proveer una vista simple de la información en HTML al utilizar los navegadores tradicionales. Simplemente es necesario digitar dicho identificador (<http://dbpedia.org/resource/Ecuador>) en cualquier navegador y se obtiene la información del recurso desde DBpedia.

1.3.10.2. SPARQL Endpoint

Este servicio permite realizar consultas directas a la base de conocimiento de DBpedia. De ahí que muchas aplicaciones clientes lo utilizan para enviar consultas a través del protocolo SPARQL. Este servicio se encuentra disponible en <http://dbpedia.org/sparql>. Adicionalmente, el Endpoint hace uso de Virtuoso Universal Server, que es el servidor que actualmente utiliza DBpedia para almacenar y ofrecer sus datos al público. Este servicio se lo puede utilizar ya sea mediante el uso de URI's o a través del Endpoint SPARQL (<http://dbpedia.org/sparql>), donde el elemento principal que debe ser descrito correctamente es la consulta SPARQL, la

misma que permitirá el retorno de los resultados según la serialización especificada que por defecto puede ser un JSON.

1.3.10.3. RDF Dumps

Ofrece serializaciones N-Triple de conjuntos de datos que son extraídos de la Wikipedia. Estos conjuntos de datos pueden ser utilizados como base para fusionar conocimientos que se obtienen de Wikipedia y que pueden ser utilizados en aplicaciones que basan su conocimiento en los recursos de la misma fuente.

1.3.10.4. Lookup index (Índice de búsqueda)

Este es un servicio que facilita la búsqueda de los recursos de DBpedia, puesto que propone URIs para una determina etiqueta o título de búsqueda. El servicio se basa en la utilización de un índice que proporciona una búsqueda de etiquetas con una clasificación de relevancia, lo que permite encontrar las coincidencias más probables para un determinado término en DBpedia. Su función está basada mediante el uso de palabras claves incrustadas en la misma URL de petición.

1.3.11. Ventajas de la DBpedia

DBpedia es un proyecto que surgió con el objetivo de organizar la información de Wikipedia y transformarla en recursos y descripciones utilizables en Linked Open Data, pero además, para consolidar a la Wikipedia como un recurso de la Web Semántica. DBpedia es el mayor concentrador de vínculos de la Web, ya que posee enlaces a un gran número de vocabularios y conjuntos de datos (Agenjo, X., Hernandez, F., 2015).

La ventaja más importante de la DBpedia es que permite consumir la información directamente de la fuente, de esa manera, el usuario final no necesita de ninguna intermediación. DBpedia proporciona información contextual, la misma que es directamente utilizable y cuya potencia informativa aumenta enormemente al ser combinada con otros datos vinculados.

1.4. Tecnología Móvil

En el continuo avance de las tecnologías móviles, las aplicaciones o apps móviles son las que marcan un mayor y rápido crecimiento en el mercado global de la telefonía móvil. Estas tecnologías están evolucionando a un ritmo muy veloz con la finalidad de dar a los usuarios una mejor experiencia de uso (Holla, S., & Katti, M., 2012).

La tecnología móvil es considerada como un medio innovador que ha surgido para mejorar las comunicaciones entre los usuarios. La constante y progresiva evolución de esta tecnología ha permitido el mejoramiento del diseño, tamaño, peso y funciones de procesamiento, y además, la capacidad de utilizar un mayor conjunto de aplicaciones móviles (Bustos, M., Perez, N., & Berón, M., 2015).

1.4.1. Sistema Operativo Android

Uno de los sistemas operativos que mayor relevancia ha tenido en el avance tecnológico, y que con seguridad se ha convertido en el Sistema Operativo más utilizado en teléfonos móviles, es el SO Android. Fue Google quien lanzó Android que es un sistema operativo de telefonía móvil con una plataforma basada en Linux, y que además, brinda una infraestructura que permite desarrollar aplicaciones móviles usando el lenguaje de programación Java (Holla, S., & Katti, M., 2012).

Android es un sistema operativo de código abierto diseñado para dispositivos móviles, el mismo que permite desarrollar un sin número de aplicaciones para teléfonos inteligentes, tabletas, reproductores de MP3, cámaras, televisores. Este sistema operativo ha tenido un crecimiento acelerado desde su primera versión 1.0 que fue lanzada el 23 de septiembre del 2008. Se puede decir, que actualmente Android estaría activado en más de 200 millones de dispositivos móviles y se estima que cada día se activan otros 550.000 nuevos dispositivos con este SO. (Vanegas, C., 2013).

1.4.2. Arquitectura Android

Android posee una arquitectura que se encuentra organizada en cuatro capas, las mismas que permite la generación de aplicaciones. Las respectivas librerías permiten el acceso a cada capa, y a su vez, cada capa puede utilizar los componentes de la capa inferior para realizar sus funciones. Las capas que componen la arquitectura de Android son (Vanegas, C., 2013):

- **Linux Kernel:** es la capa de abstracción del hardware, que permite que las aplicaciones puedan acceder a los controladores (drivers) para la administración de los recursos del teléfono y del sistema operativo.

- **Librerías:** son bibliotecas de Android que se encargan de realizar la comunicación entre la capa de abstracción de hardware con la interfaz de programación de las aplicaciones.
- **Entorno de ejecución Android:** no es considerada como una capa, sin embargo, en ella se encuentran la Máquina Virtual Dalvik, y el Núcleo de Librerías.
- **Marco de trabajo de aplicaciones:** en ella se encuentran las clases y los servicios que permiten las funciones básicas de los móviles y la programación de las aplicaciones.

La Figura 13, muestra la Arquitectura del SO Android.

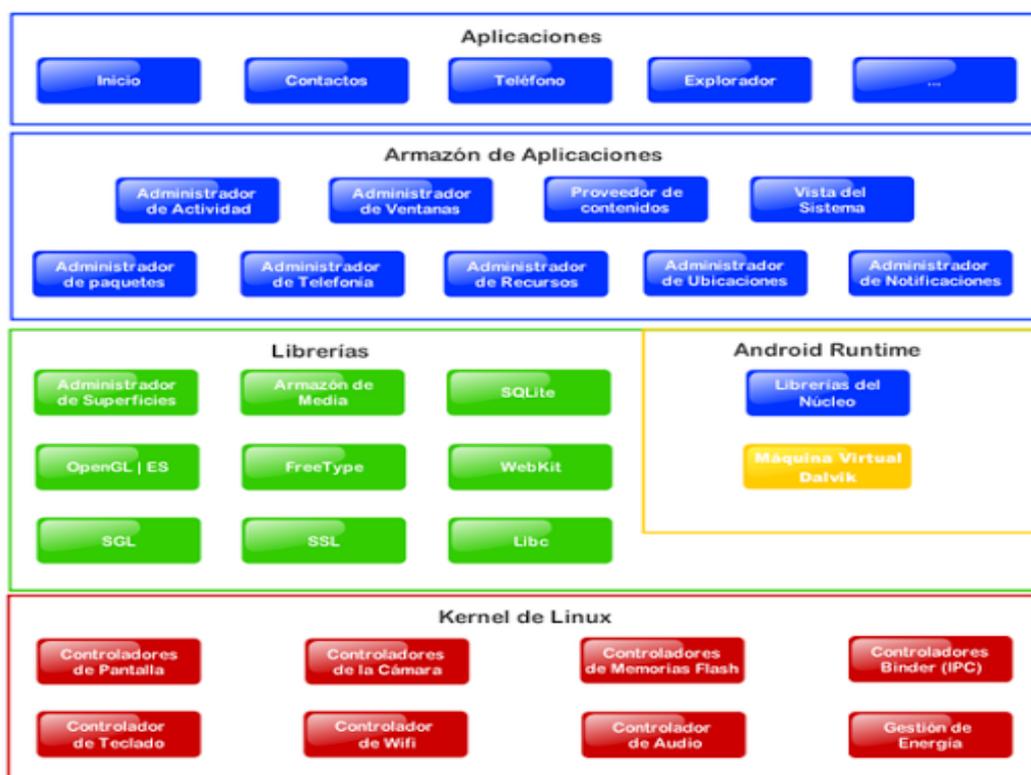


Figura 13. Arquitectura del SO Android

Fuente: (Bustos, M., Perez, N., & Berón, M., 2015).

1.4.3. Desarrollo de Aplicaciones en Android

El SDK (*Kit de desarrollo de software*) de Android provee las herramientas y APIs (*Interfaz de programación de aplicaciones*) que se necesitan para el desarrollo de aplicaciones en la plataforma de Android y el lenguaje de programación que utiliza es Java (Vanegas, C., 2013).

Si se desea utilizar Microsoft Windows para el desarrollo de aplicaciones Android, se puede utilizar el software Eclipse IDE para Java. Este software es el recomendado por Android para crear aplicaciones; adicionalmente, es necesario instalar los siguientes componentes: a) Java

Development Kit (JDK), b) SDK Android, c) Eclipse IDE for Java Developers, d) Plug-in Android para Eclipse.

Al tratarse de una plataforma abierta, Android ofrece a los desarrolladores la posibilidad de crear aplicaciones sumamente ricas e innovadoras. Los desarrolladores pueden aprovechar el hardware del dispositivo, acceder a la información de ubicación, ejecutar servicios, establecer alertas, agregar notificaciones de barra de estado y crear muchas otras funcionalidades (Holla, S., & Katti, M., 2012).

1.4.4. Características de una Aplicación Móvil

Existe una estructura definida implementada por Google para el desarrollo de una aplicación Android. De forma general, estos componentes son (Bustos, M., Perez, N., & Berón, M., 2015):

- **Activity:** una "Actividad" es la encargada de mostrar al usuario la interfaz gráfica, en otras palabras, una actividad es el equivalente a una ventana, eso la convierte en el medio de comunicación entre la aplicación y el usuario.
- **Listeners:** son mensajes que producen notificaciones o cambios de estado. Son los encargados de unir componentes dentro de la misma aplicación o de otras externas. Los *Listeners* son importantes para el desarrollo de aplicaciones móviles, especialmente para aquellas que son desarrolladas para dispositivos de pantalla táctil.
- **Views:** son componentes de la interfaz de usuario.
- **Service:** son componentes que ejecutan operaciones en segundo plano y no tienen una interfaz de usuario.
- **Content Provider:** es una clase que permite que a las aplicaciones consultar, guardar o modificar la información.
- **Manifest:** en el archivo *AndroidManifest.xml* se guarda la configuración de la aplicación.
- **Broadcast Receivers:** son componentes que responden a avisos de difusión, como por ejemplo, el mensaje de batería baja. Estos avisos provienen del sistema como también de la misma aplicación. Aunque no poseen interfaz de usuario, algunas veces pueden hacer uso de barras de progreso para mostrar avances.

1.4.5. Tecnologías y herramientas complementarias

Vamos a denominar como tecnologías complementarias, a aquellos componentes que brindan soporte adicional o son un complemento para el desarrollo e implementación de la tecnología móvil.

1.4.5.1. Códigos QR (Quick Response Code)

Los conocidos códigos QR o “códigos de respuesta rápida”, fueron creados en Japón en 1994. Originalmente, Toyota usó estos códigos para rastrear las piezas de sus automóviles y permitir que las mismas sean escaneadas a gran velocidad durante el proceso de fabricación. En el año 2000, el código QR fue aprobado como un estándar internacional ISO (ISO/IEC 18004) por la Organización Internacional de Normalización (Xu, F., 2014).

El QR es un símbolo bidimensional, pues consta de dos partes que comprenden los patrones de función y una región de codificación. Los patrones de función permiten la lectura en todas las direcciones (360°), detectando la posición del código QR y corrigiendo la distorsión del mismo; y la región de codificación se encarga de almacenar los datos y caracteres del código QR. La información que es codificada mediante QR puede contener diferentes tipos de datos, tales como caracteres numéricos y alfabéticos, símbolos, códigos binarios y de control. Un código QR puede almacenar hasta 7089 caracteres.

Gracias a su bajo costo, su fácil implementación y uso, esta tecnología tiene una gran potencial para su aplicación en dispositivos móviles inteligentes y compatibles con la Web. La Figura 14, muestra la estructura de un QR.

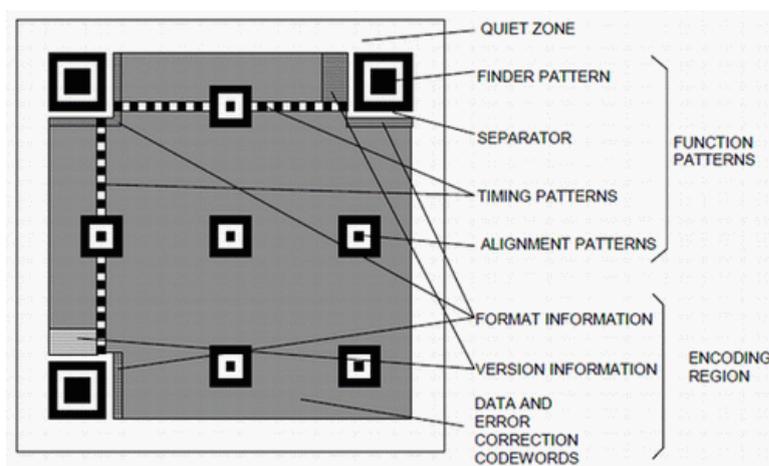


Figura 14. Estructura de un símbolo QR

Fuente: (Xu, F., 2014)

1.4.5.2. Android Studio IDE

El éxito de los desarrolladores de Android radica en la buena comprensión que tengan del lenguaje JAVA, de las API's y la propia arquitectura de Android. Esto apunta a la utilización de un adecuado y eficaz entorno de desarrollo. Es así que durante muchos años, Eclipse IDE juntamente con el complemento ADT (Android Development Tool plugin) que contenía las herramientas y librerías de desarrollo, fue la plataforma preferida para el desarrollo de aplicaciones para Android (Friesen, J., 2016).

Actualmente, Android Studio IDE (Entorno de Desarrollo Integrado) es la plataforma oficial de Google para el desarrollo de aplicaciones para Android. Esta plataforma está basada en IntelliJ IDEA⁴, que es un IDE integrado que permite el análisis del código escrito, mejora la codificación, proporciona una navegación rápida y brinda un análisis de errores inteligente (*developer.android.com.*, 2017).

Dentro de las características más generales que ofrece Android Studio están las siguientes:

- Cuenta con un entorno unificado donde se pueden desarrollar aplicaciones para todos los dispositivos Android.
- Soporta la creación de aplicaciones Android para TV y Relojes Inteligentes.
- Posee asistentes basados en plantillas para crear diseños y componentes comunes.
- Ofrece un enriquecido editor de diseño que permite a los desarrolladores arrastrar y soltar los componentes para la construcción de la interfaz gráfica, y genera una previa de visualización del diseño en varias configuraciones de pantalla.

Actualmente, la última versión de Android Studio (2.3.1-Abril 2017), ofrece las siguientes bondades (*developer.android.com.*, 2017):

- ✓ Sistema de compilación de Gradle, que es la base del sistema de compilación de Android Studio, lo que permite personalizar y configurar el proceso de compilación.
- ✓ Un emulador rápido que permite probar la ejecución de aplicaciones.
- ✓ La función "Instant Run" que permite realizar modificaciones al mismo tiempo que la app se está ejecutando sin tener que volver a compilar un nuevo APK⁵.
- ✓ Posee herramientas Lint que detectan problemas de rendimiento, usabilidad y compatibilidad de las versiones.

⁴ IntelliJ IDEA. Disponible en: <https://www.jetbrains.com/idea/features/>

⁵ Un APK es básicamente un archivo comprimido. Disponible en: [https://es.wikipedia.org/wiki/APK_\(formato\)](https://es.wikipedia.org/wiki/APK_(formato))

- ✓ Brinda un soporte incorporado para Google Cloud Plataform⁶, la misma que aliviana la carga de la administración de la infraestructura, de servidores y redes, pues permite colocar todo en servicios de la nube.

1.4.5.3. HTML5, CSS3 & JavaScript

Uno de los asuntos que se remonta a los inicios de la web, ha sido la integración de contenido multimedia en las páginas web. Es así que la web multimedia ha venido evolucionando muy rápidamente y con HTML5 ha sido posible que hoy en día, las páginas web puedan integrar gráficos, sonido y video sin problema. Esto además ha permitido que dichos contenidos multimedia puedan ser reproducidos en una gran cantidad de dispositivos que van desde equipos de escritorio tradicionales hasta dispositivos móviles de todo tipo (Cazenave, F., Quint, V., & Roisin, C., 2011).

HTML5 se complementa con el lenguaje de hojas de estilo denominado CSS3, que permite a los desarrolladores web separar de forma limpia el contenido y diseño de las páginas. Es decir, que HTML5 permite describir el contenido en una estructura lógica, mientras que CSS3 permite diseñar la presentación, y entonces, es posible separar el contenido de la presentación de una página web. Adicionalmente, JavaScript permite manejar la sincronización e interacción de los usuarios.

Es importante señalar que HTML5 es compatible y se ejecuta de forma nativa en los navegadores web modernos, mientras que el código JavaScript es ejecutado por el motor de JavaScript del navegador. Por lo tanto, el enfoque más frecuente para la construcción de páginas web multimedia y que sean adaptables a cualquier dispositivo móvil se basa en HTML, CSS y JavaScript.

⁶ Google Cloud Plataform. Disponible en: <https://cloud.google.com/why-google/?hl=es-419>

1.5. Trabajos Relacionados

Actualmente, encontramos varios trabajos relacionados con la extracción o visualización de información de la DBpedia y algunos de ellos hacen uso de dispositivos móviles. En estos trabajos se puede observar que ha sido posible implementar mecanismos de localización de lugares, búsqueda de información, y en general, todo tipo de información que tiene relación con la DBpedia.

1.5.1. Metodología de búsqueda

Estos trabajos fueron encontrados gracias al uso de las bases científicas de la Universidad Técnica Particular de Loja, tales como:

- ❖ Science Direct,
- ❖ ProQuest,
- ❖ ISI Web of Knowledge,
- ❖ SpringerLink

Se aplicó la metodología de la revisión sistemática de literatura (SLR)⁷, donde en la estrategia de búsqueda se utilizaron palabras claves como: “dbpedia extraction”, “dbpedia mobile”, “related works dbpedia” y donde el año de publicación estuviera especificado a partir del año 2014 en adelante. De esa manera, se obtuvieron todos los estudios publicados y que a continuación presentaremos algunos de estos trabajos relacionados que fueron analizados.

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

Una técnica potencial para presentar la información son las “tag clouds”, puesto que muestran la información en lenguaje natural o temas comprensibles. PediaCloud es una herramienta construida en Android, que permite crear y visualizar tag clouds de la información localizada en DBpedia, estas tag clouds permiten a los usuarios seguir las asociaciones de la Web Semántica disponibles para cada tag o etiqueta. Esto lo realiza PediaCloud utilizando artículos de Wikipedia con su ubicación, y enlaces secundarios a los que hacen referencia desde la DBpedia (Bjornar, T., Bjarte, J., Csaba, V., 2013).

1.5.3. Mobile Facets Application

Mobile Facets Application es un prototipo de aplicación móvil que permite la búsqueda y exploración de lugares, personas, organizaciones y eventos, el mismo que puede ser usado en cualquier teléfono móvil con pantalla táctil. Para realizar la búsqueda y exploración, Mobile Facets Application hace uso de temáticas o facetas, vista de mapas, listas de resultados y

⁷ La Metodología SLR (Systematic Literature Review), permite identificar y extraer de manera correcta información de un determinado ámbito de estudio para obtener respuestas fiables que aporten a la investigación. Disponible en: <http://www.rte.espol.edu.ec/index.php/tecnologica/article/view/430/297>

vistas de fotos. Una vez obtenidos los recursos, éstos son presentados detalladamente de forma visual. Un punto importante de esta aplicación, es que hace uso de DBpedia para obtener los recursos más relevantes (Kleinen, A., Scherp, A., & Staab, S., 2014).

1.5.4. A Self-medication web application built on Linked Open Data

Se trata de una aplicación web denominada Open Self-Medication, la misma que informa a las personas, de una manera conveniente y segura, como auto-medicarse para tratar dolencias en el caso de no contar con una receta o con el diagnóstico de un médico. La característica principal de esta aplicación es que todos los datos obtenidos y procesados por el sistema y que son presentados al usuario, provienen de varios conjuntos de datos semánticos entre ellos DBpedia (Curé, O., 2014).

1.5.5. VOX System: a semantic embodied conversational agent exploiting Linked data

VOX System es una plataforma que se basa en técnicas de reconocimiento de voz para obtener la entrada de usuario, generando de esa forma las palabras claves relevantes para realizar la búsqueda. Es así, que Vox es capaz de utilizar modos verbales y no verbales de comunicación para interactuar con los usuarios. La búsqueda la realiza explotando las fuentes de información actualmente disponibles en forma de Linked Data, y permite realizar además búsquedas híbridas basadas en palabras clave y búsqueda de datos semánticos estructurados (Serón, F., & Bobed, C., 2014).

1.5.6. Smart places: Multi-agent based Smart mobile virtual community management system

Es una aplicación móvil (SMVCMS) que permite la construcción y gestión de “Smart places” o lugares inteligentes en una ciudad, hospital universidad, edificio, conferencia, festival, etc., equipándolos con servidores comunitarios los mismos que proporcionan servicios a sus habitantes mediante recomendaciones. Su objetivo principal es implementar un enfoque de intercambio y adquisición de conocimientos en el contexto de Smart places, obteniendo información referente a cada servicio que se ofrece desde repositorios semánticos (Fahad, M., Boissier, O., Maret, P., Moalla, N., & Gravier, C., 2014).

1.5.7. DBpedia Viewer: An Integrative interface for DBpedia Leveraging the DBpedia Service Eco System

DBpedia Viewer tiene como objetivo mostrar de una manera más atractiva la información que se puede obtener de DBpedia. Esta herramienta integra los servicios existentes que posee la DBpedia con otras herramientas de visualización de Linked Data externas, lo cual permite una mejora en la facilidad de uso al momento de mostrar los datos al usuario (Lukovnikov, D., Kontokostas, D., Stadler, C., Hellmann, S., & Lehmann, J., 2014).

1.5.8. Linked tag: image annotation using semantic relationship between image tags

Linked tag es un sistema prototipo de anotación de imágenes semiautomático que inserta relaciones entre sus etiquetas. Es decir, el método de anotación conecta las etiquetas de las imágenes utilizando palabras que se relacionan con el contexto de la etiqueta que utiliza cada imagen. En general, utiliza datos vinculados de DBpedia para conectar las etiquetas de las imágenes con un valor de propiedad (Im, D., & Park, G., 2015).

1.5.9. A framework to support educational decision making in mobile learning

Este trabajo busca desarrollar un framework que permita relacionar la interacción que ocurre en una actividad de aprendizaje móvil con las tareas pedagógicas que son relevantes para el aprendizaje de cualquier estudiante. Para el caso de estudio, se hace uso de MeLOD (Mobile Environment for learning with Linked Open Data), el cual es un entorno que proporciona una metodología de aprendizaje basado en tareas de aprendizaje simples y que proporciona contenido didáctico según la necesidad que tenga el alumno. MeLOD enlaza los conjuntos de datos de DBpedia, Europeana y GeoNames para poder buscar y obtener los materiales de aprendizaje necesarios (Fulantelli, G., Taibi, D., & Arrigo, M., 2015).

1.5.10. ONLI: an ontology-based system for querying DBpedia using natural language paradigm.

Es un sistema basado en semántica que permite recuperar información de DBpedia a través de preguntas expresadas en lenguaje natural. Este enfoque considera la utilización de elementos textuales, en este caso, preguntas que se procesan mediante técnicas de lenguaje natural como sinónimos o palabras claves. Estas palabras son consultadas para conocer el contexto de la pregunta y la información es organizada en un modelo ontológico que procesa las posibles ambigüedades, donde también se deduce el tipo de respuesta esperado para reducir el espacio de la búsqueda. Finalmente, las preguntas se transforman a consultas SPARQL y las respuestas obtenidas son mostradas al usuario. Con este enfoque el usuario no necesita conocer los vocabularios especializados o las estructuras de las bases de conocimiento (Paredes, M., Rodríguez, M., Ruiz, A., Valencia, R., & Alor, G., 2015).

1.5.11. Knowledge Structuring and Reuse System Using RDF for Supporting Scenario Generation

Este trabajo muestra la creación de una base de datos para la reutilización y recuperación de conocimiento mediante escenarios, los mismos que contienen información derivada de datos y conocimientos relacionados a la resolución de conflictos y reducción de riesgos al tomar decisiones. Todo este conocimiento está estructurado con RDF que también almacena información obtenida de DBpedia y Data Jacket Store. El objetivo principal de este sistema es

que permita recuperar información importante respecto a la resolución de problemas (Hayashi, T., & Ohsawa, Y., 2015).

1.5.12. Memo Graph: an ontology visualization tool for everyone

Memo Graph es una aplicación web semántica que permite la visualización y navegación sobre ontologías OWL/RDF a través de gráficos. Su principal funcionalidad es poder ayudar a usuarios con problemas mnésicos, los mismos que tienen que ver con el deterioro cognitivo en edades avanzadas, específicamente de quienes padecen de Alzheimer. Esto se logra mediante un conjunto de ejercicios de relación que el usuario debe realizar utilizando información que es obtenida de la DBpedia y cuya ontología genera los gráficos (Ghorbel, F., Ellouze, N., Métais, E., Hamdi, F., Gargouri, F., & Herradi, N., 2016).

1.5.13. Mobile services discovery framework using DBpedia and non-monotonic rules

Es un framework que permite el descubrimiento de servicios móviles, tales como: servicios de restaurantes, taxis, atención médica, mecánicas, etc., los mismos que son publicados para dispositivos móviles. Esto se logra representando las propiedades funcionales de los servicios móviles mediante una ontología que representa el contexto y la calidad de la información del servicio que la obtiene de la DBpedia y que permite describir las propiedades funcionales o capacidades de los servicios móviles (Cheniki, N., Belkhir, A., & Atif, Y., 2016).

1.5.14. QueryGen: Semantic interpretation of keyword queries over heterogeneous information systems

QueryGen es un sistema que realiza interpretaciones semánticas de las consultas de palabras clave en múltiples lenguajes de consulta sobre diferentes repositorios de datos. Su objetivo es descubrir el significado de las palabras claves que son ingresadas y con ellas elaborar la sintaxis de la consulta semántica apropiada. Finalmente, una vez que ya se cuenta con la estructura de la consulta validada, el sistema dirige la consulta al repositorio semántico más apropiado de acuerdo a los datos que se necesitan obtener (Bobed, C., & Mena, E., 2016).

1.5.15. ReDyAI: A Dynamic Recommendation Algorithm based on Linked Data

ReDyAI es un algoritmo basado en Linked Data que explota las relaciones existentes entre los recursos con la finalidad de recomendar otros recursos que tienen relación entre sí. Esto lo logra analizando dinámicamente las categorías y las referencias explícitas que poseen los recursos. Este algoritmo se puede aplicar sobre cualquier conjunto de datos pero principalmente se lo aplica sobre DBpedia, el mismo que fue implementado en una aplicación móvil, y que permite recomendar películas con la información de DBpedia (Vagliano, I., et al., 2016).

1.5.16. Short Query Expansion for Microblog Retrieval

Con este trabajo se busca presentar diferentes enfoques para realizar consultas breves. Uno de los enfoques consiste en acceder a la DBpedia a través de la web y poder extraer todos los conceptos relacionados para una consulta dada, mediante una consulta SPARQL que permita obtener toda la ontología sobre un tema específico. La idea principal de este enfoque es hacer coincidir los términos de la consulta SPARQL con los conceptos de DBpedia para encontrar términos candidatos para realizar una consulta alternativa (Zingla, M., Chiraz, L., & Slimani, Y., 2016).

La Tabla 9, presenta una descripción complementaria de cada trabajo relacionado.

Tabla 9. Trabajos Relacionados

Trabajo	Tecnología utilizada	Datasets/Sources
PediaCloud	<ul style="list-style-type: none"> • SPARQL Endpoint • Android 	<ul style="list-style-type: none"> • DBpedia
Mobile Facets Application	<ul style="list-style-type: none"> • REST Services • SPARQL Endpoint 	<ul style="list-style-type: none"> • DBpedia • GeoNames • Eventful • Upcoming • Flickr
Open Self-medication	<ul style="list-style-type: none"> • HTML5, CSS3 • PHP, MySQL • JavaScript, JQuery • Google Map API v.3 • REST Services 	<ul style="list-style-type: none"> • DrugBank • DailyMed • Sider, LinkedCT • FreeBase • DBpedia
VOX System	<ul style="list-style-type: none"> • SPARQL Endpoint 	<ul style="list-style-type: none"> • DBpedia
Smart places	<ul style="list-style-type: none"> • Android • External Frameworks 	<ul style="list-style-type: none"> • WordNet • DBpedia
DBpedia Viewer	<ul style="list-style-type: none"> • SPARQL queries • HTML/RDFa • JavaScript • AngularJS • Virtuoso Server Pages • CSS 	<ul style="list-style-type: none"> • DBpedia
Linked tag	<ul style="list-style-type: none"> • Java, Jena Framework • MySQL • SPARQL queries, RDF 	<ul style="list-style-type: none"> • DBpedia

A framework to support educational decision making in mobile learning	<ul style="list-style-type: none"> • Web Services • RDF 	<ul style="list-style-type: none"> • DBpedia • Europeana • GeoNames
ONLI	<ul style="list-style-type: none"> • SPARQL queries • Sesame • Java 	<ul style="list-style-type: none"> • DBpedia
Memo Graph	<ul style="list-style-type: none"> • Java • RDF 	<ul style="list-style-type: none"> • DBpedia
Knowledge Structuring and Reuse System Using RDF for Supporting Scenario Generation	<ul style="list-style-type: none"> • RDF • SPARQL Queries 	<ul style="list-style-type: none"> • DBpedia • Data Jacket Store
Mobile services discovery framework	<ul style="list-style-type: none"> • Jena ARQ • Jena TDB RDF Store • JNOMO • SPARQL Endpoint 	<ul style="list-style-type: none"> • DBpedia
QueryGen	<ul style="list-style-type: none"> • Databases • Linked Data Endpoints • Query Languages 	<ul style="list-style-type: none"> • Watson • DBpedia • WordNet
ReDyAI	<ul style="list-style-type: none"> • Android • REST Services • JSON • Visualization Framework 	<ul style="list-style-type: none"> • DBpedia
Short Query Expansion for Microblog Retrieval	<ul style="list-style-type: none"> • SPARQL Queries 	<ul style="list-style-type: none"> • DBpedia

Fuente: Autor

CAPITULO II:
Propuesta de la Solución

2.1. Introducción

Este capítulo aborda la problemática actual y plantea una propuesta de solución basada en la información obtenida en la investigación previa y del análisis de los trabajos relacionados. Además, se explica la arquitectura utilizada, se muestran los objetivos que se desean alcanzar y finalmente se describe la metodología de desarrollo que se utilizará para desarrollar esta propuesta de solución.

2.2. Problemática

El uso de herramientas y tecnologías de la Web Semántica en los últimos tiempos ha venido en constante crecimiento, llegando a desarrollarse innumerables mecanismos relacionados con la búsqueda, organización, extracción, etc., dentro de las grandes cantidades de información que posee la web. Sin embargo, a pesar de que la Web Semántica es una tecnología muy reconocida y estudiada actualmente, ésta todavía no ha llegado a estar al alcance de todos, puesto que la misma requiere de cierto conocimiento técnico para que pueda ser utilizada. Por tanto, se puede evidenciar que el acceso al conocimiento y la información que ofrece la Web Semántica en repositorios semánticos tales como DBpedia, presenta ciertas limitaciones para que cualquier usuario la pueda obtener fácilmente como al utilizar un buscador común (desde Google), y especialmente, a través de un dispositivo móvil de Android.

2.3. Soluciones planteadas en trabajos relacionados

Al final del capítulo anterior, se realizó un análisis de varios estudios relacionados con el consumo y visualización de información proveniente de la DBpedia utilizando mecanismos y tecnologías muy diversas. Una cuestión interesante que se pudo evidenciar en estos trabajos, es que muy pocos de éstos tratan o intentan profundizar el tema del consumo y visualización de datos semánticos utilizando dispositivos móviles, específicamente, en aplicaciones construidas para el sistema operativo Android.

Es importante destacar que todos estos trabajos relacionados han generado un aporte muy valioso en el campo de su aplicación, y por tanto, se vuelve necesario analizar las principales soluciones que estos trabajos proporcionan para poder proponer una nueva alternativa de solución que nos permita crear una aplicación móvil para Android y con la cual los usuarios podrán realizar consultas semánticas y visualizar los resultados obtenidos.

A continuación, vamos a listar y resumir las principales soluciones planteadas en los trabajos relacionados:

- Para la extracción o consumo de los datos de la DBpedia se implementaron:
 - Consultas SPARQL a través de Servicios Web REST

- SPARQL Endpoint
- Consultas SPARQL sobre archivos RDF
- Librerías o bibliotecas (Jena ARQ, Sesame)
- Para la visualización de los datos obtenidos se utilizaron:
 - Páginas HTML/RDFa y JavaScript
 - API-Frameworks (Google Map API, Browsers de Linked Data)
 - Serializaciones RDF/XML y JSON
- Adicionalmente, algunas soluciones proponen almacenar la información para crear repositorios semánticos mediante:
 - Bases de datos (MySQL)
 - Linked Data Endpoint

En conclusión, podemos observar que todas las soluciones logran cumplir el objetivo de consumir y visualizar información de la DBpedia. Sin embargo, no todos los trabajos se enfocan en implementar soluciones para ambientes móviles como Android, puesto que algunos de estos mecanismos fueron desarrollados para otros ambientes distintos al móvil, y por otro lado, muchas de estas soluciones tampoco dan prioridad a la facilidad de uso que requiere un usuario común.

2.4. Propuesta de solución

Con el análisis previo, la propuesta de solución se enfocará a encontrar o desarrollar los mecanismos más adecuados que se acoplen y que puedan ser implementados para la construcción de una aplicación móvil para dispositivos Android. De esa manera, el usuario común podrá obtener información proveniente de un repositorio semántico (DBpedia), haciendo uso de un dispositivo móvil mediante la ejecución de consultas semánticas (SPARQL) o utilizando palabras claves (keywords), que le permitan conseguir de una manera fácil, información relevante que podrá ser visualizada en el dispositivo.

Las características principales que contemplará la propuesta de solución, son las que indicamos a continuación:

- **Buscador mediante palabras claves en un ambiente móvil.**
- **Digitación y ejecución de consultas SPARQL en un ambiente móvil.**
- **Lector de códigos QR derivados a consultas SPARQL en un ambiente móvil.**
- **Sitio web del proyecto y codificador de consultas SPARQL a través de códigos QR en un ambiente web.**

2.5. Arquitectura de la solución

De manera general, la arquitectura física de la solución requerirá la utilización de un dispositivo móvil, que haciendo uso de las tres características específicas de la aplicación, se conectará a la DBpedia mediante internet para obtener los resultados según cada consulta realizada por el usuario. Adicionalmente, se desarrollará un sitio web que brindará información acerca del proyecto y que permitirá la codificación de las consultas SPARQL mediante el uso de códigos QR, los mismos que son requeridos por la aplicación móvil para la ejecución automática de las consultas SPARQL. A continuación, se muestra la representación gráfica de dicha arquitectura.



Figura 15. Arquitectura de la propuesta

Fuente: Autor

La Figura 15, muestra que la aplicación móvil contará con tres servicios específicos: a) Búsqueda mediante palabras claves: este mecanismo de búsqueda hace uso del servicio denominado *Lookup DBpedia*, el mismo que permite al usuario digitar “palabras claves” y de esa manera se obtienen resultados relacionados a dichos términos. Muy similar a lo que realiza un buscador normal; b) Búsqueda mediante consultas SPARQL: este mecanismo de búsqueda hace uso del servicio *SPARQL Endpoint*, el cual permite al usuario digitar una consulta SPARQL y ejecutarla. Se obtendrán resultados solamente si la consulta SPARQL está correctamente escrita; c) Búsqueda mediante lectura de códigos QR: es quizá el mecanismo más novedoso de esta aplicación, pues aunque utiliza el mismo servicio *SPARQL Endpoint*, este mecanismo ejecuta la consulta SPARQL automáticamente una vez que es leída por el dispositivo móvil a través del código QR previamente generado. De igual manera, las consultas SPARQL deben estar correctamente escritas para que el proceso de búsqueda sea realizado exitosamente. Es importante indicar que estos tres mecanismos de búsqueda utilizan el protocolo HTTP para enviar la consulta semántica y consumir la información de la DBpedia, donde los resultados se obtienen mediante archivos serializados como son RDF/JSON y RDF/XML.

2.6. Objetivos de la propuesta

Una vez definida la propuesta de solución, es importante identificar los objetivos que se desean alcanzar al desarrollar este trabajo.

2.6.1. Objetivo General

Consumo y visualización de archivos RDF de la DBpedia en dispositivos móviles de Android.

2.6.2. Objetivos Específicos

- Establecer un marco teórico sobre el contexto de la Web Semántica, la representación de los datos semánticos, la caracterización de Linked Data y de la estructura RDF.
- Investigar sobre las fuentes de datos o “Datasets” de la DBpedia y las posibilidades de consumo de sus archivos RDF.
- Investigar sobre el sistema operativo Android y las aplicaciones de desarrollo que pueden ser utilizadas.
- Investigar y analizar los diferentes trabajos relacionados que existen actualmente respecto a esta temática.
- Desarrollar una aplicación móvil para Android que permita consumir la información de la DBpedia, y que permita visualizar los datos obtenidos.

2.7. Metodología de desarrollo

Para poder planificar, organizar y elaborar cada uno de los componentes que contempla nuestra propuesta de solución, es muy importante definir una metodología de desarrollo de software, la misma que aumentará significativamente nuestras posibilidades de éxito para obtener los productos planteados. Por esta razón, adoptaremos la metodología denominada SCRUM.

El marco de desarrollo SCRUM permite a los equipos de trabajo desarrollar productos o proyectos de forma iterativa e incremental. Es así que SCRUM estructura el desarrollo en ciclos de trabajo llamados *Sprint*, los mismos que son periodos cortos de tiempo (1 a 4 semanas). Al inicio de cada Sprint, el equipo de trabajo selecciona varios de los requisitos dados por el cliente desde una lista de prioridades, los mismos que son desarrollados y evaluados. Y al finalizar el Sprint, el equipo revisa y demuestra lo que ha logrado construir con las partes interesadas (Deemer, P., Benefield, G., Larman, C., & Vodde, B., 2010).

Básicamente, la metodología SCRUM comprende las siguientes fases (Schwaber, K., 1997):

- **Planificación:** Comprende la conceptualización y análisis de requerimientos, estimación de tiempo y costos.

- **Arquitectura:** Indica cómo se realizará la implementación de los requerimientos y qué se contemplará en el diseño de alto nivel.
- **Desarrollo de Sprint:** Comprende el desarrollo de las funcionalidades del producto, tomando en cuenta los requerimientos, tiempo, calidad y costos. Es la fase que hace que el sistema evolucione hasta su estado final.
- **Cierre:** Al ser ésta la última fase, comprende la preparación para la implementación y la documentación del producto, la realización de pruebas antes de su lanzamiento y su lanzamiento oficial.

La Planificación y Cierre consisten en procesos bien definidos, su flujo es lineal pero con algunas iteraciones en la fase de Planificación. La fase de Sprint es un proceso empírico pues muchos de sus procesos no son identificados o no están bien controlados. Los Sprint son no lineales y flexibles y éstos permiten alcanzar el producto final. El proyecto se encuentra abierto mientras no se alcance la fase de Cierre; los entregables pueden cambiarse en cualquier momento durante las fases de Planificación y Sprint, además éstos entregables son determinados a lo largo del proyecto en función del entorno de aplicación.

CAPITULO III:
Desarrollo de la Solución

3.1. Introducción

Este capítulo se enfoca en el desarrollo de la propuesta de solución aplicando la metodología SCRUM que se describió en el capítulo anterior. Para esto, se utilizarán los componentes que corresponden a esta metodología de desarrollo, tales como: Product Backlog, Backlog, Planificación de Sprint y Burndown Chart. Adicionalmente, al ser importante la especificación un documento de visión del proyecto, el mismo está detallado en el Anexo 1.

3.2. Product Backlog

También conocido como “pila de requerimientos”, es una herramienta que permite elaborar la documentación de las necesidades o deseos que se espera de un producto software. Básicamente, es la lista de los requerimientos fundamentales de un proyecto. La Tabla 10, muestra el Product Backlog de la propuesta de solución.

Tabla 10. Product Backlog

ID	Requerimientos	Prioridad
P1	Como desarrollador, necesito crear una aplicación móvil que permita el consumo y visualización de archivos RDF de la DBpedia.	1
P2	Como desarrollador, necesito implementar un sitio web que presente la información del proyecto y que permita generar la codificación de consultas SPARQL a través de códigos QR.	2

Fuente: Autor

3.3. Backlog

Esta herramienta permite dividir los requerimientos generales del proyecto en tareas más pequeñas asignándoles un responsable y una estimación de tiempo, así también es posible determinar las tareas que serán agrupadas para poderlas desarrollar en los respectivos Sprint o “incrementos” del proyecto. La Tabla 11, presenta la lista de tareas a desarrollar para cada uno de los ítems del Product Backlog de la presente propuesta.

Tabla 11. Backlog

ID Product Backlog	Tareas		Responsable	Sprint	Estimación (horas)
	ID	Descripción			
	TR001	Aplicar el manejo y uso de Android Studio para el desarrollo de la aplicación móvil.	Cristian A.	1	8
	TR002	Investigar sobre posibles librerías que permitan la conexión desde Android con DBpedia.	Cristian A.	1	8
	TR003	Investigar sobre utilización de Endpoints que permitan establecer una conexión directa desde Android con DBpedia.	Cristian A.	1	8

P1	TR004	Buscar y rediseñar imágenes asociadas a la aplicación.	Cristian A.	1	4
	TR005	Elaborar el logo de la aplicación.	Cristian A.	1	4
	TR006	Obtener los bocetos del diseño de todas las interfaces de usuario de la app móvil.	Cristian A.	1	8
	TR007	Crear la interfaz de la activity_index.	Cristian A.	2	8
	TR008	Crear la interfaz de la activity_about	Cristian A.	2	8
	TR009	Crear la interfaz de la activity_keywords	Cristian A.	2	8
	TR010	Crear la interfaz de la activity_queries_sparql	Cristian A.	2	8
	TR011	Crear la interfaz de la activity_qr_code	Cristian A.	3	8
	TR012	Diseñar menú y botones de navegación de activities.	Cristian A.	3	4
	TR013	Codificar funcionalidades de menú y botones de navegación de activities.	Cristian A.	3	8
	TR014	Crear estilos personalizados para mejorar la visualización de las activities.	Cristian A.	3	8
	TR015	Implementar el servicio de la API Lookup DBpedia y codificar conexión en la activity_keywords.	Cristian A.	4	8
	TR016	Implementar el mecanismo de lectura de los archivos JSON de respuesta y diseñar el visualizador de resultados para la activity_keywords.	Cristian A.	4	8
	TR017	Codificar funcionalidad para el ingreso de las palabras claves y los mensajes de aviso para el usuario en la activity_keywords.	Cristian A.	4	8
	TR018	Realizar pruebas de funcionamiento de la activity_keywords.	Cristian A.	4	8
	TR019	Implementar el servicio DBpedia Endpoint y codificar conexión en la activity_queries_sparql.	Cristian A.	5	8
	TR020	Diseñar el visualizador de resultados para la activity_queries_sparql.	Cristian A.	5	4
	TR021	Codificar funcionalidad para el ingreso de las consultas SPARQL y los mensajes de aviso para el usuario en la activity_queries_sparql.	Cristian A.	5	8
	TR022	Realizar pruebas de funcionamiento de la activity_queries_sparql	Cristian A.	5	8
	TR023	Implementar lector de códigos QR para la activity_queries_sparql.	Cristian A.	6	8

	TR024	Implementar el servicio DBpedia Endpoint y codificar conexión en la activity_qr_code.	Cristian A.	6	8
	TR025	Implementar el mecanismo de lectura de los archivos RDF para obtener resultados en la activity_qr_code	Cristian A.	6	8
	TR026	Diseñar el visualizador de resultados para la activity_qr_code.	Cristian A.	6	8
	TR027	Codificar funcionalidad para realizar lectura de consulta SPARQL a través del lector QR y los mensajes de aviso para el usuario en la activity_qr_code.	Cristian A.	6	8
	TR028	Realizar pruebas de funcionamiento de la activity_qr_code.	Cristian A.	6	8
P2	TR029	Aplicar HTML, CCS3 y JavaScript en la construcción del sitio web.	Cristian A.	7	8
	TR030	Diseñar y codificar página index.html	Cristian A.	7	8
	TR031	Diseñar y codificar página qrcode.html	Cristian A.	7	8
	TR032	Diseñar y codificar página contactos.html	Cristian A.	7	8
	TR033	Maquetar sitio web para que sea adaptable a dispositivos móviles.	Cristian A.	7	8
	TR034	Realizar pruebas de funcionamiento del sitio web.	Cristian A.	7	8

Fuente: Autor

3.4. Planificación y desarrollo de Sprint

Una vez definidas las tareas que serán desarrolladas, éstas son agrupadas y organizadas en los respectivos Sprint, en donde se podrá definir el tiempo de inicio y finalización de cada uno, y además será posible realizar un seguimiento diario a cada una de tareas asignadas y cuantificar el trabajo mediante la gráfica que describe la herramienta Burndown Chart. La tabla siguiente muestra la planificación de Sprint para la propuesta de solución.

Tabla 12. Planificación de Sprint

ID	Descripción	Fecha Inicio	Fecha Fin	Estimación (horas)
S1	Sprint 1	2017-03-06	2017-03-17	40
S2	Sprint 2	2017-03-20	2017-03-29	32
S3	Sprint 3	2017-03-30	2017-04-07	28
S4	Sprint 4	2017-04-10	2017-04-19	32
S5	Sprint 5	2017-04-20	2017-04-28	28
S6	Sprint 6	2017-05-01	2017-05-16	48
S7	Sprint 7	2017-05-17	2017-06-02	48

Fuente: Autor

3.4.1. Sprint 1

El tiempo estimado para completar el Sprint 1 será de 40 horas. Se trabajarán 4 horas por día y 5 días a la semana, por lo que serán necesarios 10 días para poder completar todas las tareas. La tabla siguiente presenta el avance diario de cada una de las tareas que corresponden a este Sprint.

Tabla 13. Avance diario del Sprint 1

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-03-06	6	40	Cristian A.	-	TR001	Pendiente
2017-03-07	6	36	Cristian A.	TR001	TR001	Terminada
2017-03-08	5	32	Cristian A.	TR001	TR002	Pendiente
2017-03-09	5	28	Cristian A.	TR002	TR002	Terminada
2017-03-10	4	24	Cristian A.	TR002	TR003	Pendiente
2017-03-13	4	20	Cristian A.	TR003	TR003	Terminada
2017-03-14	3	16	Cristian A.	TR003	TR004	Terminada
2017-03-15	2	12	Cristian A.	TR004	TR005	Terminada
2017-03-16	1	8	Cristian A.	TR005	TR006	Pendiente
2017-03-17	1	4	Cristian A.	TR006	TR006	Terminada
2017-03-17	0	0	Cristian A.	TR006	-	Fin Sprint 1

Fuente: Autor

3.4.1.1. Gráfico Burndown Chart

A continuación se presenta el gráfico Burndown Chart que permite visualizar el avance diario en el desarrollo de las tareas del Sprint 1.

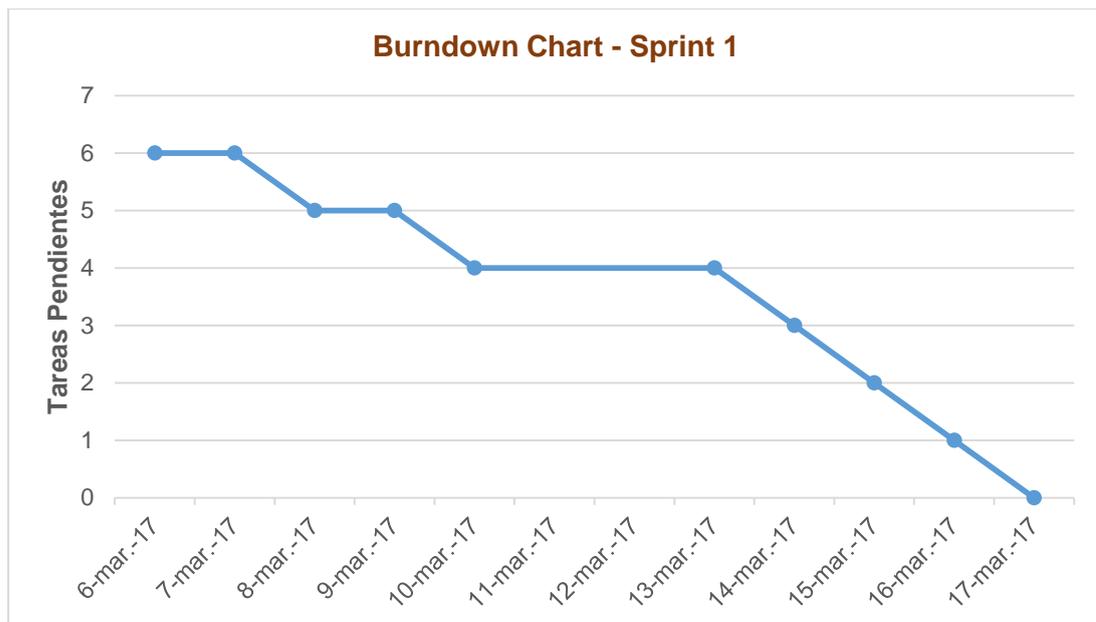


Figura 16. Burndown Chart - Sprint 1

Fuente: Autor

3.4.1.2. Elaboración del incremento

Al concluir el Sprint 1 se realizó la instalación de Android Studio en la máquina de trabajo, esta aplicación nos permitirá la construcción de la aplicación móvil. También se realizó la búsqueda de librerías que pudieran ser utilizadas en Android Studio para realizar el consumo y extracción de datos de la DBpedia. Sin embargo, se concluyó que la mejor opción será utilizar la API y el Endpoint que DBpedia ofrece para realizar una conexión directa a dicho repositorio. Finalmente se realizó el diseño de logos, imágenes y principalmente las interfaces de usuario que se pretenden desarrollar más adelante.

A continuación se muestran los bocetos de diseño para cada una de las interfaces que serán parte de la aplicación móvil, para lo cual se utilizó *NinjaMock*⁸, que es una aplicación gratuita online que permite el desarrollo de estos componentes.



⁸ NinjaMock. Disponible en: <https://ninjamock.com>



Figura 17. Prototipos de las interfaces de usuario de la aplicación móvil

Fuente: Autor

3.4.2. Sprint 2

Se estima completar el Sprint 2 en 32 horas de trabajo, tomando en cuenta que se trabajarán 4 horas por día y 5 días a la semana serán necesarios 8 días para poder completar todas las tareas. La tabla siguiente presenta el avance diario de cada una de las tareas que corresponden a este Sprint.

Tabla 14. Avance diario del Sprint 2

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-03-20	4	32	Cristian A.	-	TR007	Pendiente
2017-03-21	4	28	Cristian A.	TR007	TR007	Terminada
2017-03-22	3	24	Cristian A.	TR007	TR008	Pendiente
2017-03-23	3	20	Cristian A.	TR008	TR008	Terminada
2017-03-24	2	16	Cristian A.	TR008	TR009	Pendiente
2017-03-27	2	12	Cristian A.	TR009	TR009	Terminada
2017-03-28	1	8	Cristian A.	TR009	TR010	Pendiente
2017-03-29	1	4	Cristian A.	TR010	TR010	Terminada
2017-03-29	0	0	Cristian A.	TR010	-	Fin Sprint 2

Fuente: Autor

3.4.2.1. Gráfico Burndown Chart

El gráfico Burndown Chart muestra el cumplimiento de las tareas del Sprint 2.

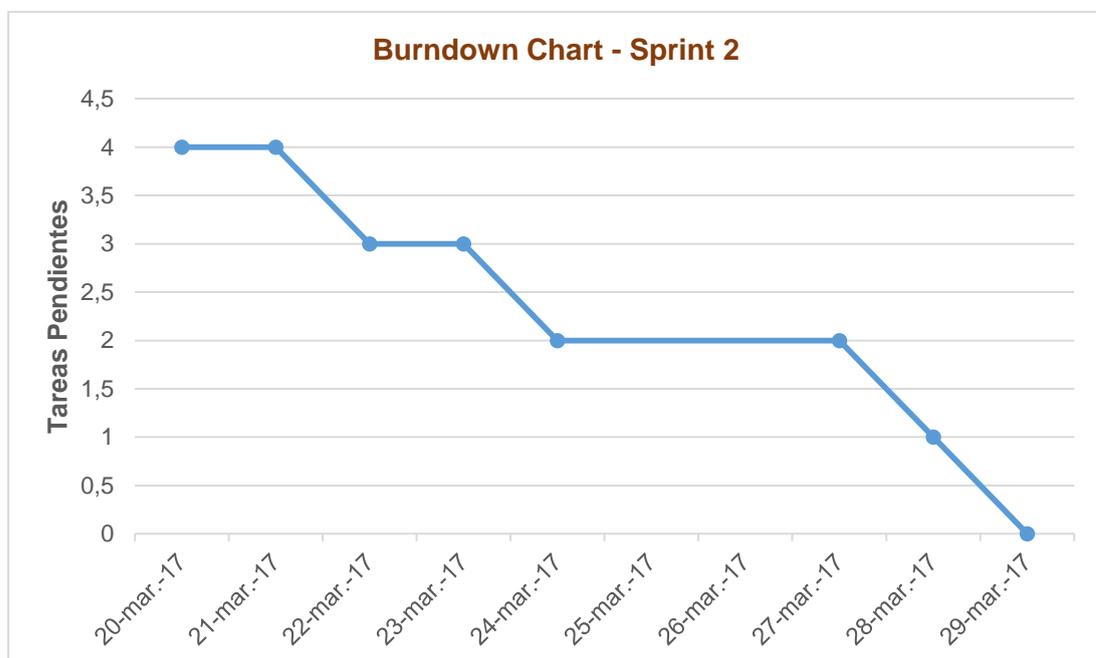


Figura 18. Burndown Chart - Sprint 2

Fuente: Autor

3.4.2.2. Elaboración del incremento

La tarea común realizada en el Sprint 2 fue la construcción de las primeras interfaces de las actividades en Android Studio. Es importante mencionar que Android posee una guía muy detallada denominada *Material Design*⁹, la cual ofrece soporte para el diseño y la construcción de cualquier tipo de componente en Android Studio.

Las interfaces desarrolladas en este Sprint se describen a continuación.

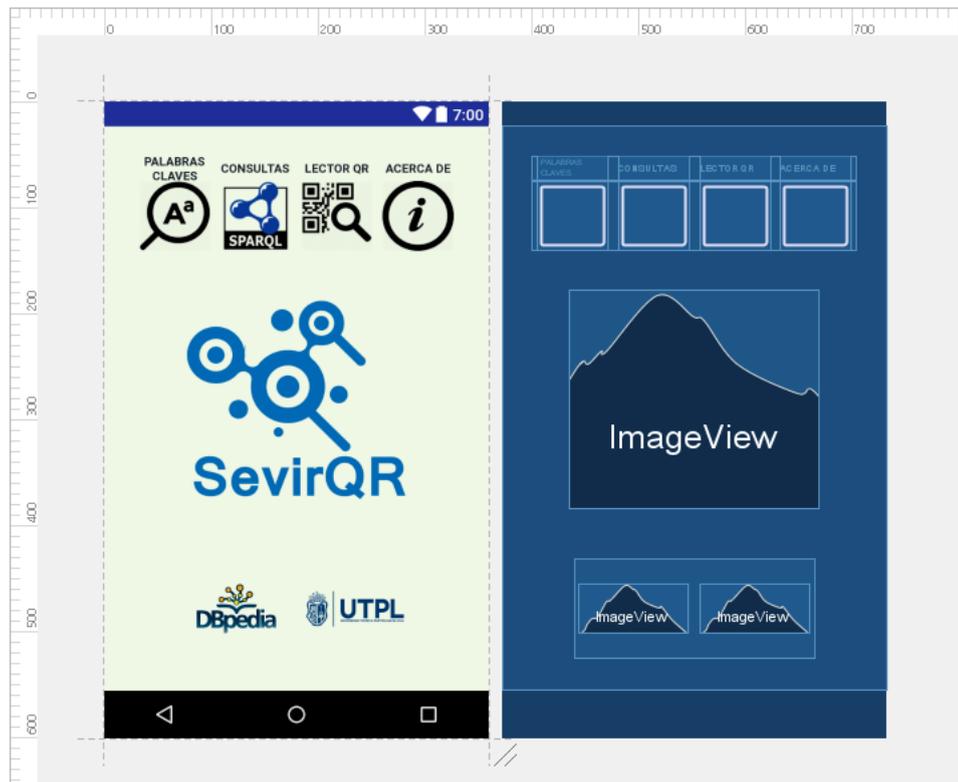


Figura 19. Activity_index

Fuente: Autor

⁹ Material Design. Disponible en: <https://material.io/>

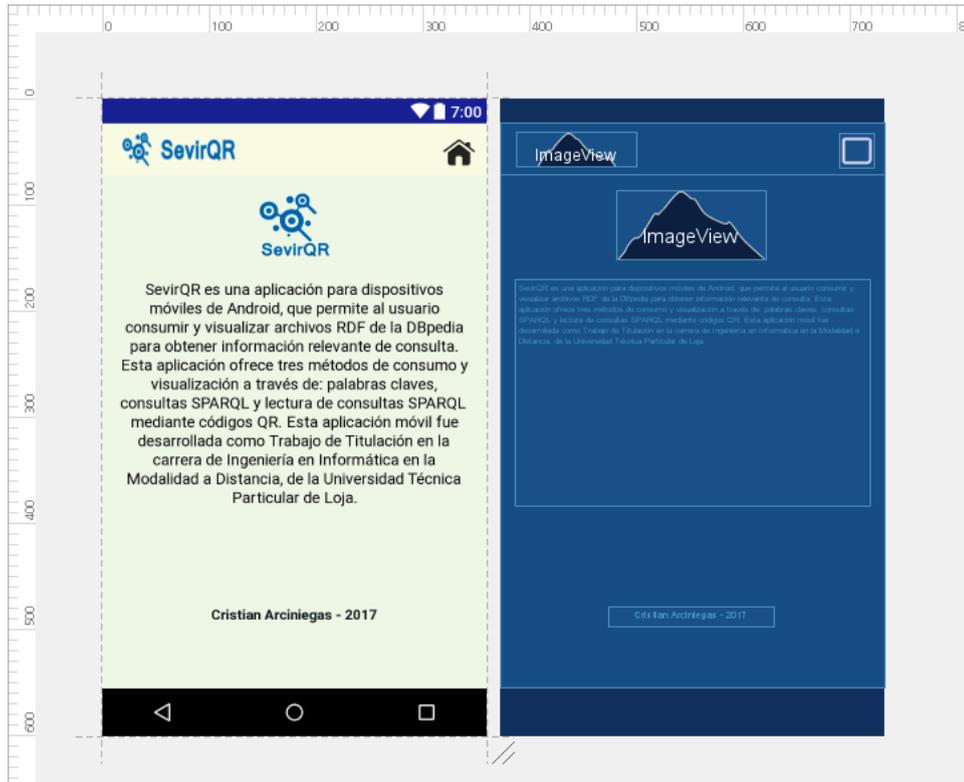


Figura 20. Activity_about

Fuente: Autor

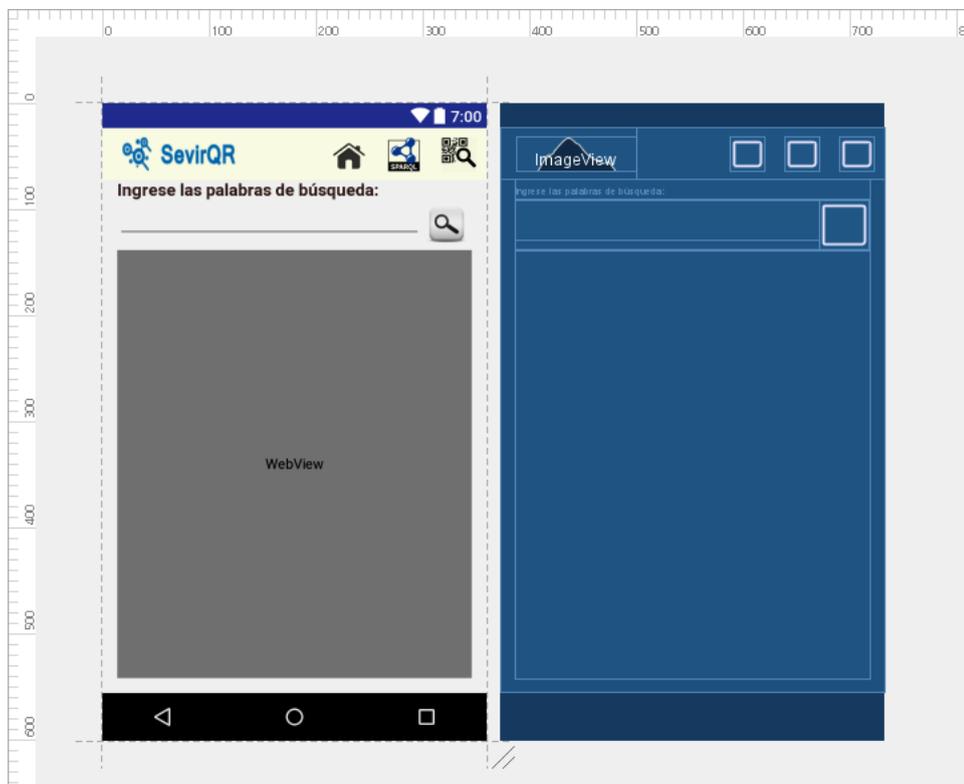


Figura 21. Activity_keywords

Fuente: Autor

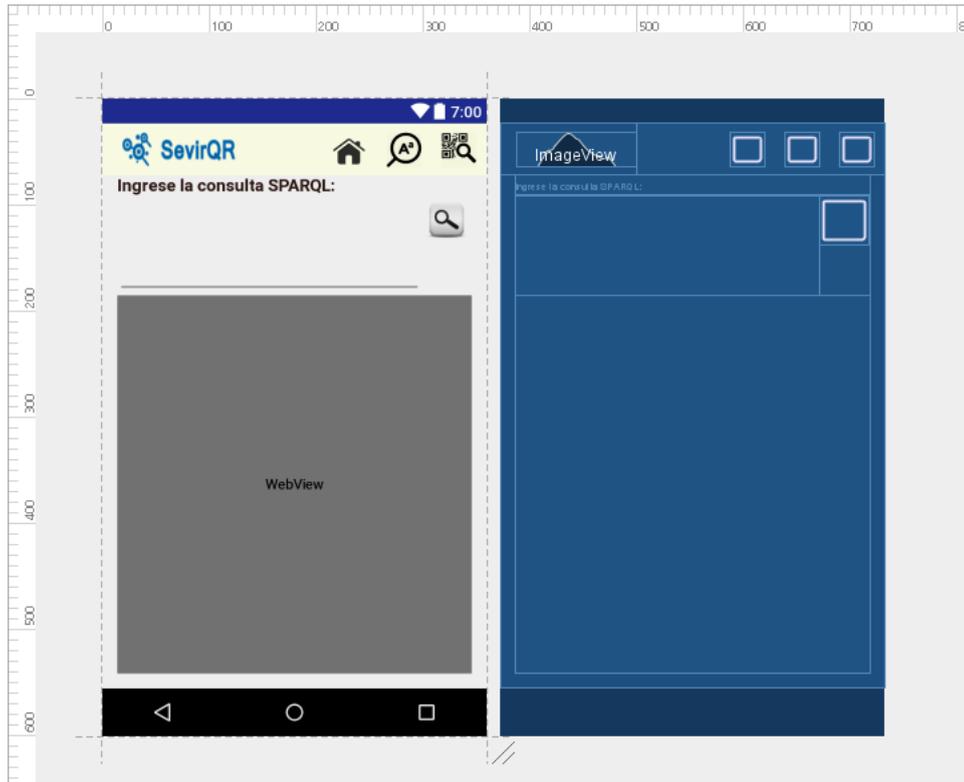


Figura 22. Activity_queries_sparql

Fuente: Autor

3.4.3. Sprint 3

Se estima completar el Sprint 3 en 28 horas de trabajo, sabiendo que se trabajarán 4 horas por día y 5 días a la semana, serán necesarios 7 días para poder completar todas las tareas de este Sprint. La tabla siguiente presenta el avance diario de cada una de las tareas.

Tabla 15. Avance diario del Sprint 3

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-03-30	4	28	Cristian A.	-	TR011	Pendiente
2017-03-31	4	24	Cristian A.	TR011	TR011	Terminada
2017-04-03	3	20	Cristian A.	TR011	TR012	Terminada
2017-04-04	2	16	Cristian A.	TR012	TR013	Pendiente
2017-04-05	2	12	Cristian A.	TR013	TR013	Terminada
2017-04-06	1	8	Cristian A.	TR013	TR014	Pendiente
2017-04-07	1	4	Cristian A.	TR014	TR014	Terminada
2017-04-07	0	-	Cristian A.	TR014	-	Fin Sprint 3

Fuente: Autor

3.4.3.1. Gráfico Burndown Chart

A continuación se muestra el gráfico Burndown Chart que muestra el cumplimiento de tareas del Sprint 3.

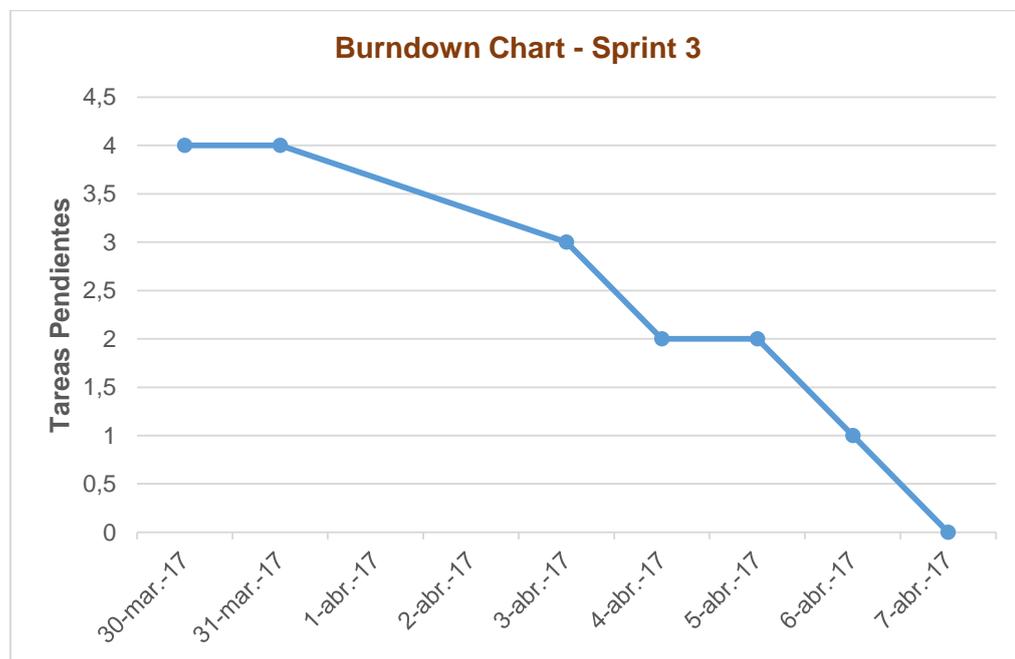


Figura 23. Burndown Chart - Sprint 3

Fuente: Autor

3.4.3.2. Elaboración del incremento

En este Sprint se completó el desarrollo de las interfaces, se implementaron las funcionalidades de los botones y del menú de la aplicación. Además, se mejoró la estructura y organización del diseño de las actividades a través del archivo *styles.xml* y *strings.xml*.

A continuación se describen los componentes desarrollados en este Sprint.

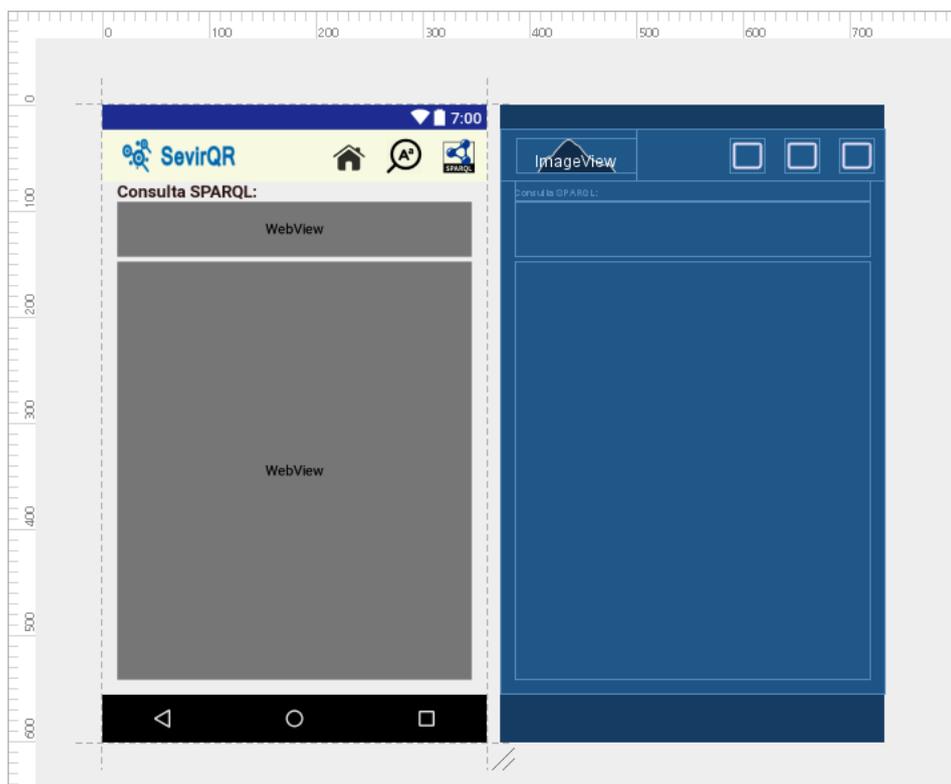


Figura 24. Activity_qr_code

Fuente: Autor

```

<resources>
  <string name="Nombre_App">SevirQR</string>
  <string name="Titulo_keyWordsActivity">Ingrese las palabras de búsqueda:</string>
  <string name="Titulo_queriesSPARQLActivity">Ingrese la consulta SPARQL:</string>
  <string name="Titulo_qrCodeActivity">Consulta SPARQL:</string>
  <string name="TextoAbout">SevirQR es una aplicación para dispositivos móviles de Android,
  que permite al usuario consumir y visualizar archivos RDF de la DBpedia para obtener información
  relevante de consulta. Esta aplicación ofrece tres métodos de consumo y visualización a través de:
  ...</string>
  <string name="Autor">Cristian Arciniegas - 2017</string>
  <string name="Menu_index">Inicio</string>
  <string name="Menu_opc1">Búsqueda por palabras claves</string>
  <string name="Menu_opc2">Consultas SPARQL-DBpedia</string>
  <string name="Menu_opc3">Lector de consultas SPARQL-QRCode</string>
  <string name="BtnkeyWords">PALABRAS CLAVES</string>
  <string name="BtnQueriesSPARQL">CONSULTAS</string>
  <string name="BtnQRCode">LECTOR QR</string>
  <string name="BtnAbout">ACERCA DE</string>
  <string name="img_appLogo">AppLogo</string>
  <string name="img_DBpedia">DBpedia</string>
  <string name="img_UTPL">UTPL</string>
</resources>

```

Figura 25. Archivo strings.xml

Fuente: Autor

```

<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <item name="colorPrimary">#64b5f6</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">#607d8b</item>
  </style>

  <!-- Títulos de la app. -->
  <style name="Títulos">
    <item name="android:textColor">#3e2723</item>
    <item name="android:textSize">17sp</item>
    <item name="android:textStyle">bold</item>
  </style>

  <!-- Textos de la app. -->
  <style name="TextoAbout">
    <item name="android:textColor">#212121</item>
    <item name="android:textSize">16sp</item>
    <item name="android:textStyle">normal</item>
  </style>

</resources>

```

Figura 26. Fragmento del archivo styles.xml

Fuente: Autor

3.4.4. Sprint 4

El tiempo estimado para completar el Sprint 4 será de 32 horas. Se trabajarán 4 horas por día y 5 días a la semana, por lo que serán necesarios 8 días para poder completar todas las tareas. La tabla siguiente presenta el avance diario de cada una de las tareas que corresponden a este Sprint.

Tabla 16. Avance diario del Sprint 4

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-04-10	4	32	Cristian A.	-	TR015	Pendiente
2017-04-11	4	28	Cristian A.	TR015	TR015	Terminada
2017-04-12	3	24	Cristian A.	TR015	TR016	Pendiente
2017-04-13	3	20	Cristian A.	TR016	TR016	Terminada
2017-04-14	2	16	Cristian A.	TR016	TR017	Pendiente
2017-04-17	2	12	Cristian A.	TR017	TR017	Terminada
2017-04-18	1	8	Cristian A.	TR017	TR018	Pendiente
2017-04-19	1	4	Cristian A.	TR018	TR018	Terminada
2017-04-19	0	-	Cristian A.	TR018	-	Fin Sprint 4

Fuente: Autor

3.4.4.1. Gráfico Burndown Chart

El gráfico Burndown Chart muestra el cumplimiento de las tareas del Sprint 4.

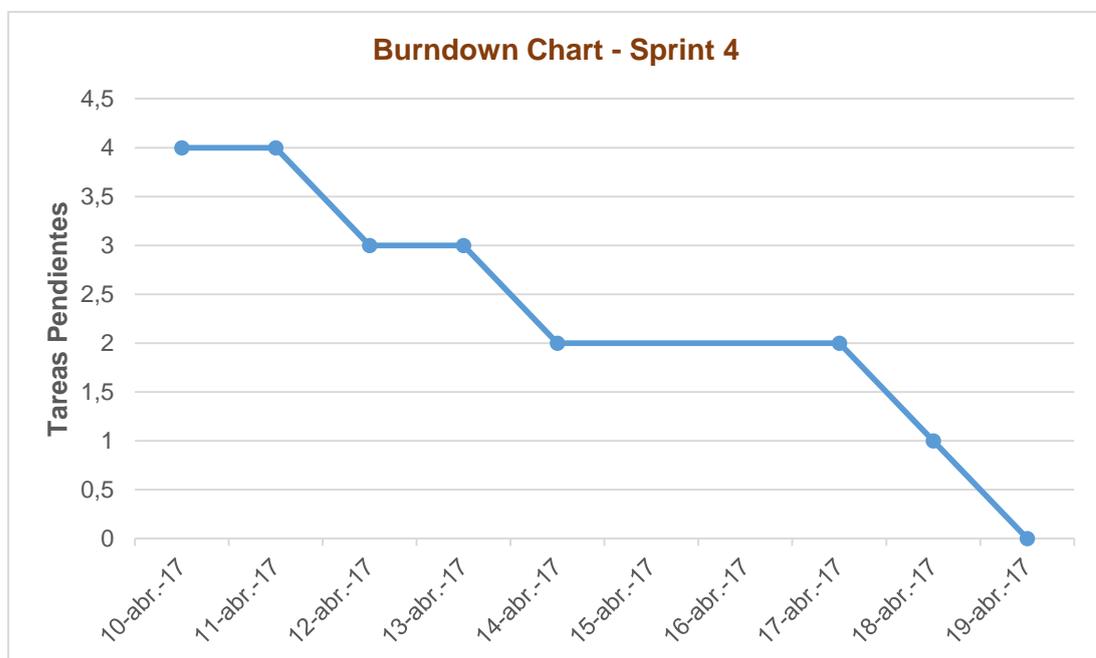


Figura 27. Burndown Chart - Sprint 4

Fuente: Autor

3.4.4.2. Elaboración del incremento

Este Sprint implementa el mecanismo de búsqueda mediante palabras claves a través de la API Lookup DBpedia. Esto se lo realiza en la clase *keyWordsActivity.java* del proyecto SevirQR de Android Studio. En este proceso de búsqueda, una vez que las palabras son ingresadas por el usuario se obtienen los datos mediante un archivo de respuesta JSON y estos son presentados utilizando el visualizador que se diseñó para este mecanismo.

Los componentes desarrollados para este mecanismo de búsqueda son presentados a continuación.

```
public class ObtenerDatosAPI extends AsyncTask<String, Integer, String[]>{
    @Override
    protected String[] doInBackground(String... params) {
        //URL de consulta a la API Lookup-DBpedia
        String
        cadenaurl="http://lookup.dbpedia.org/api/search.asmx/KeywordSearch?QueryClass=&MaxHits=
4&QueryString=";
        cadenaurl = cadenaurl + params[0];

        URL url = null;
        try {
            //Realiza conexión mediante HTTP
            url = new URL(cadenaurl);
            HttpURLConnection conexion = (HttpURLConnection)url.openConnection();
            conexion.setRequestProperty("Accept", "application/json");
            StringBuilder result = new StringBuilder();
```

Figura 28. Implementación del servicio Lookup DBpedia

Fuente: Autor

```
//Crea el Objeto JSON para acceder a los atributos del Array JSON obtenido
JSONObject stringJSON = new JSONObject(result.toString());
JSONArray resultJSON = stringJSON.getJSONArray("results");

//Obtiene los posibles resultados
String[] datos = new String[10];
if (resultJSON.length() > 0) {
    datos[0] = resultJSON.getJSONObject(0).getString("label");
    datos[1] = resultJSON.getJSONObject(0).getString("description");
    datos[2] = resultJSON.getJSONObject(0).getString("uri");
```

Figura 29. Recuperación de datos desde el archivo JSON

Fuente: Autor

```

public String ViewResults(String t1, String d1, String u1, String t2, String d2, String u2, String t3,
String d3, String u3){
    //Genera visualizador HTML de resultados
    ViewResult = "<!DOCTYPE html>" +
        "<html><head>" + ObtenerCSS() + "</head>" +
        "<body>\n" +
        "<table>" +
        "<tr>\n" +
        "<th><h3>RESULTADOS</h3></th>\n" +
        "</tr>\n" +
        "<tr>\n" +
        "<td>" +
        "<h3>Etiqueta:</h3>\n" +
        "<p>" + t1 + "\n" +
        "<h3>Descripción:</h3>\n" +
        "<p>" + d1 + "\n" +
        "<h3>URI:</h3>" +
        "<a href=\"" + u1 + "\">" + u1 + "</a>" +
        "</td>\n" +
        "</tr>\n" +
        "<tr>\n" +
        "<td>" +

```

Figura 30. Implementación del visualizador Lookup DBpedia

Fuente: Autor

3.4.5. Sprint 5

Se estima completar el Sprint 5 en 28 horas de trabajo, sabiendo que se trabajarán 4 horas por día y 5 días a la semana, serán necesarios 7 días para poder completar todas las tareas de este Sprint. La tabla siguiente presenta el avance diario de cada una de las tareas.

Tabla 17. Avance diario del Sprint 5

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-04-20	4	28	Cristian A.	-	TR019	Pendiente
2017-04-21	4	24	Cristian A.	TR019	TR019	Terminada
2017-04-24	3	20	Cristian A.	TR019	TR020	Terminada
2017-04-25	2	16	Cristian A.	TR020	TR021	Pendiente
2017-04-26	2	12	Cristian A.	TR021	TR021	Terminada
2017-04-27	1	8	Cristian A.	TR021	TR022	Pendiente
2017-04-28	1	4	Cristian A.	TR022	TR022	Terminada
2017-04-28	0	-	Cristian A.	TR022	-	Fin Sprint 5

Fuente: Autor

3.4.5.1. Gráfico Burndown Chart

A continuación se muestra el gráfico Burndown Chart que muestra el cumplimiento de las tareas del Sprint 5.

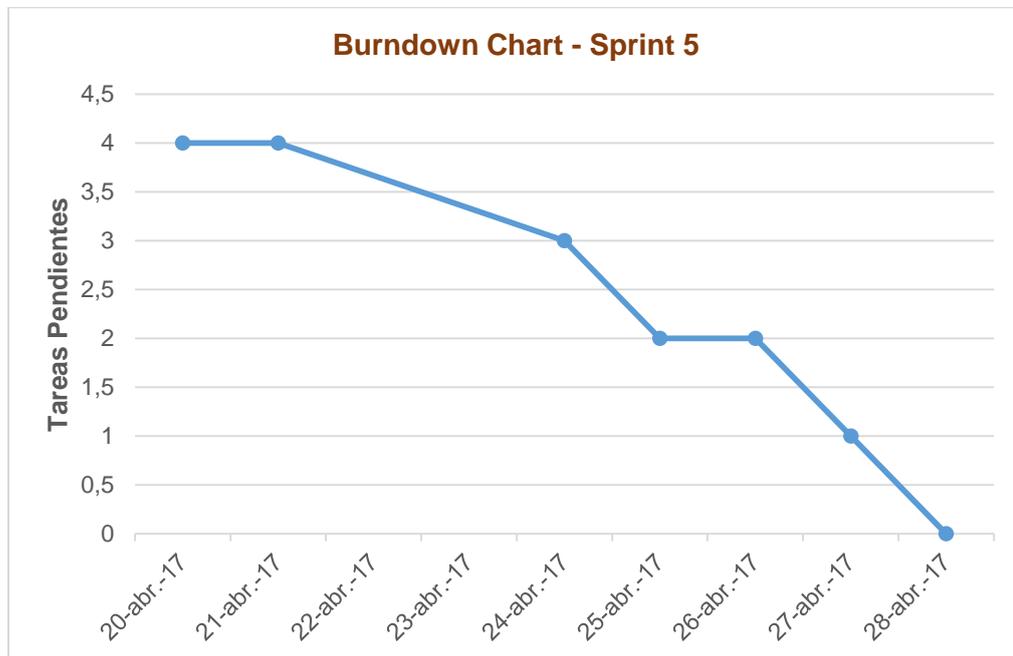


Figura 31. Burndown Chart - Sprint 5

Fuente: Autor

3.4.5.2. Elaboración del incremento

Este Sprint implementa el mecanismo de búsqueda a través del servicio SPARQL Endpoint, el mismo que permitirá al usuario digitar consultas SPARQL para obtener resultados directos de la DBpedia. Esto se lo realiza en la clase *queriesSPARQLActivity.java*, donde se implementa el servicio de consulta y cuyos resultados son obtenidos a través de un archivo HTML que devuelve todo el contenido en este formato. La consulta SPARQL deberá estar correctamente escrita para que el mecanismo pueda retornar resultados.

Los componentes que fueron desarrollados en este Sprint son descritos a continuación.

```
public class ObtenerQueryesDBpedia extends AsyncTask<String, Integer, String> {
    @Override
    protected String doInBackground(String... params) {
        //URL de consulta a DBpedia Endpoint
        String cadenaurl="http://dbpedia.org/sparql?query=" + params[0] + "&format=html";

        URL url = null;
        try {
            //Realiza conexión mediante HTTP
            url = new URL(cadenaurl);
            HttpURLConnection conexion = (HttpURLConnection)url.openConnection();
            StringBuilder result = new StringBuilder();
```

Figura 32. Implementación del servicio SPARQL Endpoint

Fuente: Autor

```
//Obtiene respuesta de conexión
if (conexion.getResponseCode() == HttpURLConnection.HTTP_OK) {
    //Prepara la entrada de Stream de datos
    InputStream in = new BufferedInputStream(conexion.getInputStream());
    BufferedReader rd = new BufferedReader(new InputStreamReader(in));

    //Pasa el Stream de datos al StringBuilder para leerlo
    String linea;
    while ((linea = rd.readLine()) != null) {
        result.append(linea);
    }
    DBpediaData = result.toString();
```

Figura 33. Recuperación de datos en formato HTML

Fuente: Autor

```
public String ViewResults(String resultados){
    //Genera visualizador HTML de resultados
    ViewResult = "<!DOCTYPE html>" +
        "<html><head>" + ObtenerCSS() + "</head>" +
        "<body>\n" +
        resultados +
        "</body>\n" +
        "</html> ";
    return ViewResult;
}
```

Figura 34. Implementación del visualizador SPARQL Endpoint

Fuente: Autor

3.4.6. Sprint 6

El tiempo estimado para completar el Sprint 6 será de 48 horas. Se trabajarán 4 horas por día y 5 días a la semana, por lo que serán necesarios 12 días para poder completar todas las tareas. La tabla siguiente presenta el avance diario de cada una de las tareas que corresponden a este Sprint.

Tabla 18. Avance diario del Sprint 6

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-05-01	6	48	Cristian A.	-	TR023	Pendiente
2017-05-02	6	44	Cristian A.	TR023	TR023	Terminada
2017-05-03	5	40	Cristian A.	TR023	TR024	Pendiente
2017-05-04	5	36	Cristian A.	TR024	TR024	Terminada
2017-05-05	4	32	Cristian A.	TR024	TR025	Pendiente
2017-05-08	4	28	Cristian A.	TR025	TR025	Terminada
2017-05-09	3	24	Cristian A.	TR025	TR026	Pendiente
2017-05-10	3	20	Cristian A.	TR026	TR026	Terminada
2017-05-11	2	16	Cristian A.	TR026	TR027	Pendiente
2017-05-12	2	12	Cristian A.	TR027	TR027	Terminada
2017-05-15	1	8	Cristian A.	TR027	TR028	Pendiente
2017-05-16	1	4	Cristian A.	TR028	TR028	Terminada
2017-05-16	0	-	Cristian A.	TR028	-	Fin Sprint 6

Fuente: Autor

3.4.6.1. Grafico Burndown Chart

El gráfico Burndown Chart muestra el cumplimiento de las tareas del Sprint 6.

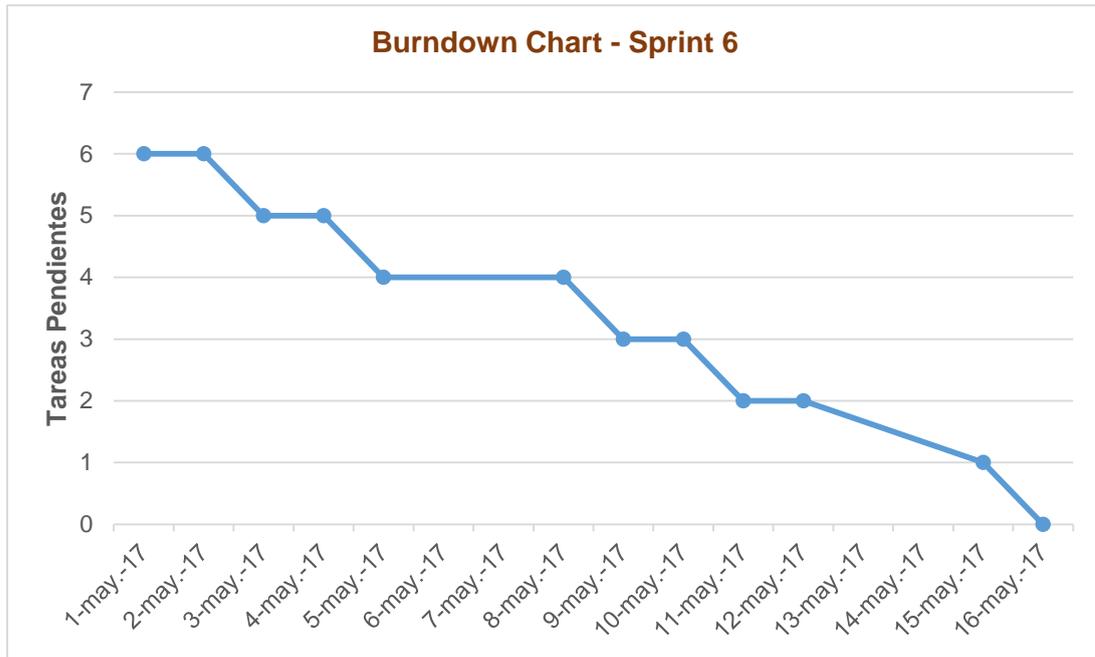


Figura 35. Burndown Chart - Sprint 6

Fuente: Autor

3.4.6.2. Elaboración del incremento

El Sprint 6 permitió implementar el mecanismo de lectura y búsqueda de información mediante la lectura de códigos QR a través del servicio SPARQL Endpoint. Esto fue implementado en la clase *qrCodeActivity.java*, donde se realiza la conexión con el repositorio de DBpedia y los datos son obtenidos a través de un archivo cuya serialización viene dada en el formato RDF/XML. Estos datos son leídos y presentados gracias al visualizador construido específicamente para este mecanismo de consulta.

Adicionalmente, fue necesario la adaptación de un proceso para la lectura de los códigos QR, que permite a la aplicación móvil leer las consultas SPARQL codificadas utilizando la cámara del dispositivo, para así poder ejecutarlas automáticamente. Este mecanismo fue implementado en la clase *BarcodeCaptureActivity.java*.

Los componentes desarrollados en este Sprint son presentados a continuación.

```

public final class BarcodeCaptureActivity extends AppCompatActivity
    implements BarcodeTracker.BarcodeGraphicTrackerCallback {

    private static final String TAG = "Barcode-reader";
    // Intent request code to handle updating play services if needed.
    private static final int RC_HANDLE_GMS = 9001;
    // Permission request codes need to be < 256
    private static final int RC_HANDLE_CAMERA_PERM = 2;
    // Constants used to pass extra data in the intent
    public static final String BarcodeObject = "Barcode";
    private CameraSource mCameraSource;
    private CameraSourcePreview mPreview;

```

Figura 36. Implementación de lector de códigos QR

Fuente: Autor

```

//Realiza lectura de QRCode y muestra consulta SPARQL
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == BARCODE_READER_REQUEST_CODE) {
        if (resultCode == CommonStatusCodes.SUCCESS) {
            if (data != null) {
                Barcode barcode = data.getParcelableExtra(BarcodeCaptureActivity.BarcodeObject);
                Point[] p = barcode.cornerPoints;
                QRrecuperado = barcode.displayValue;
                WVConsultaQR.loadData(ViewConsultaSPARQL(TextUtils.htmlEncode(QRrecuperado)),
"text/html; charset=utf-8", null);
            } else QRrecuperado = "Error de Captura";
        } else Log.e(LOG_TAG, String.format("Error de Lectura",
            CommonStatusCodes.getStatusCodeString(resultCode)));
    } else super.onActivityResult(requestCode, resultCode, data);

```

Figura 37. Lectura de código QR y obtención de la consulta SPARQL

Fuente: Autor

```

public class ConsultarQRDBpedia extends AsyncTask<String, Integer, String[]> {
    @Override
    protected String[] doInBackground(String... params) {
        //URL de consulta a DBpedia Endpoint
        String cadenauri="http://dbpedia.org/sparql?query=" + params[0] + "&format=rdf";

        URL url = null;
        try {
            url = new URL(cadenauri);
            HttpURLConnection conexion = (HttpURLConnection)url.openConnection();

```

Figura 38. Implementación del servicio SPARQL Endpoint

Fuente: Autor

```

//Obtiene respuesta de conexión
if (conexion.getResponseCode() == HttpURLConnection.HTTP_OK) {
    //Prepara la entrada del Stream de datos y lee mediante DocumentBuilder
    InputStream in = new BufferedInputStream(conexion.getInputStream());
    DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
    DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
    Document doc = dBuilder.parse(in);

    //Crea el objeto XML para leer los nodos de la estructura RDF
    Element element = doc.getDocumentElement();
    element.normalize();
    NodeList item_desc, item_img, item_etiq, item_familia, item_orden, item_link;
    Node nodoAbs = null, nodoImg = null, nodoLabel = null, nodoLink = null;

```

Figura 39. Recuperación de datos desde el archivo RDF/XML

Fuente: Autor

```

public String ViewDBpedia(String label, String image, String description, String link){
    //Genera el visualizador HTML de los resultados obtenidos
    visualizadorDBpedia = "<!DOCTYPE html>\n" +
        "<html><head>" + CSS_View() + "</head>\n" +
        "<body>\n" +
        "\n" +
        "<h3>ETIQUETA:</h3>\n" +
        "<p>" + label + "\n" +
        "<h3>IMAGEN:</h3>\n" +
        "<p><img src=\"" + image + "\" +
        "\" alt=\"DepictionDBpedia\" style=\"width:170px;height:170px;\">\n" +
        "<h3>DESCRIPCIÓN:</h3>\n" +
        "<p>" + description + "\n" +
        "<h3>LINK EXTERNO:</h3>\n" +
        "<a href=\"" + link + "\">" + link + "</a>" +
        "</body>\n" +
        "</html> ";
    return visualizadorDBpedia;
}

```

Figura 40. Implementación del visualizador SPARQL Endpoint QR

Fuente: Autor

3.4.7. Sprint 7

El tiempo estimado para completar el Sprint 7 será de 48 horas. Se trabajarán 4 horas por día y 5 días a la semana, por lo que serán necesarios 12 días para poder completar todas las tareas. La tabla siguiente presenta el avance diario de cada una de las tareas que corresponden a este Sprint.

Tabla 19. Avance diario del Sprint 7

Fecha	Tareas Pendientes	Horas Pendientes	Responsable	Actividad Anterior	Actividad Actual	Estado Actual
2017-05-17	6	48	Cristian A.	-	TR029	Pendiente
2017-05-18	6	44	Cristian A.	TR029	TR029	Terminada
2017-05-19	5	40	Cristian A.	TR029	TR030	Pendiente
2017-05-22	5	36	Cristian A.	TR030	TR030	Terminada
2017-05-23	4	32	Cristian A.	TR030	TR031	Pendiente
2017-05-24	4	28	Cristian A.	TR031	TR031	Terminada
2017-05-25	3	24	Cristian A.	TR031	TR032	Pendiente
2017-05-29	3	20	Cristian A.	TR032	TR032	Terminada
2017-05-30	2	16	Cristian A.	TR032	TR033	Pendiente
2017-05-31	2	12	Cristian A.	TR033	TR033	Terminada
2017-06-01	1	8	Cristian A.	TR033	TR034	Pendiente
2017-06-02	1	4	Cristian A.	TR034	TR034	Terminada
2017-06-02	0	-	Cristian A.	TR034	-	Fin Sprint 7

Fuente: Autor

3.4.7.1. Gráfico Burndown Chart

A continuación se muestra el gráfico Burndown Chart que muestra el cumplimiento de las tareas del Sprint 7.

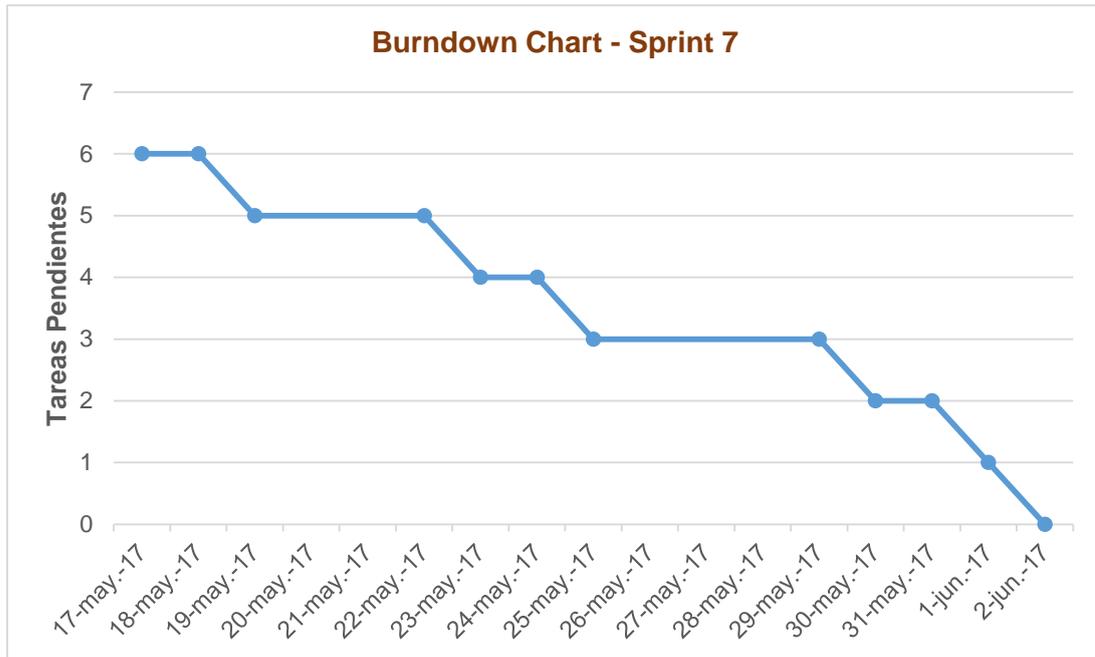


Figura 41. Burndown Chart - Sprint 7

Fuente: Autor

3.4.7.2. Elaboración del incremento

En este último Sprint se desarrolla el sitio web que permite la generación de los códigos QR, los mismos que son utilizados por la aplicación móvil para la lectura de las consultas SPARQL. Este sitio web también brinda información sobre el proyecto y los contactos de los respectivos involucrados.

El sitio web fue desarrollado utilizando como base HTML5 y CSS3 para diseñar el contenido y permitir la correcta visualización del sitio en dispositivos móviles. Para la generación de los códigos QR fue utilizada una función desarrollada en JavaScript, la cual genera el código QR automáticamente una vez que la consulta es digitada en el campo correspondiente.

El sitio web y los componentes desarrollados se describen a continuación.



Figura 42. Página index.html

Fuente: Autor



Figura 43. Página qrcode.html

Fuente: Autor



Figura 44. Página contactos.html

Fuente: Autor

```
//-----
// QRCode
//-----

function QRCode(typeNumber, errorCorrectLevel) {
    this.typeNumber = typeNumber;
    this.errorCorrectLevel = errorCorrectLevel;
    this.modules = null;
    this.moduleCount = 0;
    this.dataCache = null;
    this.dataList = new Array();
}

QRCode.prototype = {
    addData : function(data) {
        var newData = new QR8bitByte(data);
        this.dataList.push(newData);
        this.dataCache = null;
    },
}
```

Figura 45. Fragmento de la función QRCode en JavaScript

Fuente: Autor

```

*{
  margin: 0;
  padding: 0;
}

body{
  background: #f2f2f2;
  font-family: 'Source Sans Pro', sans-serif;
}
header{
  background: #e5e7e9;
  padding: 5px;
  text-shadow: 2px 2px 4px #000;
}

```

Figura 46. Fragmento del archivo estilos.ccs

Fuente: Autor

```

<section id="title">
  <h2>Generación de códigos QR</h2>
</section>
  <main>
    <section id="contenedor">
      <div id="mensaje">
        <h3>Ingrese la consulta SPARQL:</h3>
        <form name="QRform" id="QRform">
          <textarea name="textField" rows="8" cols="100" onkeyup='updateQRCode(this.value)'
onclick="this.focus();this.select();">Consulta SPARQL</textarea>
        </form>
      </div>
    </section>

    <!-- Aquí es donde aparecerá el QRCode. -->
    <div id="qrcodeg"></div>
    <script type="text/javascript">
      function updateQRCode(text) {
        var element = document.getElementById("qrcodeg");
        var bodyElement = document.body;
        if (element.lastChild)
          element.replaceChild(showQRCode(text), element.lastChild);
        else
          element.appendChild(showQRCode(text));
      }
      updateQRCode('Consulta SPARQL');
    </script>

```

Figura 47. Implementación del generador de códigos QR en el archivo qrcode.html

Fuente: Autor

CAPITULO IV:
Pruebas e Implementación

4.1. Introducción

Este capítulo detalla el desarrollo del plan de pruebas realizado para verificar el funcionamiento tanto de la aplicación móvil como del sitio web, y también describe los mecanismos de implementación utilizados. Es importante mencionar que las pruebas fueron realizadas al concluir los respectivos incrementos en cada Sprint y basados en los siguientes tipos de pruebas: aceptación, carga, usabilidad y accesibilidad. El Anexo 2 documenta el plan de pruebas.

4.2. Sprint Review – Sprint Retrospective

Durante la ejecución de los Sprints, la metodología SCRUM demanda la realización de ciertas reuniones. Para este trabajo, se priorizaron las reuniones denominadas *Sprint Review* o reuniones de revisión del Sprint y *Sprint Retrospective* o reunión de retrospectiva del Sprint. Las reuniones de revisión del Sprint permitieron presentar los incrementos del producto entregable y así poder verificar y analizar su funcionamiento, mientras que las reuniones de retrospectiva ayudaron en la planificación para mejorar el avance y la calidad de los productos en cada Sprint.

A continuación se muestran las actividades realizadas en el Sprint Review en cada uno de los incrementos de los Sprints.

Tabla 20. Actividades de Sprint Review

Sprint	Fecha	Observaciones
Sprint 1	2017-03-17	Presentación de bocetos de diseño e imágenes de la aplicación móvil.
Sprint 2	2017-03-29	Presentación de interfaces de la aplicación móvil.
Sprint 3	2017-04-07	Presentación de funcionalidades de navegación y mejora en diseño y estilo de ventanas de la aplicación móvil.
Sprint 4	2017-04-18	Presentación del funcionamiento de la <i>activity_keywords</i> .
Sprint 5	2017-04-27	Presentación del funcionamiento de la <i>activity_queries_sparql</i> .
Sprint 6	2017-05-15	Presentación del funcionamiento de la <i>activity_qr_code</i> .
Sprint 7	2017-06-01	Presentación del funcionamiento del sitio web.

Fuente: Autor

Las siguientes actividades fueron realizadas en el Sprint Retrospective en cada uno de los Sprint.

Tabla 21. Actividades de Sprint Retrospective

Sprint	Lo que funcionó correctamente	Lo que se tuvo que mejorar
Sprint 1	El manejo y uso de Android.	Las imágenes y logos a utilizar en la aplicación móvil.
Sprint 2	Las herramientas de diseño de Android.	El diseño de interfaces y el contraste de los colores.
Sprint 3	La generación del código en Android.	El estilo y diseño de las imágenes.
Sprint 4	La implementación del mecanismo de búsqueda por palabras claves.	La visualización de los datos obtenidos.
Sprint 5	La implementación del mecanismo de ejecución de consultas SPARQL.	La validación de las consultas SPARQL y datos ingresados.
Sprint 6	La implementación del mecanismo de ejecución de consultas SPARQL a través de códigos QR.	Definir el tipo de código QR y su generación para que pueda ser leído por cualquier dispositivo móvil.
Sprint 7	Las herramientas de diseño web, HTML, CSS3 y JavaScript.	Las imágenes a utilizar para el sitio web, y la adaptabilidad del sitio web en dispositivos móviles.

Fuente: Autor

4.3. Pruebas de aceptación

Las pruebas de aceptación fueron realizadas al culminar los incrementos de Sprint donde fue posible probar una parte funcional terminada. Estas pruebas sirvieron para verificar el cumplimiento de los requerimientos planteados en el Product Backlog basado en la propuesta de solución.

A continuación se detallan las actividades de cada una de las pruebas de aceptación realizadas.

4.3.1. Prueba de aceptación 1

Las actividades realizadas en la prueba de aceptación 1 se describen a continuación.

Tabla 22. Prueba de aceptación 1

Prueba de aceptación 1			
ID: P1	Nombre: Búsqueda mediante palabras claves en un ambiente móvil.		
Descripción: Se probará el mecanismo de búsqueda mediante la digitación de palabras claves.			
Responsable: Cristian Arciniegas		Fecha: 2017-04-18	
Pre-Condiciones	1. La interfaz y funcionalidad de la activity_keywords deben estar completos. 2. El mecanismo de búsqueda por palabras debe estar implementado.		
Post-Condiciones	1. Visualizar resultados obtenidos de la búsqueda por palabras claves.		
Paso	Acción	Respuesta del Sistema	Resultado
1	Ingresar a la aplicación móvil.	Presenta pantalla inicial de menú.	OK
2	Ingresar las palabras o términos de búsqueda.	Si el campo de texto está vacío se presentará un mensaje de alerta. Si no existen resultados se presentará un mensaje de alerta.	OK
3	Validar conexión a internet.	Si no existe conexión de internet se presentará un mensaje de alerta. Caso contrario continúa al paso 4.	OK
4	Realizar búsqueda.	Presenta los resultados obtenidos.	OK

Fuente: Autor

En seguida, se presentan los resultados obtenidos en cada uno de los pasos realizados para esta prueba de aceptación.



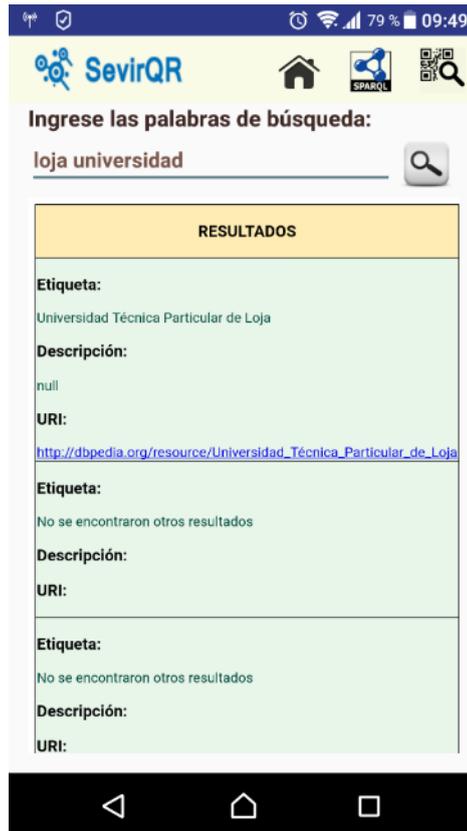


Figura 48. Resultados de la prueba de aceptación 1

Fuente: Autor

4.3.2. Prueba de aceptación 2

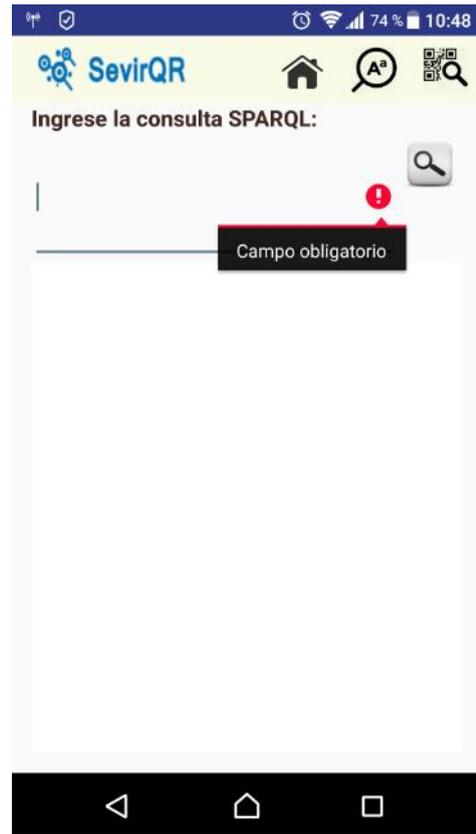
Las actividades realizadas en la prueba de aceptación 2 se detallan a continuación.

Tabla 23. Prueba de aceptación 2

Prueba de aceptación 2			
ID: P1	Nombre: Digitación y ejecución de consultas SPARQL en un ambiente móvil.		
Descripción: Se probará el mecanismo de búsqueda mediante la digitación de consultas SPARQL.			
Responsable: Cristian Arciniegas		Fecha: 2017-04-27	
Pre-Condiciones	<ol style="list-style-type: none"> 1. La interfaz y funcionalidad de la activity_queries_sparql deben estar completos. 2. El mecanismo de ejecución de consultas SPARQL debe estar implementado. 		
Post-Condiciones	<ol style="list-style-type: none"> 1. Visualizar resultados obtenidos de la ejecución de consultas SPARQL. 		
Paso	Acción	Respuesta del sistema	Resultado
1	Ingresar a la aplicación móvil.	Presenta pantalla inicial del menú.	OK
2	Digitar consulta SPARQL.	<p>Verifica que la consulta SPARQL esté correctamente escrita, de lo contrario presentará un mensaje de alerta.</p> <p>Si el campo de texto está vacío se presentará un mensaje de alerta.</p>	OK
3	Validar conexión a internet.	<p>Si no existe conexión de internet se presentará un mensaje de alerta.</p> <p>Caso contrario continúa al paso 4.</p>	OK
4	Ejecutar consulta SPARQL	Presenta los resultados obtenidos.	OK

Fuente: Autor

A continuación se presentan los resultados obtenidos en cada uno de los pasos realizados para esta prueba de aceptación.



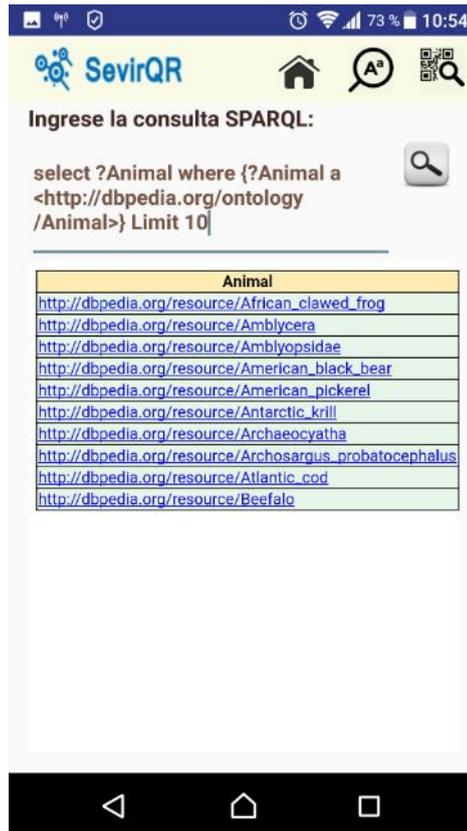


Figura 49. Resultados de la prueba de aceptación 2

Fuente: Autor

4.3.3. Prueba de aceptación 3

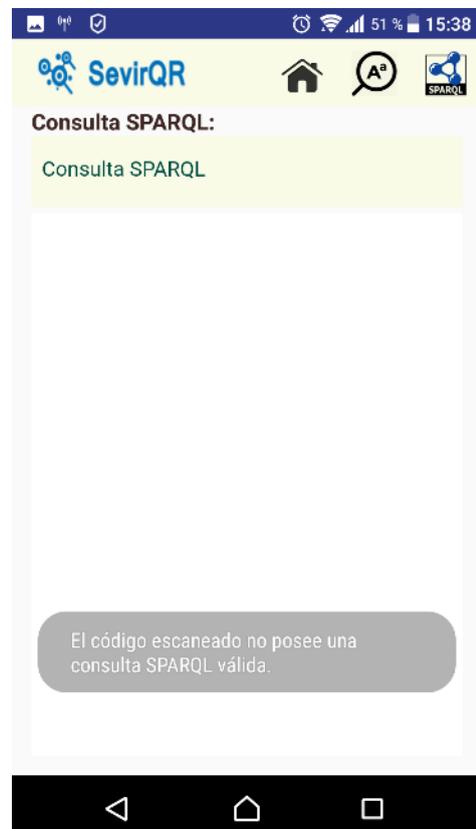
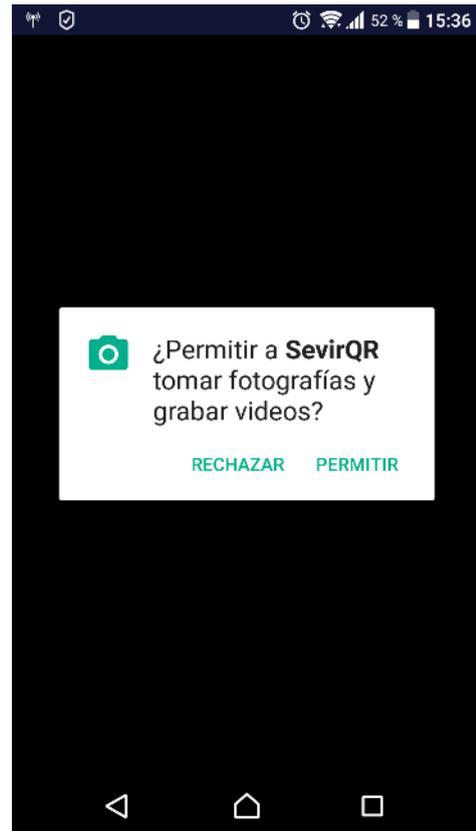
Las actividades realizadas en la prueba de aceptación 3 se describen a continuación.

Tabla 24. Prueba de aceptación 3

Prueba de aceptación 3			
ID: P1	Nombre: Lectura de códigos QR derivados a consultas SPARQL en un ambiente móvil.		
Descripción: Se probará el mecanismo de lectura de códigos QR para la ejecución de consultas SPARQL.			
Responsable: Cristian Arciniegas		Fecha: 2017-05-15	
Pre-Condiciones	<ol style="list-style-type: none"> 1. La interfaz y funcionalidad de la activity_qr_code deben estar completos. 2. La lectura de códigos QR y el mecanismo de ejecución de consultas SPARQL deben estar implementados. 		
Post-Condiciones	<ol style="list-style-type: none"> 1. Visualizar resultados obtenidos de la consulta SPARQL codificada en código QR. 		
Paso	Acción	Respuesta del sistema	Resultado
1	Ingresar a la aplicación móvil.	Presenta pantalla inicial del menú.	OK
2	Leer códigos QR codificados con consultas SPARQL.	Solicita permisos para usar la cámara del dispositivo como lector de los códigos QR. Verifica que la consulta SPARQL esté correctamente escrita, de lo contrario presentará un mensaje de alerta.	OK
3	Validar conexión a internet.	Si no existe conexión de internet se presentará un mensaje de alerta. Caso contrario continúa al paso 4.	OK
4	Ejecutar búsqueda usando consulta SPARQL decodificada.	Presenta los resultados obtenidos.	OK

Fuente: Autor

En seguida, se presentan los resultados obtenidos en cada uno de los pasos realizados para esta prueba de aceptación.



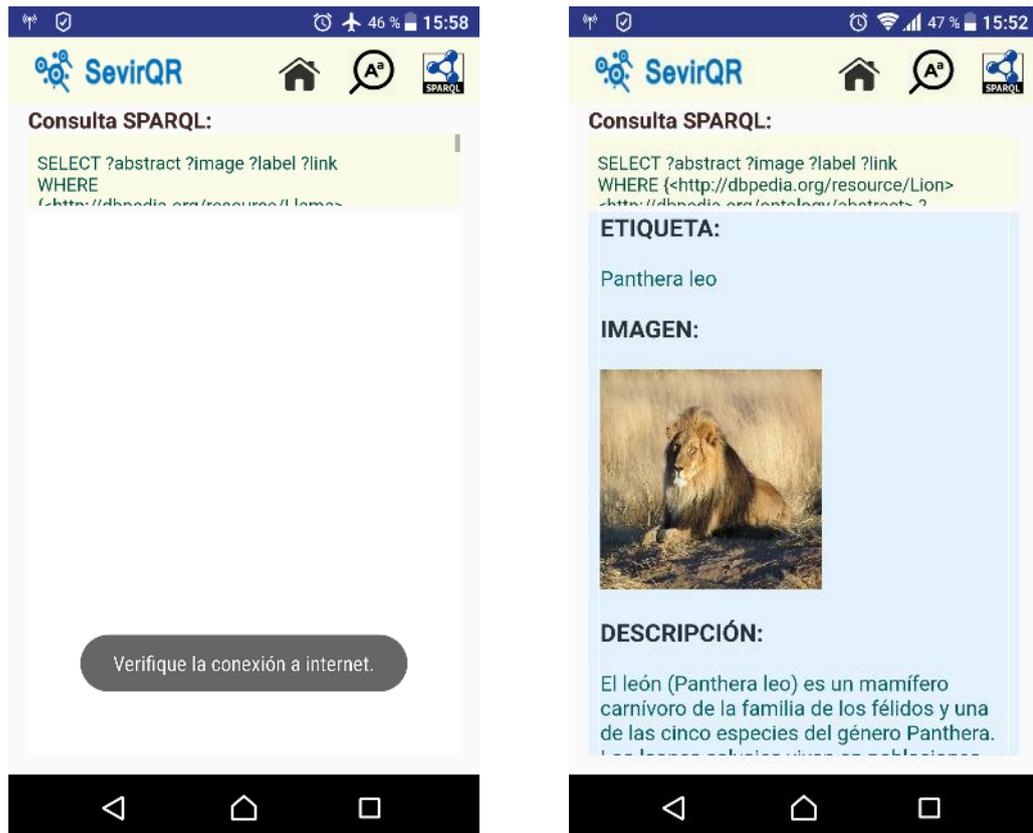


Figura 50. Resultados de la prueba de aceptación 3

Fuente: Autor

4.3.4. Prueba de aceptación 4

Las actividades realizadas en la prueba de aceptación 4 se detallan a continuación.

Tabla 25. Prueba de aceptación 4

Prueba de aceptación 4			
ID: P2	Nombre: Codificación de consultas SPARQL a través de códigos QR en un ambiente web.		
Descripción: Se probará la generación de los códigos QR que codificarán consultas SPARQL.			
Responsable: Cristian Arciniegas		Fecha: 2017-06-01	
Pre-Condiciones	<ol style="list-style-type: none"> 1. La interfaz y funcionalidad del sitio web deben estar completos. 2. La función que generará los códigos QR para la codificación de las consultas SPARQL debe estar implementada. 		
Post-Condiciones	<ol style="list-style-type: none"> 1. Visualizar y obtener mediante archivos de imagen los códigos QR generados y codificados con consultas SPARQL. 		
Paso	Acción	Respuesta del sistema	Resultado
1	Ingresar al sitio web.	Presenta la página <i>index.html</i> donde se encuentra el menú.	OK
2	Hacer clic en la opción "Generador QR".	Presenta la página <i>qrcode.html</i> .	OK
3	Digitar o pegar la consulta SPARQL a codificar.	La función JavaScript genera automáticamente la imagen del código QR. Existe un modelo que se indica para elaborar la consulta SPARQL.	OK
4	Obtener el código QR generado.	Al hacer clic derecho sobre la imagen del código QR generado, el navegador permitirá descargarlo en un archivo con el formato PNG.	OK

Fuente: Autor

A continuación se presentan los resultados obtenidos en cada uno de los pasos realizados para esta prueba de aceptación.





Figura 51. Resultados de la prueba de aceptación 4

Fuente: Autor

4.4. Pruebas de Carga

Estas pruebas permitieron observar el rendimiento de los aplicativos frente a la demanda de sus respectivos servicios. En el caso de la aplicación móvil, se utilizará la herramienta *Google Play Console*¹⁰ que permite realizar pruebas de rendimiento y recopilar información de la aplicación antes de su lanzamiento y publicación. Es importante indicar que el tiempo de respuesta de las solicitudes de búsqueda dependerá de la velocidad de ancho de banda, la tecnología del dispositivo móvil y la disponibilidad de los servicios Lookup DBpedia y DBpedia Endpoint según la arquitectura implementada en este proyecto.

A continuación presentamos los resultados de rendimiento de la aplicación móvil, que se obtuvieron en diferentes dispositivos.

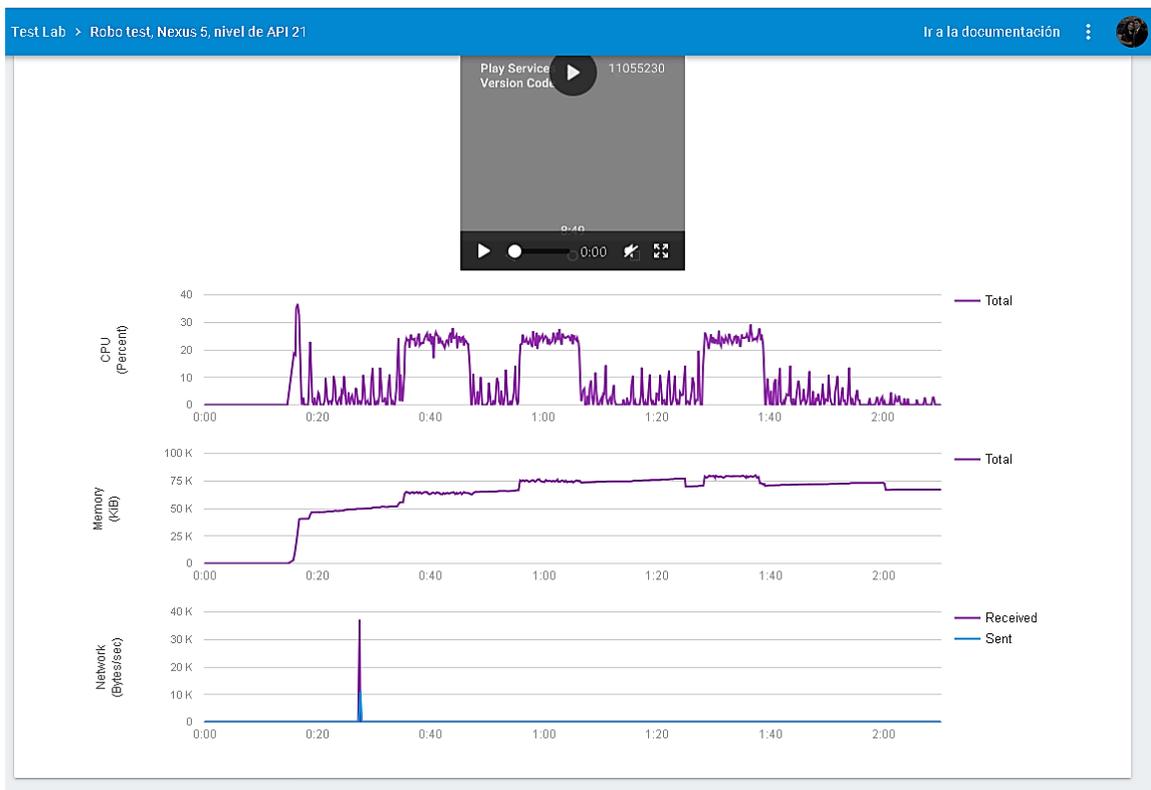


Figura 52. Test de rendimiento - Nexus 5 (Android 5.0)

Fuente: Autor

¹⁰ Google Play Console. Disponible en: <https://developer.android.com/distribute/console/index.html?hl=es-419>

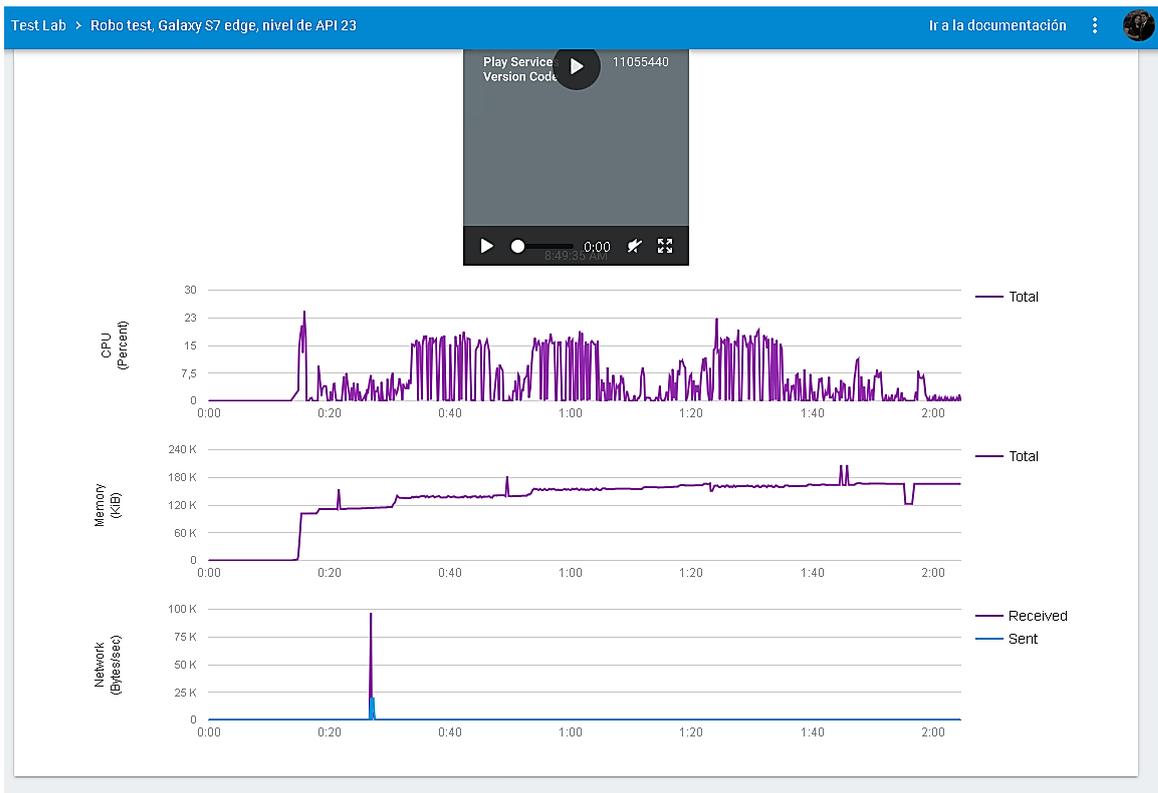


Figura 53. Test de rendimiento - Galaxy S7 Edge (Android 6.0)

Fuente: Autor



Figura 54. Test de rendimiento - Galaxy S7 (Android 7.0)

Fuente: Autor

Como se puede observar, la aplicación móvil no requiere de grandes capacidades de CPU (38%) o memoria (230 Kb), además se estima que no se requerirá de una gran cantidad de ancho de banda (98 Kbps).

Para la realización de las pruebas de carga y rendimiento del sitio web se utilizó la herramienta *Pingdom Tools*¹¹, la cual permite analizar el contenido de una página web, verificar su peso y el tiempo de carga de sus objetos. A continuación se detallan los resultados obtenidos.

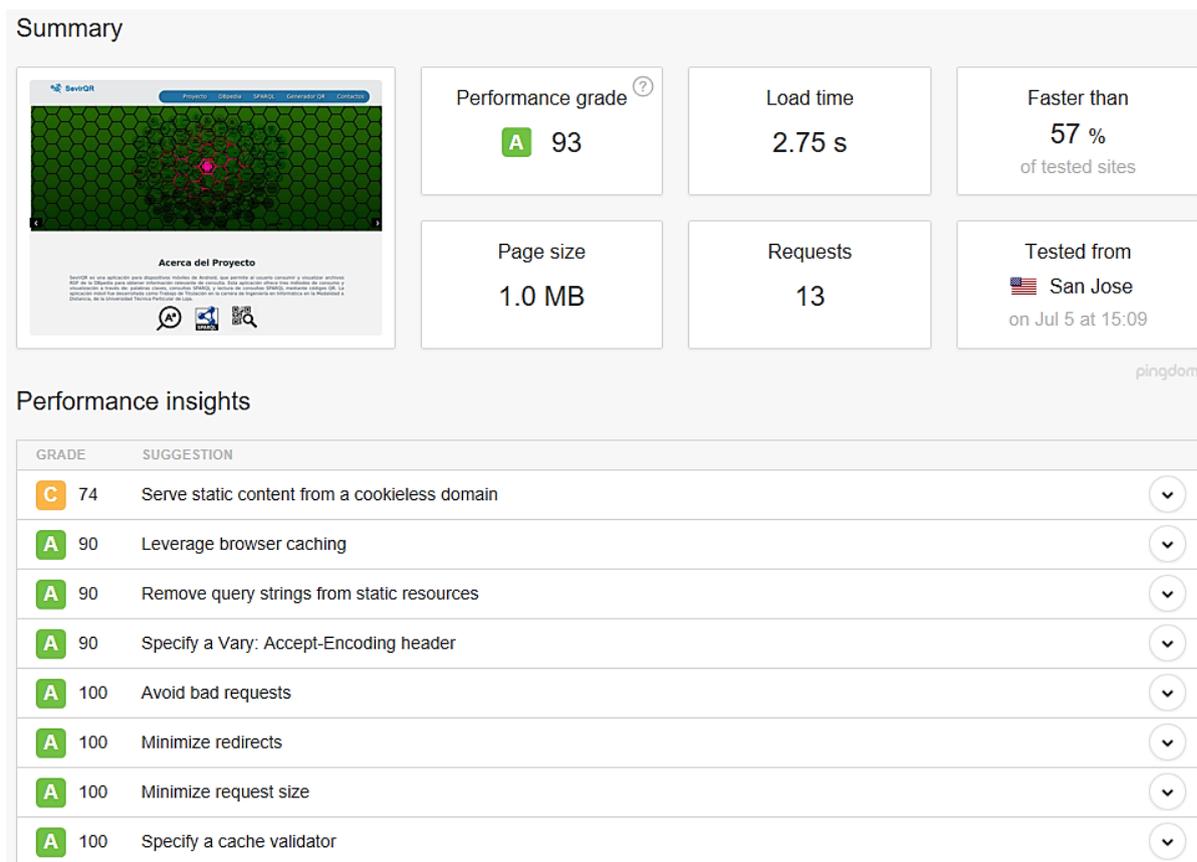


Figura 55. Resultados de carga del sitio web

Fuente: Autor

¹¹ Pingdom Tools. Disponible en: <https://tools.pingdom.com/>

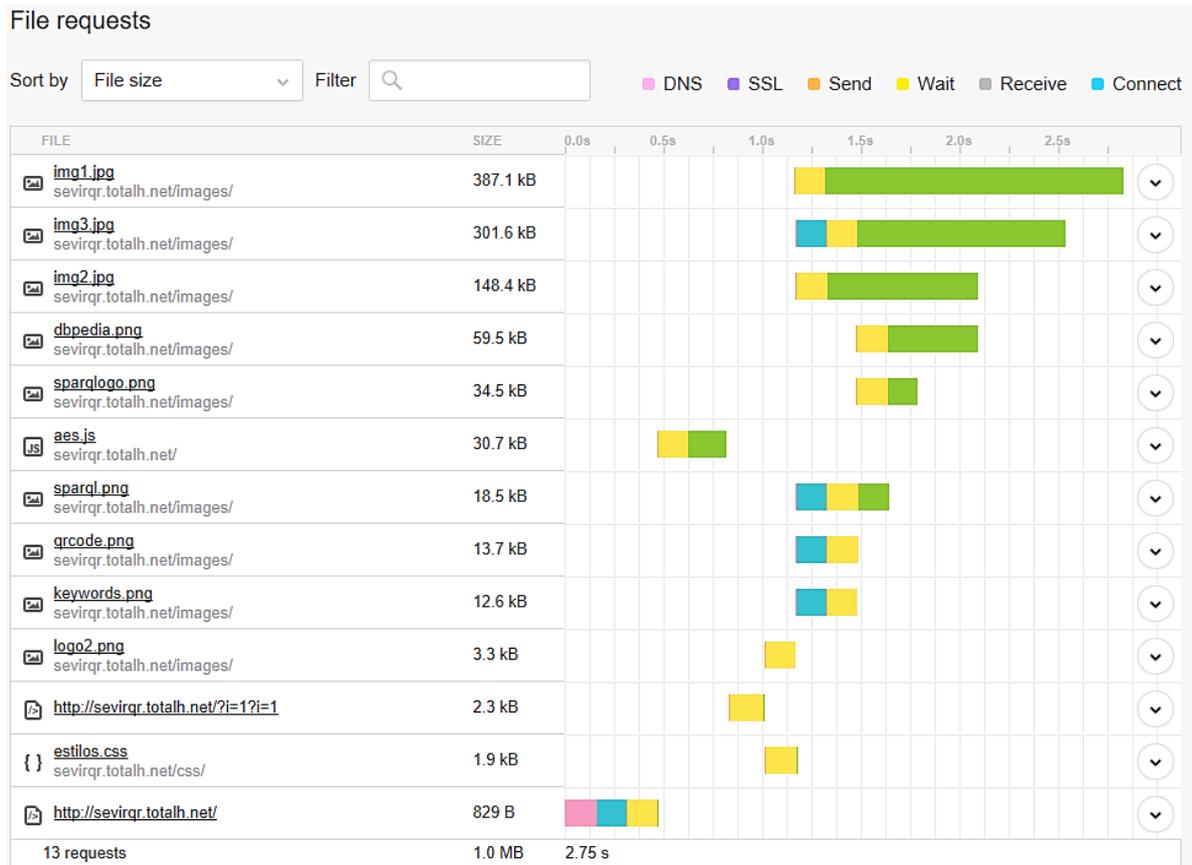


Figura 56. Resultados de la carga de objetos del sitio web

Fuente: Autor

Estas pruebas permitieron conocer que el sitio web tiene un peso bastante ligero (1 MB) y su tiempo de carga es muy corto (2.75 s). Sin embargo, se estima que su velocidad de carga dependerá del ancho de banda.

4.5. Pruebas de usabilidad y accesibilidad

Dentro de estas pruebas se contemplaron factores específicos como: facilidad de uso, diseño y compatibilidad con los dispositivos. Con ayuda de la herramienta *Google Play Console*, a continuación se detalla un mapa de las actividades con los servicios que posee la aplicación móvil. Esto permitió generar una tabla con la respectiva valoración en cuanto a complejidad de uso y calidad de diseño de la aplicación móvil.

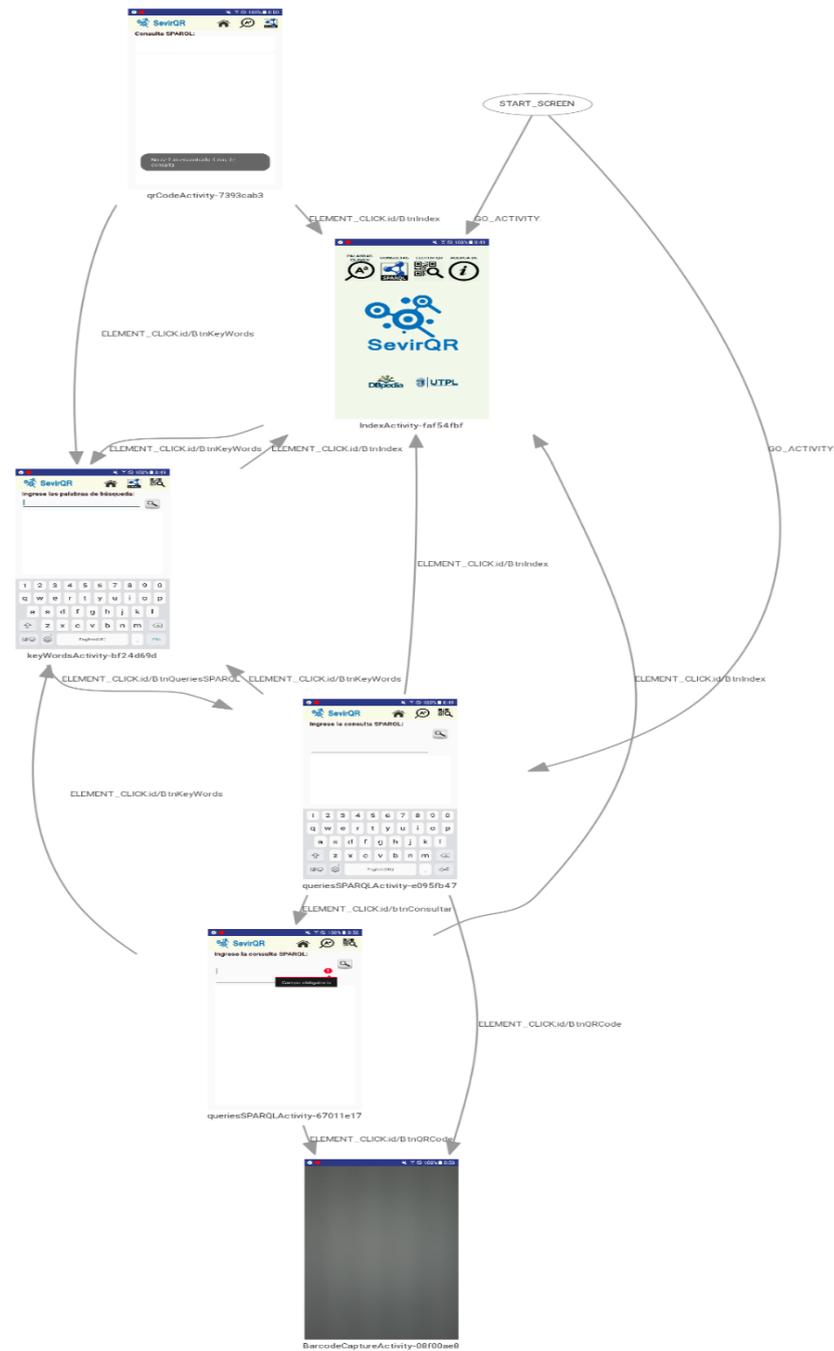


Figura 57. Mapa de actividad y navegación de la aplicación móvil

Fuente: Autor

Tabla 26. Resultados de usabilidad - Aplicación móvil

Servicio	Complejidad de uso	Calidad de diseño
Activity Principal	Fácil	Satisfactoria
Búsqueda por palabras clave	Fácil	Satisfactoria
Consultas SPARQL	Medio	Buena
Lector QR	Medio	Satisfactoria
Activity Acerca de	Fácil	Satisfactoria

Fuente: Autor

La siguiente figura describe una lista de algunos dispositivos móviles donde fue probada la aplicación y no se presentaron problemas.

Resultados de las pruebas de la versión 1 del APK			
TODAS LAS EDADES		PROBLEMAS	
Dispositivos con problemas	Dispositivos sin problemas	Dispositivos probados	
0	14	14	
Nombre del modelo	Versión de Android	Configuración regional	Descripción
✓ Nexus 5	Android 4.4	Árabe	-
✓ Nexus 5	Android 5.1	Inglés	-
✓ Galaxy Note3	Android 4.4	Inglés	-
✓ Galaxy J7(2016)	Android 6.0	Inglés	-
✓ Moto X (2nd Gen)	Android 4.4	Inglés	-
✓ Galaxy S6	Android 5.1	Inglés	-
✓ Nexus 7 (2013)	Android 5.0	Inglés	-
✓ Nexus 9	Android 5.0	Inglés	-
✓ Galaxy S7 Edge	Android 6.0	Inglés	-
✓ Galaxy J1 Ace	Android 5.1	Inglés	-
✓ LG G3	Android 4.4	Inglés	-
✓ One (M8)	Android 4.4	Alemán	-
✓ Pixel	Android 7.1	Inglés	-
✓ Pixel	Android O Preview	Inglés	-

Figura 58. Lista de dispositivos Android usados para prueba

Fuente: Autor

En el caso del sitio web se ha utilizado una herramienta llamada *PageSpeed Tools*¹², la cual analiza y permite realizar la optimización del sitio según las mejores prácticas, ya que identifica factores que mejoran la navegación y ayuda a que el sitio web sea amigable y adaptable a entornos móviles. A continuación se muestran los resultados obtenidos.

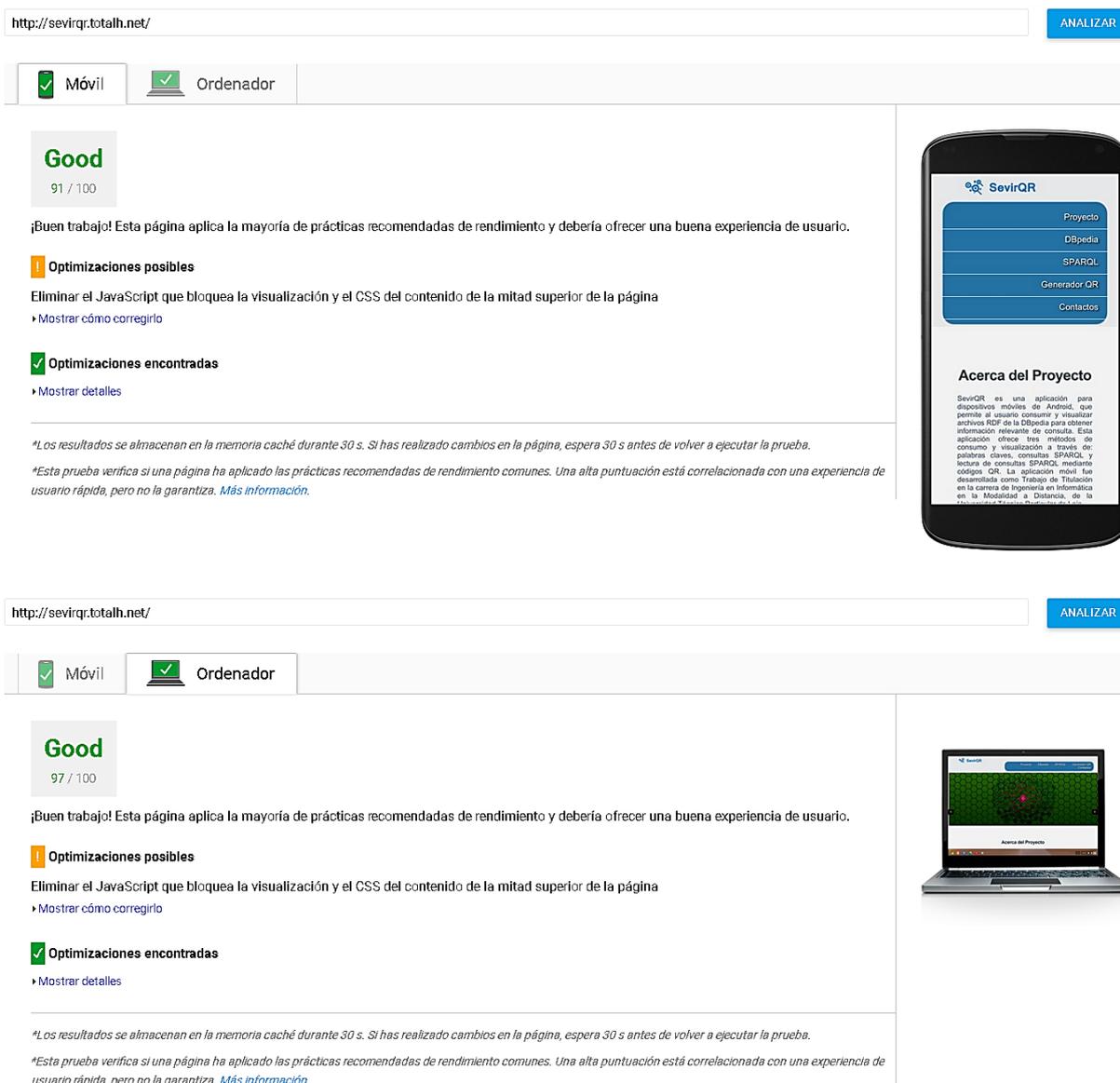


Figura 59. Resultados de optimización del sitio web

Fuente: Autor

De esta manera, es posible observar que el sitio web está optimizado y presenta buenas puntuaciones para ser utilizado tanto en dispositivos móviles (91/100) como en equipos de cómputo (97/100).

¹² PageSpeed Tools. Disponible en: <https://developers.google.com/speed/pagespeed/>

Adicionalmente, es importante mencionar que el sitio web fue probado en la siguiente lista de navegadores.

Tabla 27. Lista de navegadores usados para prueba

Navegador	Versión
Firefox	54.0.1 (32-bit)
Internet Explorer	11.0.43
Microsoft Edge	40.15063
Google Chrome	59.0.3071.115

Fuente: Autor

4.6. Implementación

La implementación de la aplicación móvil se la realizó a través de *Google Play*¹³, que es el principal servicio de Google para ofrecer aplicaciones móviles a todos los usuarios que utilizan dispositivos de Android.

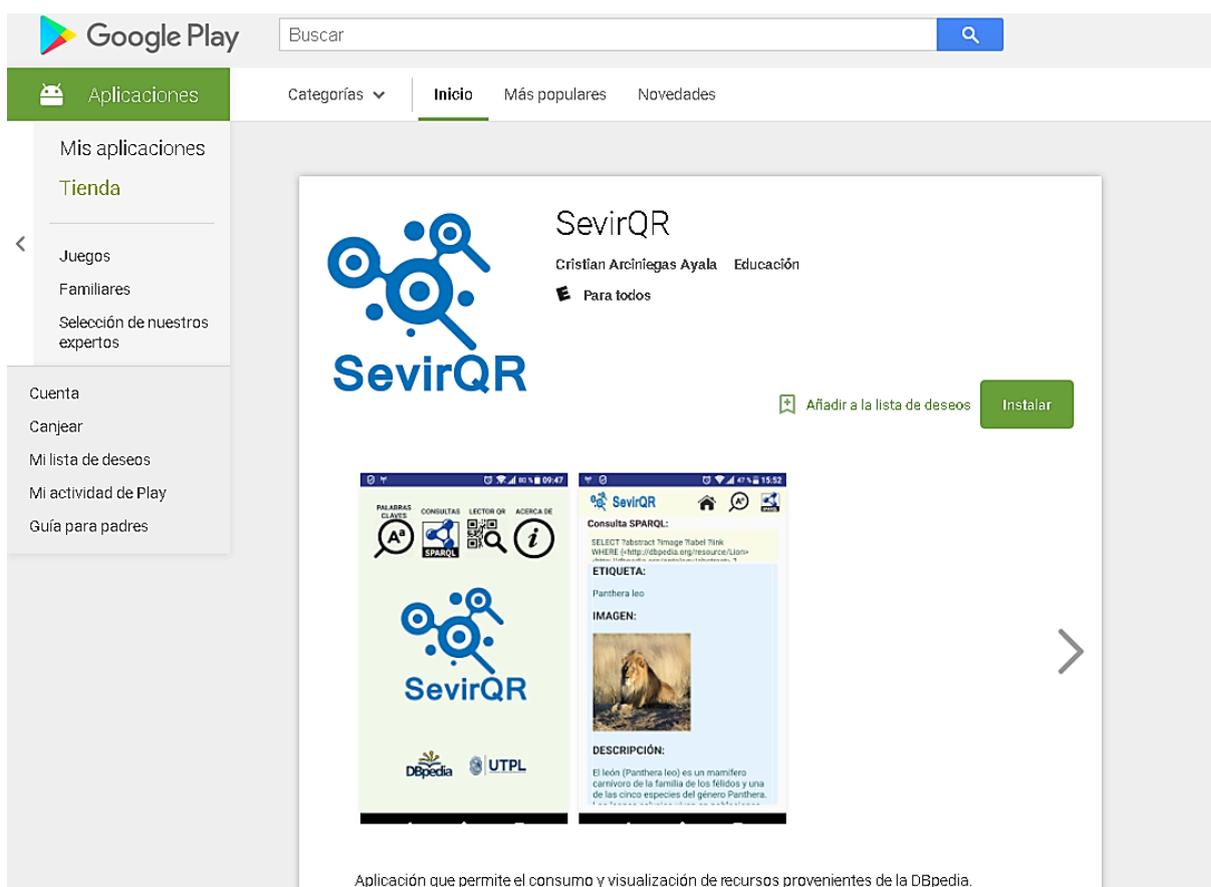


Figura 60. Implementación de la aplicación móvil en Google Play

Fuente: Autor

¹³ Google Play. Disponible en: <https://play.google.com/store?hl=es>

La implementación del sitio web se la realizó mediante la utilización del hosting gratuito Eshost¹⁴, donde <http://sevirqr.totalh.net>, es la dirección actual del sitio web implementado.

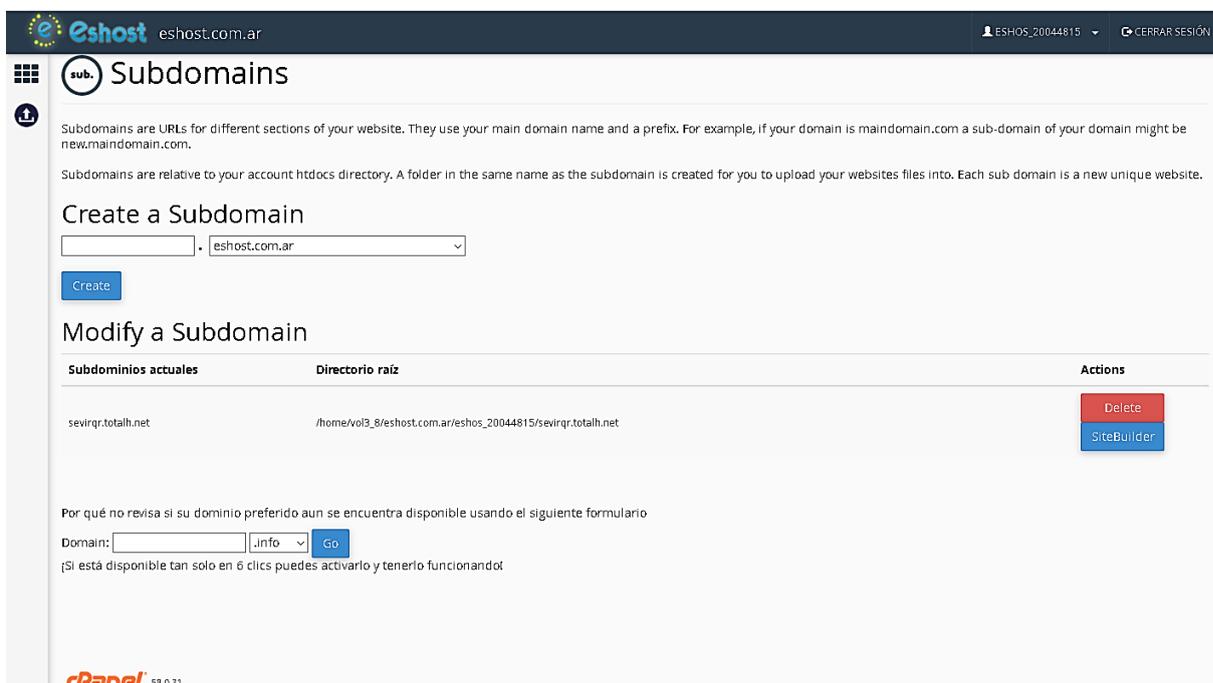


Figura 61. Implementación del sitio web en hosting gratuito

Fuente: Autor

¹⁴ Eshost. Disponible en: <http://eshost.com.ar>

CONCLUSIONES

Es importante destacar la importancia y el potencial que tiene la Web Semántica en la representación de los datos, que ya gracias a ello fue posible obtener valiosa información y conocimiento de uno de los repositorios semánticos más conocidos como lo es la DBpedia.

Las librerías en java que proporcionan ayuda para la conexión hacia los datos que provee la DBpedia, están mejor desarrollados y adaptados para ambientes web que para los ambientes móviles, especialmente si se requiere utilizar la herramienta Android Studio.

Lograr entender el funcionamiento de los mecanismos nativos de obtención de datos que proporciona la DBpedia, fue crucial para poderlos implementar en un ambiente móvil ya que fue muy complicada la adaptación de librerías externas.

Fue interesante e importante conocer las diferentes características que poseen los códigos QR, ya que esto ayudó a mejorar la calidad de codificación y generación de las consultas SPARQL para su posterior lectura.

Es muy relevante e importante poseer un buen conocimiento y manejo del lenguaje de consultas semánticas SPARQL, ya que como desarrolladores nos ayuda a saber elaborar mejores consultas SPARQL capaces de obtener resultados optimizados en las búsquedas sobre cualquier repositorio semántico.

Los recursos de DBpedia están en continua actualización y optimización, es así que cualquier consulta SPARQL que funciona correctamente en este momento, podría dejar de funcionar adecuadamente en el futuro. Con esto no se quiere decir que los recursos que ofrece la DBpedia sean muy variables o inestables.

Una posible mejora a este trabajo sería proponer un diseño mejorado de visualización de los datos obtenidos o estudiar una posible optimización en el consumo, independientemente del ancho de banda y de las características de los dispositivos móviles utilizados.

RECOMENDACIONES

Es recomendable la utilización de los mecanismos nativos para la comunicación y obtención de datos desde la DBpedia a diferencia de otros mecanismos externos, debido a su facilidad de implementación en cualquier plataforma y a su baja complejidad de codificación.

En el mecanismo de búsqueda por palabras donde se utiliza el servicio Lookup DBpedia, se recomienda en lo posible no utilizar frases largas (es preferible usar términos individuales específicos) puesto que en dichos casos es posible que no se obtengan los resultados buscados, o en definitiva, puede que no se obtenga ningún resultado.

Puesto que para la generación de los códigos QR se utilizó la versión de código 18 y un nivel de recuperación-corrección L, se recomienda utilizar consultas SPARQL que contengan hasta un máximo de 1.046 caracteres (alfanuméricos).

Para la construcción de sitios web en otros trabajos similares, es recomendable utilizar HTML5 y CSS3 ya que permiten la elaboración de sitios web ligeros y adaptativos a dispositivos móviles.

BIBLIOGRAFÍA

- Agenjo, X., Hernandez, F. (2015). *Cómo y qué consumir en el mundo Linked Open Data; cómo y qué producir en Linked Open Data*. FESABID'15. Recuperado a partir de <http://www.digibis.com/images/PDF/fesabid-2015-lod.pdf>
- Aghaei, S., Nematbakhsh, M., & Farsani, H. (2012). *Evolution of the World Wide Web: From Web 1.0 to Web 4.0*. *International journal of Web & Semantic Technology*, 3(1), 1-10. Recuperado a partir de <https://doi.org/10.5121/ijwest.2012.3101>
- Almendros-Jiménez, J. M. (2008). *An RDF Query Language based on Logic Programming*. *Electronic Notes in Theoretical Computer Science*, 200(3), 67-85. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S1571066108003083>
- Android Studio | Developer.android.com*. (2017). *Conoce Android Studio*. Recuperado el 18 de abril de 2017, desde <https://developer.android.com/studio/intro/index.html?hl=es-419>
- Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). *DBpedia: A nucleus for a web of open data*. *The semantic web* (pp. 722–735). Springer. Recuperado a partir de http://link.springer.com/chapter/10.1007/978-3-540-76298-0_52
- Berners-Lee, T., Hendler, J., Lassila, O., & others. (2001). *The semantic web*. *Scientific american*, 284(5), 28–37. Recuperado a partir de https://www.sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf
- Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). *DBpedia - A crystallization point for the Web of Data*. *Web Semantics: Science, Services and Agents on the World Wide Web.*, 7(3), 154-165. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S1570826809000225>
- Bjornar, T., Bjarte, J., Csaba, V. (2013). *Mobile Location-Driven Associative Search in DBpedia with Tag Clouds*. *Proceedings of the I-SEMANTICS 2013 Posters & Demonstrations Track*. Recuperado a partir de <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.402.2305&rep=rep1&type=pdf>
- Bobed, C., & Mena, E. (2016). *QueryGen: Semantic interpretation of keyword queries over heterogeneous information systems*. *Information Sciences*, 329, 412-433. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S0020025515006672>
- Bustos, M., Perez, N., & Berón, M. (2015). *Plataformas para el desarrollo de aplicaciones móviles*. En XVII Workshop de Investigadores en Ciencias de la Computación (Salta, 2015). Recuperado a partir de <http://sedici.unlp.edu.ar/handle/10915/45851>
- Castro, V. (2014). *Web semántico e Linked Open Data: Best practices, prospettive e criticità*. *Nuovi Annali Della Scuola Speciale Per Archivisti e Bibliotecari*, 28, 207-221. Recuperado a partir de <http://search.proquest.com/docview/1651714733?accountid=45668>
- Cazenave, F., Quint, V., & Roisin, C. (2011). *Timesheets.js: when SMIL meets HTML5 and CSS3*. En *Proceedings of the 11th ACM symposium on Document engineering* (pp. 43–52). ACM. Recuperado a partir de <http://dl.acm.org/citation.cfm?id=2034700>
- Cheniki, N., Belkhir, A., & Atif, Y. (2016). *Mobile services discovery framework using DBpedia and non-monotonic rules*. *Computers & Electrical Engineering*, 52, 49-64. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S0045790616300325>
- Codina, L., & Rovira, C. (2006). *La web semántica. Tendencias en documentación digital*.

- Recuperado a partir de <http://eprints.rclis.org/handle/10760/8899>
- Consortium, W.W.W., & others. (2008). *SPARQL Lenguaje de consulta para RDF*. Recuperado a partir de <http://travesia.mcu.es/portalnb/jspui/handle/10421/6401>
- Curé, O. (2014). *On the design of a self-medication web application built on linked open data*. *Web Semantics: Science, Services and Agents on the World Wide Web*, 24, 27-32. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S157082681400002X>
- Curé, O., & Blin, G. (2015). *RDF Database Systems: Triples Storage and SPARQL Query Processing*. Morgan Kaufmann: Boston. Chapter Three - RDF and the Semantic Web Stack, 41-80. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/B9780127999579000031>
- Deemer, P., Benefield, G., Larman, C., & Vodde, B. (2010). *The scrum primer. Scrum Primer is an in-depth introduction to the theory and practice of Scrum*. Recuperado a partir de <http://assets.scrumtraininginstitute.com/downloads/1/scrumprimer121.pdf>
- Fahad, M., Boissier, O., Maret, P., Moalla, N., & Gravier, C. (2014). *Smart places: Multi-agent based smart mobile virtual community management system*. *Applied Intelligence*, 41(4), 1024-1042. Recuperado a partir de <http://link.springer.com/article/10.1007%2Fs10489-014-0569-2>
- Friesen, J. (2016). *Android studio for beginners, part 1: Installation and setup*. *JavaWorld*. Recuperado a partir de <https://search.proquest.com/docview/1813808422?accountid=45668>
- Fulantelli, G., Taibi, D., & Arrigo, M. (2015). *A framework to support educational decision making in mobile learning*. *Computers in Human Behavior*, 47, 50-59. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S074756321400332X>
- Ghorbel, F., Ellouze, N., Métais, E., Hamdi, F., Gargouri, F., & Herradi, N. (2016). *MEMO GRAPH: An Ontology Visualization Tool for Everyone*. *Procedia Computer Science*, 96, 265-274. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S1877050916319408>
- Hallo, M., Martínez-González, M., & De la Fuente Redondo, P. (2012). *Las tecnologías de Linked Data y sus aplicaciones en el gobierno electrónico*. *SCIRE: representación y organización del conocimiento*, 18(1), 49–61. Recuperado a partir de <http://ibersid.eu/ojs/index.php/scire/article/view/3923>
- Hayashi, T., & Ohsawa, Y. (2015). *Knowledge Structuring and Reuse System Using RDF for Supporting Scenario Generation*. *Procedia Computer Science*, 60, 1281-1288. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S1877050915023601?via%3Dihub>
- Historia de la Web 2.0 – Historia de la Informática*. (2010). *Histinf.blogs.upv.es*. Recuperado el 13 de febrero de 2017, desde <http://histinf.blogs.upv.es/2010/12/12/historia-de-la-web-2-0/>
- Holla, S., & Katti, M. (2012). *Android based mobile application development and its security*. *International Journal of Computer Trends and Technology*, 3(3), 486–490. Recuperado a partir de <http://ijcttjournal.org/Volume3/issue-3/IJCTT-V3I3P130.pdf>
- Im, D., & Park, G. (2015). *Linked tag: image annotation using semantic relationships between image tags*. *Multimedia Tools and Applications*, 74(7), 2273-2287. Recuperado a partir de <http://link.springer.com/article/10.1007%2Fs11042-014-1855-z>

- Kleinen, A., Scherp, A., & Staab, S. (2014). *Interactive faceted search and exploration of open social media data on a touchscreen mobile phone*. *Multimedia Tools and Applications*, 71(1), 39-60. Recuperado a partir de <http://link.springer.com/article/10.1007%2Fs11042-013-1366-3>
- Lukovnikov, D., Kontokostas, D., Stadler, C., Hellmann, S., & Lehmann, J. (2014). *DBpedia Viewer-An Integrative Interface for DBpedia leveraging the DBpedia Service Eco System*. Recuperado a partir de <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.650.4911&rep=rep1&type=pdf>
- Lupiani, E., et al. (2010). INVOCA: Consultando la Linked Open Data en Lenguaje Natural. *Procesamiento Del Lenguaje Natural*, 45, 323-324. Recuperado a partir de <http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/828>
- Paredes, M., Rodríguez, M., Ruiz, A., Valencia, R., & Alor, G. (2015). *ONLI: An ontology-based system for querying DBpedia using natural language paradigm*. *Expert Systems with Applications*, 42(12), 5163-5176. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S0957417415001414>
- Piedra, N., Chicaiza, J., Cadme, E., & Guaya, R. (2016). *Una aproximación basada en Linked Data para la detección de potenciales redes de colaboración científica a partir de la anotación semántica de producción científica: Piloto aplicado con producción científica de investigadores ecuatorianos*. *Maskana*, 65(Supl.). Recuperado a partir de https://www.researchgate.net/publication/274834503_Una_aproximacion_basada_en_Linked_Data_para_la_deteccion_de_potenciales_redes_de_colaboracion_cient'ifica_a_partir_de_la_anotacion_semantica_de_produccion_cient'ifica_Piloto_aplicado_con_produccion_ci
- Powell, J., & Hopkins, M. (2015). A librarian's guide to graphs, data and the semantic web. *Chados Publishing*. 4 - *RDF and its serialization*, 21-30; 6 - *SPARQL*, 45-53. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/B978184334753800004X> & <http://www.sciencedirect.com/science/article/pii/B9781843347538000063>
- RDF 1.1 JSON Alternate Serialization (RDF/JSON)*. (2013). *Dvcs.w3.org*. Recuperado el 1 de noviembre de 2016, desde <https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-json/index.html>
- Resource Description Framework (RDF) Model and Syntax Specification*. (1999). *W3.org*. Recuperado el 31 de octubre de 2016, desde <https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>
- Semantic Web - XML2000 - Slide "Architecture"*. (2000). *W3.org*. Recuperado el 21 de octubre de 2016, desde <https://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html>
- Schwaber, K. (1997). *Scrum development process*. Recuperado a partir de http://link.springer.com/10.1007/978-1-4471-0947-1_11
- Serón, F., & Bobed, C. (2014). *VOX system: a semantic embodied conversational agent exploiting linked data*. *Multimedia Tools and Applications*, 75(1), 381-404. Recuperado a partir de <http://link.springer.com/article/10.1007%2Fs11042-014-2295-5>
- Vagliano, I., Figueroa, C., Rocha, O., Torchiano, M., Faron-Zucker, C., & Morisio, M. (2016). *ReDyAl: A Dynamic Recommendation Algorithm based on Linked Data*. En 3rd Workshop on New Trends in Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems (RecSys 2016) (Vol. 1673). Recuperado a partir de <https://hal.inria.fr/hal-01379996/>
- Vanegas, C. (2013). *Desarrollo de aplicaciones sobre Android*. *Revista Vínculos*, 9(2), 129-145. Recuperado a partir de

<http://revistas.udistrital.edu.co/ojs/index.php/vinculos/article/view/4275>

Xu, F. (2014). *QR Codes and library bibliographic records*. VINE, 44(3), 345-356. Recuperado a partir de <http://www.emeraldinsight.com/doi/abs/10.1108/VINE-12-2013-0070>

Zapater, J., & Calderaro, J. (2013). *Tema 1. Evolución de la Web*. Recuperado a partir de <http://ocw.uv.es/ingenieria-y-arquitectura/aplicaciones-web-avanzadas/tema1.pdf>

Zingla, M., Chiraz, L., & Slimani, Y. (2016). *Short Query Expansion for Microblog Retrieval*. Procedia Computer Science, 96, 225-234. Recuperado a partir de <http://www.sciencedirect.com/science/article/pii/S1877050916319366?via%3Dihub>

ANEXOS

*CONSUMO Y VISUALIZACIÓN DE ARCHIVOS RDF
DE LA DBPEDIA EN DISPOSITIVOS MÓVILES DE
ANDROID.*



Documento de Visión

Versión 1.0

HISTORIAL DE REVISIONES

Fecha	Versión	Descripción	Autor
2017-03-06	1.0	Versión inicial del documento de visión para el desarrollo de la propuesta.	Arciniegas Ayala Cristian Vinicio

1. Introducción

El siguiente documento de visión describe el proyecto SevirQR, el mismo que trata sobre el desarrollo de una aplicación móvil que realizará el consumo y visualización de información semántica obtenida de la DBpedia, con el fin de facilitar el acceso a esta información y que cualquier usuario la pueda consultar y visualizar.

1.1 Propósito

Este documento proporciona una visión general del proyecto denominado SevirQR (Search & Visualization RDF with QR).

1.2 Alcance

Este proyecto plantea el desarrollo de una aplicación para dispositivos móviles de Android que permita al usuario realizar consultas y obtener información de la DBpedia, cuyo repositorio ofrece una cantidad valiosa de conocimiento.

Para alcanzar este objetivo, la aplicación móvil proveerá tres mecanismos que faciliten al usuario realizar consultas y observar los datos obtenidos a través de visualizadores. Además, se desarrollará un sitio web que a más de informar acerca de este proyecto, proporcionará un generador de códigos QR para poder codificar consultas SPARQL.

2. Posicionamiento

2.1. Oportunidades de Negocio

Con el desarrollo de este proyecto se pretende facilitar el acceso a los datos e información que ofrece un repositorio semántico como es DBpedia, a través del uso de un dispositivo móvil. De esa forma, también se busca conseguir que la Web Semántica pueda estar un poco más al alcance del usuario común para que éste pueda apreciar las bondades que esta web ofrece.

2.2. Planteamiento del problema

Tabla 28. Planteamiento del problema

Problema:	Se puede evidenciar que el acceso al conocimiento y la información que ofrece la Web Semántica en repositorios semánticos tales como DBpedia, presenta ciertas limitaciones para que cualquier usuario la pueda obtener fácilmente como lo haría al utilizar un buscador común (desde Google), y especialmente, a través de un dispositivo móvil de Android.
Afectados:	Usuarios finales
Impacto:	El conocimiento que se puede obtener de los repositorios semánticos (DBpedia) es muy valioso, y al no contar con mecanismos sencillos que permitan explotar este recurso, los mismos no pueden ser aprovechados por los usuarios.
Solución exitosa:	Desarrollar una aplicación para dispositivos móviles de Android que permita al usuario realizar consultas y obtener información de la DBpedia, para lo cual se implementarán mecanismos sencillos de consulta y visualizadores que permitan mostrar los datos obtenidos.

Fuente: Autor

3. Descripción de Stakeholders

Es importante identificar a los interesados del proyecto y sus diferentes perfiles, para conocer cuál es el rol que desempeñan y qué influencia pueden ejercer en el desarrollo del mismo.

3.1 Resumen de Stakeholders

A continuación se describen a los diferentes interesados del proyecto.

Tabla 29. Resumen de Stakeholders

Stakeholder	Descripción	Responsabilidad
Director del Proyecto	Responsable de llevar el proyecto a un término exitoso.	Proporcionar las pautas y directrices para el correcto desarrollo del proyecto.
Administrador del Proyecto	Responsable del desarrollo y la gestión del proyecto	Planificar, organizar y ejecutar el proceso de construcción de los aplicativos.
Usuarios Técnicos	Personas con conocimientos en Web Semántica.	Consumir y visualizar los datos de la DBpedia.
Usuarios Finales	Personas sin conocimientos previos de Web Semántica.	Consumir y visualizar los datos de la DBpedia.

Fuente: Autor

3.2 Perfil de los stakeholders

Los perfiles de los interesados permiten describir de una forma más detallada a cada uno de los implicados en el proyecto. En seguida se coloca el perfil de cada stakeholder.

3.2.1 Director del Proyecto

Tabla 30. Perfil de los stakeholders: Director del Proyecto

Nombre	Ramiro Ramírez – Director del Proyecto
Descripción	Responsable por la culminación exitosa del proyecto.
Función	Director
Responsabilidad	Brindar las pautas y directrices para el desarrollo normal del proyecto. Evaluar el avance en el desarrollo del proyecto.
Criterios de éxito	Poder finalizar el proyecto en los tiempos establecidos. Hacer que el proyecto cumpla con los requerimientos solicitados.

Fuente: Autor

3.2.2 Administrador del Proyecto

Tabla 31. Perfil de los stakeholders: Administrador del Proyecto

Nombre	Cristian Arciniegas – Administrador del Proyecto
Descripción	Responsable por el desarrollo exitoso del proyecto.
Función	Tesista
Responsabilidad	Velar por la correcta ejecución y desarrollo del proyecto. Gestionar y solucionar los problemas que obstaculicen el normal progreso del proyecto.
Criterios de éxito	Poder entregar un producto funcional al culminar el tiempo establecido. Haber alcanzado todos los objetivos planteados en el proyecto.

Fuente: Autor

3.2.3 Usuarios Técnicos y Finales

Tabla 32. Perfil de los stakeholders: Usuarios técnicos y usuarios finales

Nombre	Usuarios Técnicos y Finales
Descripción	Usuarios que utilizan el producto terminado.
Función	Usuario
Responsabilidad	Hacer uso de los aplicativos construidos y emitir su opinión.
Criterios de éxito	Obtener un producto que verdaderamente supla las necesidades y requerimientos de consultas en repositorios semánticos.

Fuente: Autor

4. Visión General del Producto

4.1 Detalle del producto

En este proyecto se busca desarrollar una aplicación para dispositivos móviles de Android que permita el consumo y visualización de información proveniente del repositorio de la DBpedia. Para esto, se plantea la implementación de tres mecanismos de consulta que permitan realizar el proceso de extracción y visualización de la información. Adicionalmente, se desarrollará un sitio web que proveerá un generador de códigos QR, el cual permitirá la codificación de consultas SPARQL que serán utilizadas posteriormente por la aplicación móvil para la ejecución de consultas automáticas.

4.2 Resumen de características del producto

Tabla 33. Resumen de características del producto

Beneficios	Características
Buscador mediante palabras claves.	Permite realizar búsquedas ingresando palabras claves, y cuyos resultados obtenidos tendrán relación con dichas palabras.
Buscador mediante digitación de consultas SPARQL.	Permite digitar o copiar y pegar consultas SPARQL que se ejecuten y devuelvan información directamente de la DBpedia.
Buscador mediante lector de códigos QR.	Permite leer un código QR que contiene una consulta SPARQL, la misma que será leída y ejecutada automáticamente, para luego mostrar los resultados obtenidos.
Visualizadores de datos obtenidos.	Muestran al usuario la información o datos obtenidos de cada mecanismo de consulta utilizado.
Generador de códigos QR para consultas SPARQL	Permite codificar las consultas SPARQL digitadas en los respectivos códigos QR que son usados por la aplicación móvil.

Fuente: Autor

5. Limitaciones del producto

Existen algunas limitantes que se deberán tomar en cuenta en relación al producto final. Estas limitaciones son:

- ❖ La aplicación móvil estará disponible para ciertas versiones de software específicas de Android. (Se recomienda usarla en dispositivos Android con la versión 5.0 o superior)
- ❖ Al realizar la búsqueda mediante el mecanismo de palabras claves, no es posible utilizar términos compuestos o frases largas, puesto que el mecanismo no será capaz de encontrar resultados exactos.
- ❖ Las consultas SPARQL deben estar correctamente escritas para que puedan ser ejecutadas por la aplicación móvil.

*CONSUMO Y VISUALIZACIÓN DE ARCHIVOS RDF
DE LA DBPEDIA EN DISPOSITIVOS MÓVILES DE
ANDROID.*



Documento de Plan de Pruebas

Versión 1.0

HISTORIAL DE REVISIONES

Fecha	Versión	Descripción	Autor
2017-04-18	1.0	Versión inicial del plan de pruebas para el desarrollo de la propuesta.	Arciniegas Ayala Cristian Vinicio

1 Introducción

El documento de Plan de Pruebas tiene como objetivo detallar las diferentes pruebas que serán aplicadas a los respectivos productos descritos en este proyecto, para verificar que se cumplan los objetivos propuestos y además garantizar su correcto funcionamiento previo al proceso de implementación.

1.1 Propósito

Ejecutar un plan de pruebas permitirá descubrir problemas o carencias de los productos construidos, con lo cual se podrá realizar las correcciones correspondientes y así garantizar que todos los componentes funcionen de la manera adecuada y cumplan el propósito para el que fueron diseñados. De esa manera, se asegura que los productos funcionarán una vez que sean liberados para su utilización por parte de los usuarios.

1.2 Alcance

Este documento pretende ser una guía en la realización de las pruebas a realizar, pues al utilizar la metodología de desarrollo SCRUM se tiene la flexibilidad de elegir la documentación a utilizar. Por tal razón, este documento servirá para documentar las pruebas de software realizadas en los diferentes incrementos de Sprint.

2 Estrategia de Pruebas

A continuación se detallan los tipos de pruebas que se ejecutarán a cada uno de los aplicativos de este proyecto:

- **Pruebas de aceptación:** que permitirán verificar si los productos cumplen con los requerimientos y objetivos planteados.
- **Pruebas de carga:** que indicarán la capacidad que tienen los aplicativos para poder atender las solicitudes de los usuarios.
- **Pruebas de usabilidad-accesibilidad:** que permitirán conocer si los aplicativos son de fácil uso, si las interfaces son amigables al usuario y en qué dispositivos o sistemas podrán ser instaladas o utilizadas las aplicaciones desarrolladas.

3 Herramientas para ejecución del plan de pruebas

Las herramientas que se utilizarán para apoyar el plan de pruebas se describen a continuación.

Tabla 34. Herramientas para ejecución del plan de pruebas

Tipo de Prueba	Ambiente	Herramienta	Descripción
Aceptación	Web - Móvil	Test	Permite verificar el cumplimiento de los requerimientos funcionales.
Carga	Móvil	Google Play Console	Permite comprobar la capacidad que una aplicación puede soportar frente a la demanda de los usuarios.
	Web	Pingdom Tools	
Usabilidad – Accesibilidad	Móvil	Google Play Console	Permite conocer que tan fácil de usar son las aplicaciones, si tiene un diseño amigable y sobre qué tipos y versiones de dispositivos funcionarán.
	Web	PageSpeed Tools	

Fuente: Autor

4 Ejecución de Pruebas

Para la realización de las pruebas, se configurará cada uno de los aplicativos en ambientes similares al que trabajarán una vez que sean implementados para su respectiva utilización por parte de los usuarios.

En el caso de la aplicación móvil se utilizará el archivo *.apk* generado por Android Studio, y para el sitio web, las pruebas se realizarán directamente en el mismo hosting donde se piensa realizar su implementación final.