UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 1

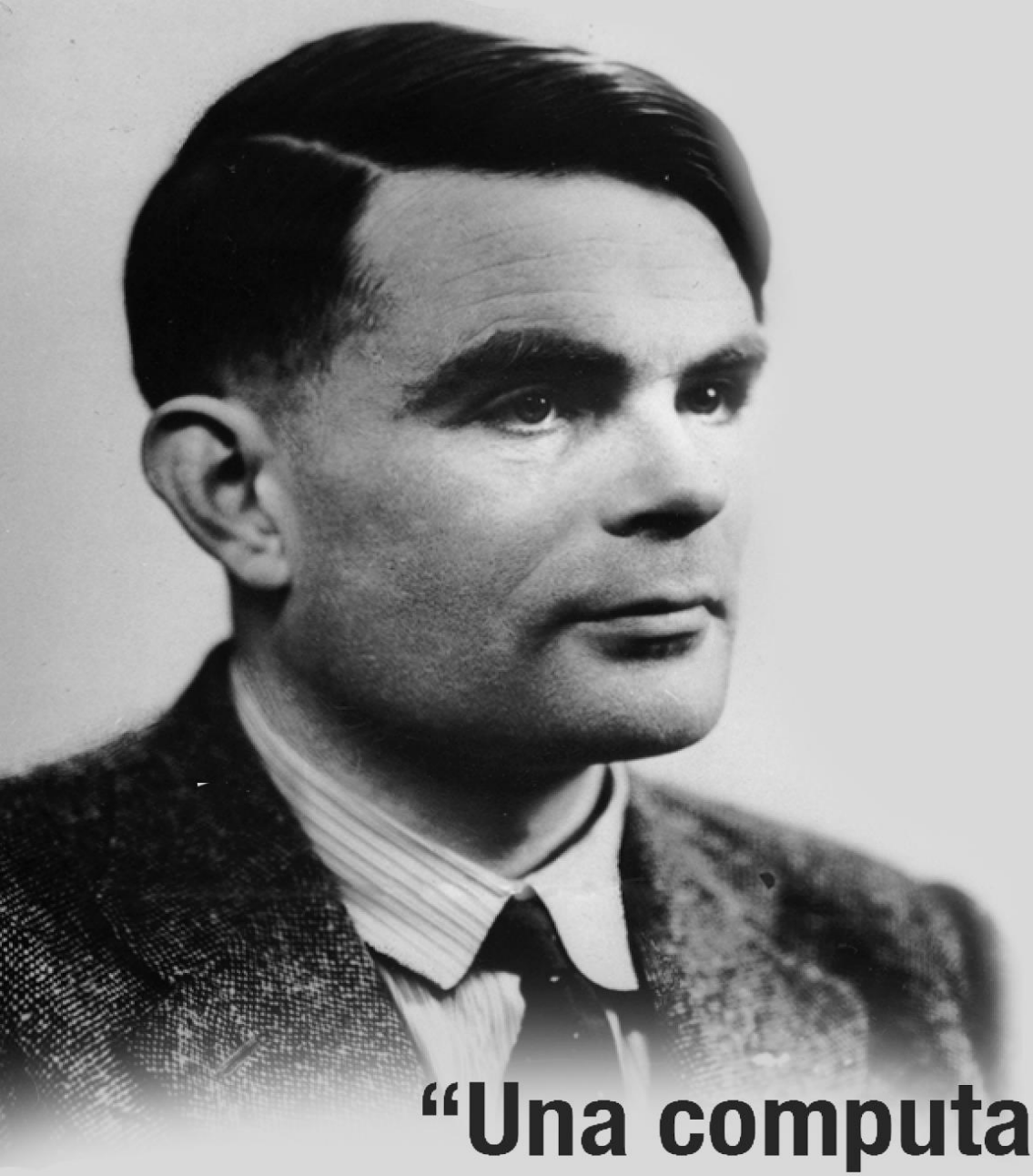
SUBSISTEMA PARA DETERMINAR RESPUESTA EXACTA EN LA PLATAFORMA C.U.B.A.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:
Jorge Amado Hernández Betancourt

Tutores:
Ing. Serguey González Garay
Ing. Eric Bárbaro Utrera Sust

La Habana, Junio 2018



“Una computadora puede ser llamada inteligente si logra engañar a una persona haciéndole creer que es un humano”

Alan Mathison Turing
1912–1954

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo Jorge Amado Hernández Betancourt, con carnet de identidad 94021304904, soy el autor principal del trabajo titulado “Subsistema para determinar respuesta exacta en la plataforma C.U.B.A.” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales de la misma con carácter exclusivo.

Para que así conste firmo la presente a los____ días del mes de ____ del año _____.

Firma del Autor:

_____

Jorge Amado Hernández Betancourt

Firma del Tutor:

_____

Ing. Serguey González Garay

Firma del Tutor:

_____

Ing. Eric Bárbaro Utrera Sust

AGRADECIMIENTOS

Agradezco principalmente a mis padres, por haber luchado junto conmigo en este largo camino que es la formación de mi persona.

Me siento agradecido de mi familia en general, a los que están aquí, y los que están lejos también.

También a todos mis hermanos, que se encuentran en mi escuela y que juntos intercambiamos momentos buenos y malos; Mauro, Raúl, Manuel, Carlitos, Juan Carlos, Michel, Ramón y toda la familia del edificio 123.

Muchas gracias a mis amigos de toda la vida; Rubén, Arbelio, Joel, Lisset y René, que a pesar de haber pasado el tiempo y seguir diferentes caminos siguen ahí conmigo para lo que haga falta.

Le agradezco a mis compañeros de aula y año, los que llegaron conmigo hasta el día de hoy y los que no también.

Agradezco a todos mis profesores que me enseñaron durante 5 años a ser lo que soy ahora y me encuentro conforme con los resultados.

A la ACM-ICPC, que a lo largo de mi carrera ha significado mucho para mí y ha contribuido a mi preparación.

A los integrantes del movimiento de programación competitiva de la UCI, que juntos hemos intercambiado experiencias inolvidables.

Me siento agradecido de mis tutores, Serguey y Eric, que me han encaminado a lo largo del desarrollo de esta tesis.

Le agradezco de manera general a todas esas personas que, a lo largo de mi carrera, me han ayudado y me han apoyado, habiendo alcanzado mi sueño de ser Ingeniero en Ciencias Informáticas.

Por último, quiero darle un agradecimiento especial a Claudia, por su gran apoyo incondicional y por aguantarme tantos años sin dejar de estar a mi lado.

DEDICATORIA

Le dedico mi tesis a mi hermana, a mi mamá y a mi papá. Son las personas que más han contribuido a mi formación como persona; y hoy todo lo que he logrado, lo he hecho gracias a ellos.

RESUMEN

Los sistemas de pregunta-respuesta perfeccionan el procesamiento de la información al mejorar la precisión y exhaustividad de los resultados obtenidos ante una necesidad de búsqueda de los usuarios. En los últimos tiempos se han desarrollado subsistemas internacionales incluidos en los buscadores Web que permiten responder de manera exacta a una pregunta utilizando tecnologías de web semántica. A Cuba se les dificulta el acceso a estos y varios no representan el contenido de la información cubana. La presente investigación tiene como objetivo contribuir a mejorar los resultados de las búsquedas realizadas por los usuarios en la intranet nacional. Con este fin, se realiza un estudio a los sistemas pregunta-respuesta a partir de un grupo de características que permiten definir los factores a tener en cuenta para el desarrollo de la solución propuesta. Para validar el resultado obtenido como parte de la investigación se realiza un cuasi-experimento que permite comparar los resultados obtenidos al realizar búsquedas en la plataforma de contenidos unificados para la búsqueda avanzada con la propuesta de solución, en cuanto a un indicador precisión. Esta validación permitió corroborar el aporte de la solución propuesta al entregar a los usuarios en un tiempo menor, un resultado más preciso de la información buscada en la plataforma.

Palabras clave: base de conocimiento, búsqueda, sistema de pregunta-respuesta, semántica, Web.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Cada año crece la población en el mundo, la cantidad de personas con acceso a internet y la cantidad de publicaciones en la Web en general; en conjunto con las capacidades de computación y almacenamiento. Esto ha provocado que los documentos en la web pierdan facilidad de acceso o localización. En disímiles ocasiones los usuarios no saben cómo encontrar lo que realmente necesitan entre tantos documentos existentes en la web. A continuación, en la Figura 1 se muestra cómo crece el tráfico de IP[1] en internet.

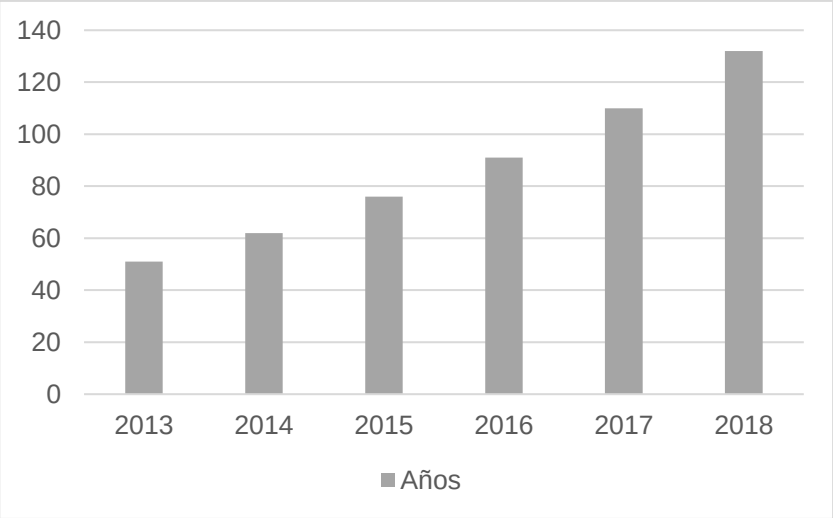


Figura 1: Crecimiento del tráfico de IP dado en Exabytes (Cisco, 2018)

La cantidad de información almacenada trajo consigo la necesidad de crear herramientas con el fin de revisar de manera general todo lo que se almacena en la Web. Estas herramientas, son conocidas como buscadores y cada vez son más imprescindibles en los contextos cotidianos. Estos deben determinar la respuesta en un tiempo determinado, así como la relevancia de los artículos que se van encontrando con respecto a las necesidades que determina el usuario (Romá, 2014).

Según López (2014) un buscador web es un Sistema de Recuperación de Información que tiene por misión devolver, dada una consulta planteada por un usuario, los documentos más relevantes de acuerdo a la consulta. Los documentos pertenecen a un índice donde se almacena luego del proceso de rastreo en Internet. Estos se recuperan generalmente mediante algoritmos básicos de trabajo con cadenas y utilizando

[1] Protocolo de Internet (Moreno, 2015)

filtros para la búsqueda (Rao, 2017), causando que muchas veces recuperen documentos con poca relevancia a la interrogante.

Aun así, con la llegada de la web 3.0 y la necesidad de interactuar naturalmente con sistemas informáticos, varios buscadores utilizan web semántica, una manera de analizar el contenido web utilizando inteligencia artificial. Estos sistemas utilizan una red de conocimiento amplia y permiten que el buscador interprete el significado de la pregunta e incluso puedan entender contenido implícito y buscarlo (Frosini, 2017).

A este grupo de sistemas se les denomina sistemas de pregunta-respuesta o QAS (*Question Answering Systems*) por sus siglas en inglés. Reciben como entrada una pregunta formulada por el usuario y la respuesta se devuelve en lenguaje natural. Esto permite que el usuario reciba la solución a la interrogante que se está buscando (Chávez, 2016).

Varios buscadores emplean en parte este tipo de sistema como es el caso de Google. Dando la opción al usuario de manejar una comunicación más natural con el sistema informático, debido a que se entiende mejor la respuesta y se determina la solución a la interrogante (Kadambi, 2016). Algunos buscadores internacionales no indexan todo el contenido web cubano. Los sitios no son clasificados como relevantes y por ende quedan mal posicionados en la búsqueda o simplemente no aparecen. Muchos de los buscadores internacionales responden a leyes de otros países y en general bloquean parte de sus servicios a Cuba. También se evidencia una manipulación en los documentos haciendo relevancia en la información que responde a los intereses de grandes compañías (Rodríguez, 2016).

Como parte del proceso de informatización de la sociedad cubana, la Universidad de las Ciencias Informáticas (UCI) está encargada de desarrollar la plataforma C.U.B.A. (Contenidos Unificados para la Búsqueda Avanzada), basada en tecnologías del buscador Orión[2] para Cuba. Esta plataforma presenta una serie de problemas que hacen que la precisión de los resultados mostrados al usuario no tenga una adecuada calidad. Uno de los problemas que repercuten en los resultados es la incapacidad que tiene el sistema de no interpretar adecuadamente las necesidades de información de un usuario a partir de una pregunta. Además, no es capaz de analizar semánticamente la consulta debido a que su modelo de recuperación de información se basa en el análisis sintáctico de palabras clave. Está imposibilitado de proporcionar en tiempo real la respuesta a una pregunta, aunque su respuesta no se encuentre dentro de

[2] Motor de búsqueda desarrollado en Cuba en el 2010 (Nieto, 2010)

su índice. Por otra parte, el sistema no otorga respuestas en lenguaje natural a preguntas también en lenguaje natural. En ocasiones las respuestas a las preguntas están distribuidas a lo largo de un documento, o incluso entre múltiples documentos de la colección; sin embargo, no existe una fusión coherente y exacta de la respuesta, obligando al usuario a revisar documento por documento. Todo esto conlleva a que un usuario que interactúe con la plataforma resulte insatisfecho, se demore en encontrar la respuesta, no encuentre lo que está buscando o encuentre otra información ajena a la consulta.

Considerando la situación problemática anteriormente descrita, se plantea como problema de la investigación: **¿Cómo contribuir a mejorar la precisión de los resultados de las búsquedas en la plataforma C.U.B.A.?** Para la realización de la investigación se define como **objeto de estudio**: el proceso de pregunta y respuesta exacta; enfocado en el proceso de pregunta y respuesta exacta en buscadores web como **campo de acción**.

Para dar cumplimiento al problema planteado anteriormente se determina el siguiente **objetivo general**: Desarrollar un subsistema de respuesta exacta, para contribuir a mejorar la precisión de los resultados de búsqueda en la plataforma C.U.B.A..

Con el propósito de cumplimentar gradualmente el objetivo general antes mencionado, el mismo se ha desglosado en los siguientes **objetivos específicos:**

1. Caracterizar los fundamentos teóricos relacionados con los sistemas de respuesta exacta que utilizan anotación semántica.
2. Definir las tecnologías, las herramientas y la metodología de desarrollo para la implementación del subsistema de respuesta exacta para la plataforma C.U.B.A..
3. Diseñar el sistema de respuesta exacta para la plataforma C.U.B.A..
4. Implementar el sistema de respuesta exacta para la plataforma C.U.B.A..
5. Validar la solución propuesta.

Luego de haber tratado los elementos fundamentales del área de la ciencia a incidir y los objetivos primordiales, se formula la siguiente **hipótesis de la investigación**: El desarrollo de un componente de respuesta exacta, con tecnologías de web semántica, contribuirá a mejorar la precisión de los resultados de búsqueda en la plataforma C.U.B.A.. Teniendo en cuenta la hipótesis formulada se define la siguiente **variable independiente**: Componente de respuesta exacta. Como **variable dependiente**: mejorar la

precisión de los resultados de búsqueda. Esta variable mide el grado de precisión de los resultados en las consultas que brinda el buscador cubano según la pregunta realizada en Alta, Media y Baja.

En el desarrollo de la investigación se utilizarán los siguientes métodos de investigación:

Métodos teóricos:

- **Histórico – Lógico:** Se utilizará con el objetivo de constatar cómo, a lo largo del tiempo, han evolucionado los SPR, los sistemas basados en recuperación de información, así como las herramientas y tecnologías utilizadas.

- **Analítico – Sintético:** Se utilizará con el objetivo de analizar las teorías, documentos y otros elementos referentes relacionados con los SPR y los sistemas de recuperación de información.

Métodos empíricos:

- **Modelación:** Se observará utilizado en la representación, mediante el uso de diagramas, de las características del sistema, relaciones entre objetos; y las actividades que intervienen en los procesos implementados por el subsistema para determinar respuesta exacta.

Estructura del documento:

Capítulo 1: Fundamentación teórica del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.: En este capítulo se expondrán un conjunto de conceptos y criterios fundamentales asociados al objeto de estudio de la investigación. Además, se realizará una comparación entre los principales sistemas basados en recuperación de información. Finalmente, se expondrán todas las tecnologías a utilizar en el desarrollo del componente, así como la metodología de desarrollo seleccionada para guiar el proceso de construcción de la solución que se propone.

Capítulo 2: Análisis y diseño del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.: Se exponen las características del sistema, incluyendo los requisitos funcionales y no funcionales, patrones de diseño y arquitectura utilizados; además, algunos de los artefactos que requiere la metodología de desarrollo utilizada.

Capítulo 3: Implementación y validación del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.: Se exponen algunos aspectos asociados con la implementación de la solución informática, así como los componentes que la integran. Además, se presentan los diseños de casos de

prueba a utilizar en la validación del sistema y se analizan los resultados de las pruebas realizadas. También se muestran los resultados de la validación de la hipótesis como método que sustentará el objetivo principal de esta investigación.

Capítulo 1: Fundamentación teórica del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.

En este capítulo, con el objetivo de alcanzar una mejor comprensión de la investigación que se presenta, se realiza un análisis bibliográfico acerca de los elementos y áreas del conocimiento que engloban al objeto de estudio y al campo de acción. Además, se hace un análisis y evaluación de los principales SPR. Se exponen las características esenciales de las herramientas por la cuales llevaron a su elección, así como las tecnologías y la metodología de desarrollo de software que se utilizarán para la implementación de la solución.

1.1 Fundamentos teóricos asociados al dominio del problema

Sistemas de pregunta-respuesta (SPR)

Los sistemas de pregunta-respuesta (SPR), también conocidos como sistemas de "Búsqueda de Respuestas" (sistemas BR), y mucho más conocidos por su término inglés *Question-Answering Systems* (QAS), son un tipo de mecanismo de recuperación de información insertados generalmente en buscadores, lo que no responden a la búsqueda de documentos que contienen la respuesta a una pregunta, sino que la respuesta en sí, generalmente un hecho (Alupului, 2018).

Las capacidades útiles e importantes y las expectativas que espera un usuario final en un sistema de pregunta-respuesta son principalmente las siguientes (Barco, 2007).

• **Precisión**: Lo más importante en un SPR es la precisión de la respuesta (Barco, 2007). Se considera que una respuesta incorrecta representa mayor inconformidad para el usuario que no responder (Unger, 2014).

• **Respuesta en tiempo corto**: La respuesta a una pregunta se debe proporcionar en tiempo real, aunque exista un nivel elevado de concurrencia en el sistema (Costa, 2017).

• **Usabilidad**: A menudo, hay que tratar con fuentes de información heterogéneas, por ejemplo: textos, bases de datos, video clips o cualquier otro formato multimedia. Los nuevos datos deberán incorporarse al sistema tan pronto como se encuentren disponibles incluso en el caso de hechos y eventos muy recientes (Hoffner, 2017).

• **Completitud**: Se espera que el sistema del conocimiento total que puede encontrar, lo fusione, para

responder con una solución concisa y que se adapte a la necesidad del usuario (Hoffner, 2017).

• **Relevancia**: La respuesta a una pregunta del usuario debe ser relevante en un contexto específico. La complejidad de la pregunta y la taxonomía de preguntas relacionada no pueden ser estudiadas sin tener en cuenta la representación del contexto (Sun, 2015; Hoffner, 2017).

Las aplicaciones de los SPR son diversas y existe una gran variedad tanto del problema a tratar como de los agentes que intervienen: diferentes tipos de usuario, formatos de datos, clases de dominio, entre otros. (Magnini, 2005). Estos sistemas pueden clasificarse por varios parámetros entre los que se encuentran:

• **Atendiendo al tipo de acceso a la información se puede encontrar SPR que buscan sobre**: Datos estructurados (bases de datos), datos semi-estructurados (XML, estructuras de texto en bases de datos), texto libre o combinación de todos ellos.

• **Atendiendo al tipo de colección sobre la que se busca**: La Web, colección de documentos o un texto simple.

• **Atendiendo al tipo de dominio**: Dominio libre o dominios específicos (alta precisión).

• Atendiendo al modo en el que se presenta la información: Texto, imágenes o datos hablados o videos.

Cuando un usuario requiere saber una información específica acerca de algo en Internet por lo general redacta la pregunta en lenguaje natural y el sistema realiza el proceso de búsqueda en su base de datos en un lenguaje de consulta determinado, lo cual lleva al estudio del procesamiento del lenguaje natural.

Procesamiento del lenguaje natural

El procesamiento del lenguaje natural en un buscador o NLP por sus siglas en inglés (*Natural Language Process*) es el proceso de traducir una pregunta expresada en lenguaje natural por un usuario a un lenguaje de consulta (Zhang, 2014).

En su primer paso se encuentra el pre-procesamiento. Esto incluye la normalización del texto eliminando palabras no deseadas. Si la sentencia tiene palabras innecesarias o irrelevantes como artículos, entonces se eliminan, por otra parte, el procesamiento agregará palabras al corpus que se necesitan para ser determinada la sentencia. Este resultado se le devuelve al analizador, este organiza el contenido en un árbol;

de manera que pueda realizar una búsqueda de conceptos en un grafo de ontologías (Kulkarni, 2014).

De esta manera, se organizan los conceptos y acciones en tripletas[3] de lenguaje de consultas para después ejecutar la búsqueda.

Lenguaje de consulta SPARQL

Según el consorcio internacional que genera estándares y recomendaciones en la World Wide Web (W3C) (Koivunen, 2015) como lenguaje de consulta para modelo de datos en forma de grafos RDF, se utiliza el lenguaje de consultas SPARQL (por sus siglas recursivas en inglés *SPARQL Protocol and RDF Query Language*). Es comparable casi siempre con SQL y muy usado en el procesamiento de consulta y en sistemas de inferencia. Con este lenguaje se puede consultar una tripleta dividida en partes. SPARQL sostiene restricción de consultas, emparejamiento o *matching*[4] de patrones arbitrarios, patrones opcionales de grafos en el procesamiento de conjunciones y disyunciones. También se puede realizar una limitación de expresiones regulares por la palabra “FILTER” (Nagdeve, 2016).

Python

Es un lenguaje de programación creado en 1991. Lenguaje con soporte para multitarea, de alto rendimiento, portable, seguro, robusto, interpretado, distribuido, simple y orientado a objetos. Fue inicialmente diseñado por Guido van Rossum y actualmente en nuestros días bajo el mando de Python Software Foundation. Es un lenguaje multiplataforma de licencia publica de GNU, presenta una licencia de código abierto. Presenta extensas librerías que muchas pueden descargarse, plataformas para el desarrollo Web, gestor de paquetes, plataformas para el desarrollo de gráficos y sistemas de gestión de contenidos (Chollet, 2017).

Marco de Descripción de Recurso

Según la W3C, define como método general para la descripción conceptual o modelado de la información que se implementa en los recursos web al Marco de Descripción de Recursos (RDF por sus siglas en inglés, *Resource Description Framework*) (World Wide Web Consortium, 2014).

Ontología

Una ontología es el conjunto de palabras y conceptos usados para describir y para representar un área del

[3] Conjunto de elementos en una consulta semántica que presenta sujeto, predicado y objeto (Sulé, 2016)
[4] En programación, hacer coincidir dos objetos de una misma clase (Emir, 2007)

conocimiento. Donde se representan objetos, reglas, propiedades, relaciones, funciones y restricciones entre ellos. El objetivo que se quiere con las ontologías es conceptualizar el conocimiento de un dominio especifico y su objetivo principal es el de proveer un entendimiento común y compartido entrelazándolos. Otro objetivo que se persigue es la interoperabilidad entre las personas y otros sistemas (Kang, 2014). De otra manera se puede decir que busca representar el conocimiento de un dominio específico para que pueda ser usado en diferentes conceptos.

Web semántica

Termino que fue introducido por la organización W3C con tendencia a la creación de tecnologías para publicar datos legibles por aplicaciones informáticas. Se basa principalmente en añadir metadatos semánticos y ontológicos a la Web que describen el significado, contenido y la relación de la información de manera formal. De esta manera los sistemas informáticos y las personas podrán acceder a una información de manera natural (Blomqvist, 2014).

Bases de conocimiento

Las bases de conocimiento son un tipo especial de base de datos para la gestión del conocimiento. Provee herramientas para la recolección, organización y recuperación informatizada de conocimiento. Las bases de conocimiento pueden ser utilizadas tanto por máquinas como por el mismo hombre. Un grupo de instancias de las clases de una ontología constituye una base de conocimiento (Uribe, 2017).

1.2 Estudio de sistemas existentes que implementan respuesta exacta

En la actualidad existen en el mundo diversas aplicaciones web que permiten realizar respuestas exactas, atendiendo a la pregunta formulada por el usuario. Algunas de las más utilizadas son Google Knowledge Graph Question Answering System, IBM Watson, Wolfram Alpha, entre otros menos conocidos (Pelikánová, 2014). A continuación, se realiza un estudio a estos para lograr una mejor compresión de las características de los sistemas homólogos tanto en el ámbito internacional como nacional.

Google Knowledge Graph Question Answering System

Es el sistema de pregunta-respuesta de Google que utiliza una base de conocimiento que el buscador Google predeterminó en mayo de 2012 con el nombre de Google Knowledge Graph. El objetivo de este sistema es mejorar la búsqueda, eliminar las expresiones ambiguas y permitir a los usuarios ampliar sus conocimientos sobre temas relacionados o relacionarlos con otros. La base de datos que utiliza para realizar

la búsqueda semántica es privada de Google, lo que provoca que no se pueda tener acceso a ella por otro buscador o persona. La forma que Google quiere lograr con esta mejora, a diferencia del concepto clásico de búsqueda cuando las palabras clave solo se entienden como cadenas de caracteres, ahora se crea un cuadro extenso (Pelikánová, 2014).

Este sistema recrea lo que se denominaría como sistema de pregunta-respuesta, incluyendo opciones para cálculo matemático, conversión de valores, contenido multimedia variado referente a la búsqueda, entre otros. Sin embargo, la parte más visible de Knowledge Graph son los paneles laterales, que también se puede llamar Knowledge Boards. Estos se colocan en la página de resultados siempre a la derecha. En general: esbozarán una definición breve, y adjuntarán imágenes y otra información que difieren según el tema. La desventaja de estos paneles puede ser el hecho de que la mayoría de la información proviene de Wikipedia o Freebase, independientemente de si estas son realmente las fuentes más apropiadas para los temas. Sin embargo, es la solución más simple y más barata, y los paneles solo deben usarse para una definición breve (Pelikánová, 2014).

IBM Watson

El superordenador creado por IBM contiene un sistema informático de pregunta-respuesta que utiliza inteligencia artificial para responder preguntas elaboradas en lenguaje natural. Forma parte de una tecnología novedosa para el desarrollo de los SPR denominado DeepQA (Pelikánová, 2014). El SPR de Watson requiere de una base de datos interna para responder las preguntas, contenido extraído de muchas fuentes como lo son enciclopedias, diccionarios y bases de datos de conocimiento como DBpedia[5], y WordNet[6] (Chen, 2016).

Watson, una tecnología de computación cognitiva, se ha configurado para respaldar la investigación en ciencias de la vida. Watson también se ha desarrollado con una comprensión específica de la terminología científica para que pueda establecer conexiones novedosas en millones de páginas de texto. Watson se ha aplicado a algunos estudios piloto en las áreas de identificación de objetivos de drogas y readaptación de medicamentos. Los resultados piloto sugieren que Watson puede acelerar la identificación de nuevos fármacos candidatos y nuevos objetivos farmacológicos mediante el aprovechamiento del potencial del Big

[5] Proyecto para el desarrollo de extracción de Wikipedia (Witbrock, 2015)
[6] Base de datos léxica que agrupa palabras en inglés en conjunto de sinónimos (Yao, 2014)

Data (Chen, 2016).

Wolfram Alpha

Es un sistema de recuperación de información solamente para la búsqueda de respuestas en lenguaje natural. Es de propósito comercial y fue desarrollado en mayo de 2009 por la compañía Wolfram Research. Es un servicio en línea que responde preguntas mediante el procesamiento de la respuesta extraída de una base de datos estructurada en vez de procesar documentos. Wolfram incorpora capacidades de procesamiento matemático y algebraico, así como también el procesamiento simbólico y de visualizaciones (Abramovich, 2015).

Plataforma cubana de búsqueda

Plataforma C.U.B.A., (por sus siglas en español: Contenidos Unificados para Búsqueda Avanzada), es una plataforma que brinda acceso a los contenidos publicados en la red cubana. Esta logra que dicha información sea accedida por los usuarios y responde a las búsquedas realizadas por ellos emitiendo un listado de los sitios que coinciden con los criterios introducidos. C.U.B.A., está basada en la tecnología del buscador Orión, desarrollado en el centro de Ideoinformática de la Universidad de las Ciencias Informáticas. Tiene como objetivo principal el brindar al usuario cubano contenido confiable y de interés cultural, informativo e investigativo de origen nacional. Permite recuperar información de varios tipos de formato, sea una página Web, una imagen o un documento (CubaRed, 2018).

Esta plataforma no presenta un sistema de pregunta-respuesta integrado y tampoco existe un sistema de recuperación de información cubano basado solamente en pregunta-respuesta.

Resultados del estudio de los sistemas similares

El estudio realizado sobre los sistemas de recuperación de información, tanto en el ámbito nacional como internacional, arrojó los siguientes resultados:

- Los sistemas de recuperación de información estudiados no pueden ser utilizados para ofrecer solución al problema planteado. Esto se debe, en el caso de los internacionales, a que su código fuente es privativo. Estos presentan características específicas diferentes a las necesidades para la cual se requiere realizar un sistema de pregunta-respuesta en Cuba. Un ejemplo de esto puede ser la mala clasificación, el mal posicionamiento o la omisión de la información cubana en internet.

- Por otra parte, la plataforma cubana no cuenta con un mecanismo que permita determinar una respuesta exacta y no se puede utilizar para darle solución al problema científico planteado.

Debido a que estos sistemas no pueden ser usado como parte de la solución al problema planteado en esta investigación, se decidió desarrollar un sistema para determinar respuesta exacta como propuesta de solución al problema planteado. No obstante, el estudio de las diferentes bases de conocimiento que utilizan estos sistemas homólogos internacionales, permitió definir qué base de conocimiento se utilizan más para este tipo de sistema, como es el caso de DBpedia, Freebase y Cyc. También permitió visualizar los SPR para determinar sus principales características, como el formato de la información que reciben y la que devuelven. Esta información fue utilizada para el desarrollo de la propuesta de solución.

1.3 Herramientas, lenguajes y tecnologías

Para desarrollar la propuesta de solución, surge la necesidad de estudiar varias herramientas, lenguajes y tecnologías disponibles para dar cumplimiento al objetivo general planteado.

1.3.1 Bases de conocimiento existentes para representar contenido en bases de conocimiento:

Freebase

Freebase es un proyecto iniciado en marzo de 2007 que no se le brinda soporte desde marzo del 2015, pero su base de datos continúa activa. Actualmente solo contiene un solo idioma, el inglés. Se trata de una colección en línea de datos estructurados o metadatos creados principalmente por miembros de su comunidad, incluidos principalmente de contribuciones wiki entre otras múltiples fuentes. Los datos de Freebase son para uso libre, comercial y no comercial bajo una licencia libre, y para los programadores de agrega una API[7] abierta y un punto final de RDF y descarga de bases de datos (Pelikánová, 2014).

Cyc

Es un proyecto iniciado en 1984 por Doug Lenat, inicialmente software propietario pero una gran parte bajo una licencia de código abierto y hoy se conoce como OpenCyc[2]. Su principal propósito es intentar ensamblar una ontología comprensiva y una base de conocimiento general con el fin de que tanto humanos como máquinas puedan hacer consultas a su base de conocimiento. La Base de datos actualmente contiene aproximadamente 100 000 conceptos y 1 000 000 afirmaciones o relaciones entre conceptos, las

[7] Interfaz de programación de aplicaciones (Aggarwal, 2014)

declaraciones de esta se encuentran en lenguaje CycL[8], basado en cálculos de predicados de primer orden con otros de nivel más alto (Witbrock, 2015).

DBpedia

Es una organización no comercial que responde bajo las licencias de GNU[9] de software libre, es un proyecto desplegado el 10 de enero de 2007 para la extracción de datos de Wikipedia, cuenta con 111 idiomas, disponibles en inglés, español, entre otros. En la base de datos en total se escriben más de 3,7 millones de entidades entre ellas relaciones de imágenes, videos, música, entre otros. El tipo de lenguaje de consulta que emplea para acceder a su base de datos es SPARQL y se almacena en grafos con formato RDF (Lehmann, 2015).

En la tabla 1 se establece la comparación entre estas principales bases de conocimiento en cuanto a características principales de estas.

Tabla 1: Comparación entre bases de conocimiento (Yao, 2014; Pelitakova, 2014; Witbrock, 2015; García, 2016; Lehmann, 2015).

Características	**Freebase**	**Cyc**	**DBpedia**
Lenguaje de consulta	MQL[10]	CycL	SPARQL
Formato de base de datos de conocimiento	RDF y formato privado de Metaweb	CycL	RDF
Idiomas de la BD	Inglés con algunas traducciones a otros lenguajes	Inglés	111 incluyendo español
Software libre o propietario	Parte de la base de datos a código abierto Licencia Creative Commons	Software propietario, pero cuenta con una biblioteca abierta OpenCyc con menor	Si, en su totalidad

[8] Lenguaje de consulta creado por CycCorp (Witbrock, 2015)
[9] Proyecto colaborativo de software libre (Stallman, 2015)
[10] Lenguaje de consulta para búsqueda de subestructuras permitiendo propiedades nominales y numéricas (Yao, 2014)

	(CC)	contenido	
Contenido Disponible	46 millones de registros y 2.7 mil millones de hechos	230,000 términos, y 15,000 predicados	3.77 millones de entidades en total
Origen de bases de datos	Wikipedia y una parte creada manualmente por la comunidad	Propietaria	Wikipedia

Resultados del estudio de las bases de conocimiento

Debido a que Freebase es un software que todavía no contiene una base de datos en idioma español completamente, y a que Cyc es una herramienta privativa y no reúne las características para el sistema que se quiere implementar. El autor asume el empleo de DBpedia como la base de conocimiento que más se ajusta para la propuesta de solución que se pretende obtener como resultado de la investigación. Además, el lenguaje de consulta SPARQL y el formato RDF son tecnologías abiertas de libre acceso, y se adaptan fácilmente a la plataforma C.U.B.A..

1.3.2 Lenguajes existentes para realizar consultas en base de datos ontológicas

CycL

CycL es un lenguaje expresivo que soporta la ontología de OpenCyc del proyecto Cyc. El vocabulario de definiciones usado para expresar la información taxonómica en OpenCyc refleja muchas de las expresiones necesarias y las cuestiones encontradas durante el proceso de construir una base de datos de conocimiento Cyc (García, 2016).

En la Tabla 2 se muestra una ligera comparación para medir los lenguajes en cuanto a sus principales características:

Tabla 2: Comparación entre lenguajes de consulta (García, 2016; Lehmann, 2015)

Características	CycL	SPARQL
Bases de conocimiento que soportan	Cyc, OpenCyc database	DBpedia
Plataformas compatibles	Privadas en el caso de Cyc	Apache Jena, Java EE, Python

Resultados del estudio de lenguajes de consulta para bases de conocimiento

SPARQL soporta tecnologías de libre acceso para el centro de Ideoinformática de la UCI, como es el caso de los lenguajes de programación Java y Python. En el caso de CycL, es un lenguaje de consulta privado, pero se puede utilizar en una base de datos abierta como OpenCyc con el objetivo de dar soporte a la web semántica, pero no con fines lucrativos (García, 2016). El autor decide emplear el lenguaje SPARQL para la implementación de la propuesta de solución debido a que SPARQL representa un lenguaje más completo y libre en su totalidad, además de que DBpedia presenta soporte solamente para lenguaje de este tipo.

1.3.3 Herramientas para el análisis semántico y generación de la consulta SPARQL

TextGrocery

Herramienta para el procesamiento del lenguaje natural en el lenguaje de programación C++. Es utilizada generalmente para cortos volúmenes de texto y contiene una la librería llamada jibea para la tokenización[11]. Es bastante usada y su calidad de código es de nivel 1 (Lumnify, 2018).

spaCy

Es una biblioteca para el procesamiento avanzado del lenguaje natural para Python. Es basado en últimas investigaciones del procesamiento del lenguaje natural, pero no es un software que esté actualmente recibiendo soporte. Fue diseñado desde los primeros días para ser utilizado en productos reales. Es un software comercial de código abierto, publicado bajo una licencia libre. Está programado en HTML[12] y es bastante usada (Libhunt, 2018).

[11] Proceso de un compilador el cual se convierten las palabras en tokens o símbolos (Addati, 2017)
[12] Hace referencia al lenguaje de marcado para la elaboración de páginas web (Beati, 2015)

Apache OpenNLP

Este proyecto maneja trabajos comunes del NLP. Consiste en una sucesión de elementos que cumplen tareas específicas, accediendo a la formación de modelos y el soporte para pruebas de estos modelos. El camino frecuente utilizado por OpenNLP, es instanciar un modelo que soporta la ejecución desde un archivo y luego ejecuta métodos contra el modelo para realizar una tarea (Reese, 2015).

NLTK

Natural Language Toolkit, o más comúnmente NLTK, es un conjunto de bibliotecas y programas para el procesamiento simbólico y estadístico para el procesamiento del lenguaje natural escrito en el lenguaje de programación Python. Fue desarrollado por Steven Bird y Edward Loper en el Departamento de Computación e Informática de la Universidad de Pensilvania en Estados Unidos. NLTK incluye demostraciones gráficas y datos de muestra. Se acompaña de un libro que explica los conceptos subyacentes detrás de las tareas de procesamiento de lenguaje compatibles con el kit de herramientas (Bird, 2004).

NLTK está destinado a apoyar la investigación y la enseñanza en NLP o áreas estrechamente relacionadas, incluida la lingüística empírica, la ciencia cognitiva, la inteligencia artificial, la recuperación de información y el aprendizaje automático. NLTK se ha utilizado con éxito como una herramienta de enseñanza, como una herramienta de estudio individual, y como una plataforma para la creación de prototipos y la construcción de sistemas de investigación. Hay 32 universidades en los Estados Unidos y 25 países que usan NLTK en sus cursos. NLTK admite funciones de clasificación, tokenización, derivación, etiquetado, análisis sintáctico y semántica (Almohaimeed, 2017).

Stanford CoreNLP

Es un conjunto de materiales para las tareas del NLP calificado por el Grupo de NLP de la Universidad de Stanford en los Estados Unidos. Entre los idiomas que admite, se pueden encontrar el inglés, el chino y el español. Este mecanismo trabaja mediante la licencia GPL, pero no permite ser usada en aplicaciones comerciales, aunque existe una licencia de carácter comercial (Reese, 2015).

Quepy

Quepy es un marco de trabajo para transformar preguntas formuladas naturalmente a lenguaje de consulta semántico. Utiliza la librería NLTK para realizar el procesamiento del lenguaje natural y utilizar conjuntos de

entrenamiento para el etiquetado de entidades nombradas. Se puede personalizar fácilmente para diferentes tipos de preguntas en lenguaje natural y consultas de bases de datos. Con poca codificación se puede construir un sistema para el acceso de lenguaje natural a la base de datos. Actualmente Quepy brinda soporte para los lenguajes de consulta SPARQL y MQL (Bansal, 2014).

Quepy es utilizada con frecuencia en el mundo y cuenta con un nivel de calidad de código de nivel 5 según Lumnify (2018). Además, ofrece una herramienta para el procesamiento del lenguaje natural, con un modelo de plantillas para generar preguntas que facilitan el trabajo.

Resultados del estudio de las herramientas

Aunque las herramientas antes presentadas son libres, de código abierto y son APIs para la plataforma Java y Python, el autor determina que la herramienta a utilizar para el procesamiento del lenguaje natural y la formulación de la consulta SPARQL será Quepy. Se escoge la misma debido a que esta ofrece una estructura organizada en plantillas, que concederá al desarrollador facilidad en la implementación del subsistema y presenta una alta calificación de calidad de código con respecto a otras herramientas. También se selecciona porque utiliza la librería NLTK y se puede utilizar con un conjunto de entrenamiento de Stanford CoreNLP para el análisis de la pregunta, pues presenta modelos de soporte en varios idiomas, entre ellos el español, además de ser una herramienta usada con frecuencia.

1.3.4 Lenguaje de programación de la aplicación

El entorno de trabajo para realizar el procesamiento del lenguaje natural y formular la consulta SPARQL, Quepy, tiene soporte solamente para lenguaje de programación Python.

1.3.5 Entornos de trabajo para el desarrollo de aplicaciones Web en Python

Flask

Flask es un micro-entorno de trabajo minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código. La aplicación “*Hello World*” de Flask es la más simple, registrando 7 líneas de código en un único archivo de Python. El "micro" en micro-entorno significa que Flask tiene como objetivo mantener el núcleo simple pero extensible (Aggarwal, 2014).

Django

Django es un entorno de trabajo para el desarrollo web de código abierto, que respeta el patrón de diseño conocido como Modelo–Vista–Controlador, aunque los desarrolladores le suelen llamar Modelo-Plantilla-

Vista. La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis en el re-uso, la conectividad y extensibilidad de componentes y el desarrollo rápido. La herramienta es multiplataforma y contiene una licencia libre. Django es un amplio sistema con una serie de características que lo hacen ser un entorno de trabajo completo. Entre estas características se destacan tener una API para la base de datos, un sistema extensible de URLs basado en expresiones regulares, soporte de internacionalización, incluyendo traducciones incorporadas de la interfaz de administración y un sistema incorporado de "vistas genéricas" que ahorra tener que escribir la lógica de ciertas tareas comunes (Holovaty, 2009).

Pyramid

Pyramid es un entorno de trabajo para el desarrollo de aplicaciones Web en Python. Está diseñado para desarrollar aplicaciones web de una manera sencilla. Funciona en todas las versiones compatibles de Python y es de código libre. Tiene una gran cantidad de recursos útiles para dibujar. Extending Pyramid es una lista organizada y filtrable de complementos, paquetes y aplicaciones creadas para funcionar con Pyramid. Cuando la aplicación a desarrollar crece, Pyramid ofrece características que hace menos dificultosa la escritura de software complejo. La aplicación *"Hello World"* de Pyramid es relativamente corta con 9 líneas de código (Jolion, 2012).

Resultados del estudio de los entornos de trabajo para el desarrollo Web

El subsistema a implementar requiere solamente recibir y responder una petición HTTP[13] por la cual se enviará la pregunta en lenguaje natural utilizando una API RESTful[14], por lo que no requiere un entorno de desarrollo Web robusto. Para una mejor compilación, orden en el código de trabajo y facilidad de implementación, se define la utilización de un entorno minimalista que cumpla con los requisitos necesarios para el desarrollo de la propuesta de solución. A partir de estas necesidades, se escoge Flask como entorno de trabajo para desarrollo de la propuesta de solución.

1.3.6 Metodología de desarrollo

Para mejorar la organización en el proceso productivo del software se selecciona una metodología de desarrollo, la cual constituirá un mecanismo de trabajo que proporcionará una base de procesos para llevar

[13] Protocolo de transferencia de hipertexto (Warren, 2015)
[14] Constituye una opción de estilo de uso de los Servicios Web emulando al protocolo Hypertext Transfer Protocol o protocolos similares (Navarro, 2017)

a cabo este proyecto informático. Esta brindará un soporte a la toma de decisiones en el equipo de trabajo del proyecto asignando las tareas respectivamente (Pressman, 2005).

AUP-UCI

Según Meriño (2015) es una modificación de la metodología diseñada por la UCI referente a la metodología AUP *(Agile Unified Process)* por sus siglas en inglés. Fue creada con el objetivo de adaptarse al ciclo de vida definido para la actividad productiva de la universidad. Describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles.

Define tres fases fundamentales para componer el ciclo de vida de un proyecto:

- Inicio, donde se mantiene la fase que tiene AUP, pero cambiando los objetivos a actividades más relacionadas con el proyecto.
- Ejecución donde se unifican tres restantes fases que propone AUP. Se ejecutan las actividades requeridas para obtener el software.
- Cierre, donde esta fase se incorpora a la metodología en relación con AUP. Se analizan los resultados del proyecto, su ejecución y se llevan a cabo actividades consecuentes de cierre del proyecto.

1.3.7 Herramientas

Entorno de desarrollo Integrado

Para el desarrollo de la propuesta de solución se hace necesario estudiar diferentes herramientas que ayuden en la implementación del subsistema. Dentro de estas herramientas está el caso de los entornos de desarrollo integrado (IDE), los cuales son aplicaciones informáticas que proporcionan servicios integrales para facilitarle al desarrollador o programador el desarrollo del software (Ujhelyi, 2015).

Eclipse PyDev

Eclipse es plataforma de software que cuenta con un entorno de desarrollo integrado. Este IDE fue creado por IBM en lenguaje de programación Java y cuenta con una licencia pública de Eclipse. El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés *plug-ins*) para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse

usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos. (Chen, 2005)

El módulo para programar en Python se llama PyDev: es libre de costo y está lleno de características poderosas para programar de manera eficiente en Python. Es un módulo de código abierto y se ejecuta en Eclipse en cualquiera de sus versiones. PyDev contiene integración con Django y Flask, completa el código de manera automática, presenta soporte multilenguaje, plantillas de código, análisis de código, marcado de errores y mucho más. Se mantiene siempre actualizado y contiene una gran comunidad de usuarios y empresas de patrocinio como Liclipse, Squish, TraceTronic entre otras. Aunque PyDev califica como uno de los mejores IDE de Python de código abierto, también viene empaquetado junto con otro producto llamado Liclipse, un producto comercial construido sobre Eclipse que proporciona mejoras en la usabilidad y temas adicionales (PyDev-Python, 2018).

PyCharm

PyCharm es un entorno de desarrollo integrado (IDE) utilizado en la programación de computadoras, específicamente para el lenguaje Python. Está desarrollado por la compañía checa JetBrains. Proporciona análisis de código, un depurador gráfico, un probador de unidades integrado, integración con sistemas de control de versiones (VCS) y es compatible con el desarrollo web con Django (Sarda, 2017).

PyCharm es multiplataforma, con versiones de Windows, macOS y Linux. La edición de la comunidad se publica bajo la Licencia de Apache, y también hay una edición profesional lanzada bajo una licencia propietaria la cual tiene características adicionales (Heimlich, 2016).

Estas son algunas de las principales características de JetBrains:

- Asistencia y análisis de codificación: finalización de código y sintaxis, resaltado de errores, integración de linter y soluciones rápidas.
- Navegación de proyecto y código: vistas de proyecto especializadas, vistas de estructura de archivos y saltos rápidos entre archivos, clases, métodos y usos.
- Refactorización de Python: incluye cambio de nombre, método de extracción, introducir variable, introducir constante, pull up, push down y otros.
- Soporte para entornos de trabajo web: Django, web2py y Flask.
- Depurador integrado de Python.

- Prueba de unidad integrada, con cobertura de código línea por línea.
- Desarrollo de Google App Engine Python.
- Integración de control de versiones: interfaz de usuario unificada para Mercurial, Git, Subversion, Perforce y CVS con listas de cambios y fusión.

Resultados del estudio de los entornos de desarrollo integrado

El estudio realizado sobre los entornos de desarrollo integrados arrojó que el IDE PyCharm a pesar de que cuenta con una licencia comunitaria y presenta muchas características que lo hacen situarse entre los mejores IDE para programar en Python, no puede ser utilizado con fines lucrativos. Por lo tanto, se escoge a la herramienta Eclipse con el módulo PyDev para el desarrollo de la propuesta de solución, el cual cuenta con una total integración con todos los componentes y herramientas que se escogieron para el desarrollo de la propuesta de solución; además de que posee con una licencia para su libre uso.

Herramienta para el modelado de procesos de software

Para agilizar la realización de determinadas actividades del proceso de desarrollo del software se describirá la herramienta CASE[15] seleccionada.

Visual Paradigm

Visual Paradigm es una herramienta CASE (*Computer Aided Software Engineering* o Ingeniería de Software Asistida por Computadora) es una aplicación informática destinada a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y dinero. También ayuda a mejorar la evolución del usuario en su proyecto mediante el diseño de un gran número de artefactos para la ingeniería de software. Permite la representación de bases de datos, conversión de diagramas, mapeos de entidades y objetos, gestión de requisitos y modelación de diagramas de negocio entre otras opciones (Kefalakis, 2015).

SPARQL endpoint

Al igual que la mayoría de las iniciativas Linked Data, la Biblioteca del Congreso Nacional cuenta con un punto web de consultas SPARQL, herramienta que permite realizar consultas SPARQL sobre un grafo de entrada compuesto por tripletas RDF. A nivel técnico, un endpoint SPARQL implementa una interfaz descrita

[15] Herramienta de Ingeniería de Software Asistida por Computadora para aumentar la productividad de procesos de Software

en la especificación SPROT de W3C que define una operación, un mensaje de entrada y dos mensajes de salida. El mensaje de entrada corresponde a la consulta SPARQL que se desea ejecutar, además del grafo (que es opcional) sobre el cual se ejecuta la consulta (Lehmann, 2015).

La operación permite ejecutar una consulta SPARQL sobre la lógica de aplicación en que existe el endpoint. El primer mensaje de salida corresponde a los resultados obtenidos a partir de la consulta, lo cual se da en el caso de que no existan errores. El segundo mensaje de salida corresponde a mensajes de error en el caso de falla de la consulta (que puede estar causada por errores de sintaxis, semánticos, excepciones en tiempo de ejecución, u otros) (Knuth, 2015; Lehmann, 2015).

1.4 Conclusiones parciales

- El estudio de los conceptos asociados a los SPR y las bases de conocimiento permitió lograr alcanzar un dominio adecuado en cuando a las más usadas en el mundo y cual se ajusta a las necesidades y características de Cuba con el fin de asumir los supuestos teóricos para la presente investigación.

- El análisis de las características y funcionalidades de los SPR utilizados en el mundo permitió definir la base de conocimiento y las características esenciales con las que debe contar el subsistema a implementar.

- La selección de la metodología, herramientas y tecnologías con soporte multiplataforma y basadas en software libre, posibilitó obtener una base tecnológica enfocada en los componentes que utilizan los SPR estudiados.

Capítulo 2: Análisis y diseño del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.

En este capítulo se abordarán aspectos fundamentales relacionados con el diseño del sistema a desarrollar. Entre los elementos a destacar se encuentran el diagrama del modelo del dominio, mediante el cual se representan las clases conceptuales significativas del problema a resolver. Como vía para definir las futuras funcionalidades de la propuesta de solución y qué usuarios podrán tener acceso a las mismas, se generaron los artefactos relacionados a la especificación de los requerimientos funcionales y no funcionales que deberá poseer el subsistema; así como la especificación de los casos de uso. Como parte del diseño de la aplicación se definieron los estilos y patrones de arquitectura que se emplearán para lograr buenas prácticas de diseño y programación. A lo largo del capítulo se mostrarán los principales artefactos de ingeniería de software correspondientes al caso de uso.

2.1 Modelo de dominio

Un modelo del dominio es una representación de las clases conceptuales del mundo real, no de componentes de software. No se trata de un conjunto de diagramas que describen clases de software, u objetos de software con responsabilidades, sino que modela clases conceptuales significativas en un determinado problema (Larman, 2004).

Para lograr un mejor entendimiento de la presente investigación se hace necesario describir el procedimiento de los SPR mediante una serie de conceptos, entidades y sus relaciones, agrupándose en un modelo del dominio con el fin de contribuir a la comprensión del contexto actual del problema.

Diagrama de Clases de Dominio

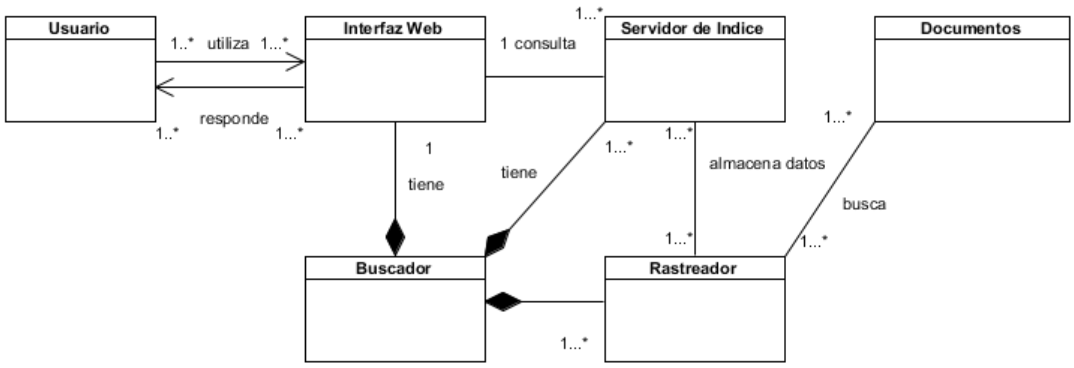


Figura 2: Diagrama de clases del modelo del dominio

La Figura 2 muestra la relación existente entre todos los conceptos que intervienen en la investigación que se presenta. Se puede observar que los documentos que se encuentran en la Web serán primeramente recopilados por el rastreador y seguidamente almacenados en el servidor de índice. Tanto el servidor de índice como el rastreador forman parte del buscador que contiene una interfaz web mediante la cual se le presentan los resultados al usuario.

Descripción de Clases del Modelo de Dominio

La modelación del dominio constituye la herramienta fundamental para garantizar la comprensión y descripción de las clases o conceptos y sus relaciones más importantes dentro del contexto del problema. A continuación, se presenta la descripción de los conceptos identificados por el autor en la presente investigación.

Tabla 3: Descripción de las clases del modelo de dominio

Conceptos	Descripción
Usuario	Persona que realiza las búsquedas.
Buscador	Constituye una herramienta de recuperación de información en la Web.
Base de conocimiento	Base de datos utilizada para representar el conocimiento mediante relaciones.
Herramienta para NLP	Servicio web que se encarga de recibir una pregunta y construir una consulta SPARQL.
Interfaz web del buscador	Constituye la vista mediante la cual se le muestran los resultados al usuario.
Sitio web	Colección de páginas en internet relacionadas y comunes a un dominio de internet o subdominio.

2.2 Estilo Arquitectónico

Un estilo arquitectónico es la organización fundamental de un sistema encarnada en sus componentes, las relaciones de los componentes con cada uno de los otros y con el entorno, y los principios que orientan su diseño y evolución (Reynoso, 2004).

Descripción del subsistema propuesto

El subsistema que se propone desarrollar, pretende mejorar la precisión de los resultados de búsqueda de

la plataforma cubana. Este sistema debe permitir que una vez el usuario de la plataforma C.U.B.A. realice una petición en forma de pregunta y esta la envié al subsistema implementado; se reciba la información mediante un servicio API RESTful por una petición HTTP GET[16].

Una vez recibida la petición se deberá realizar un análisis de la forma que tiene la pregunta, haciendo coincidir entidades nombradas y palabras clave con plantillas que generarán la consulta SPARQL correspondiente al tipo de pregunta. El subsistema deberá reconocer si la consulta no se encuentra en la base de datos del sistema o si se encuentra, pero está obsoleta y necesita ser modificada. Posteriormente se realizará la consulta en la base de conocimiento DBpedia, se normalizarán los datos, se guardarán en la base de datos del sistema y finalmente se devolverán los resultados. En caso de encontrarse la consulta en la base de datos del sistema se devolverán los resultados a la plataforma.

Arquitectura del subsistema propuesto

En este epígrafe se definirán los elementos principales de la propuesta de solución junto con sus características. Así como la descripción del procesamiento del subsistema y la arquitectura en la que se basa.

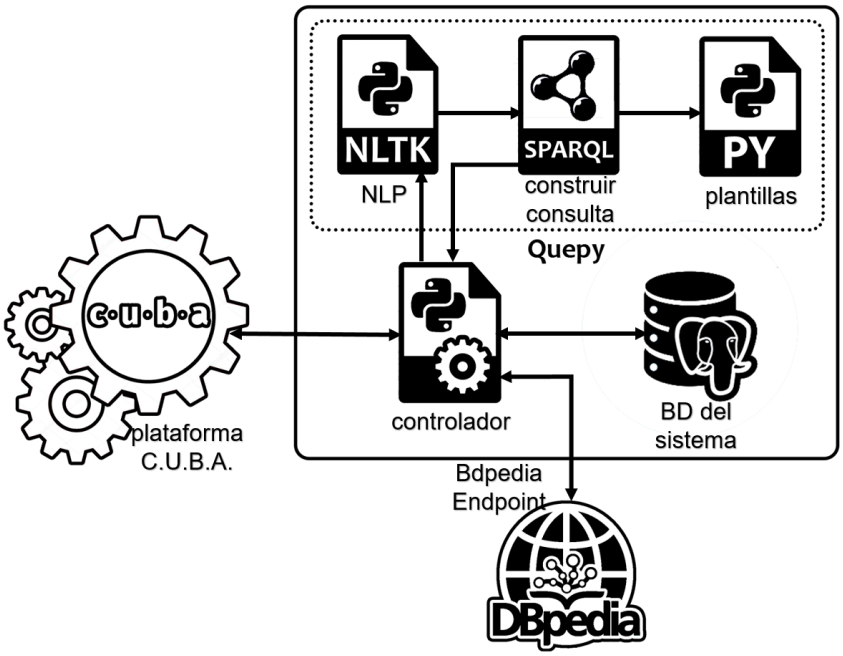


Figura 3: Arquitectura de la propuesta de solución

[16] Petición HTTP que transmite información mediante parámetros de la URL (Watanabe, 2015)

Como se aprecia en la Figura 3, el subsistema implementado en forma de servicio Web, se inicia recibiendo una pregunta formulada en lenguaje natural de la plataforma C.U.B.A. Esta es recibida por el controlador del servicio web, se envía al entorno Quepy, el cual recrea un procesamiento del lenguaje natural utilizando la librería de Python NLTK. Esta librería tokeniza las palabras y le asigna etiquetas gramaticales para después ejecutar una lematización[17]. Para la conversión de lenguaje natural una vez procesado a lenguaje de consulta SPARQL, el subsistema utiliza expresiones regulares y una librería llamada "refo" que proporciona una funcionalidad para secuencias arbitrarias y permite tratarlas en forma de objetos.

Estas expresiones regulares se guardan en formas de plantillas Python (*templates*) para cada tipo de pregunta, ejemplo: ¿Cuáles son…? ¿Dónde es…? Listar países…, entre otras. De esta manera se realiza un análisis utilizando una plantilla determinada para cada forma de la interrogante y se obtiene una consulta en lenguaje SPARQL. Se utiliza la herramienta del servicio SPARQLWrapper para ejecutar la consulta en la base de conocimiento DBpedia. Esta utiliza el endpoint DBpedia para realizar las consultas directas en su base de conocimiento. Una vez conectado al endpoint se ejecuta la consulta y se toman los resultados devueltos en formato JSON. Se normalizan los datos eliminando resultados de la consulta irrelevantes y dándole formato de texto. Se guardan los resultados en la base de datos del sistema y se devuelve el resultado a la plataforma.

El estilo de arquitectura que se utilizó es el *Peer-to-Peer* debido a que el sistema implica el intercambio de información entre computadoras individuales en una red y no hay necesidad de que esta información se almacene o gestione de manera centralizada. El patrón arquitectónico que se refleja es el Orientado a Servicios debido a que la propuesta de solución responde a un enfoque RESTful de intercambio de información mediante el protocolo HTTP (Somerville, 2011).

2.3 Especificación de Requisitos de Software

Un proyecto no puede ser exitoso sin una especificación correcta y exhaustiva de los requerimientos (Mendoza, 2016). Una vez que se han definido los conceptos principales relacionados con el objeto de estudio, se puede comenzar a analizar qué debe hacer la aplicación para que cumpla con los objetivos

[17] Proceso por el cúal se agrupan las formas flexionadas de una palabra para que puedan analizarse como un solo elemento identificado por la forma del diccionario de la palabra (Sánchez, 2017)

planteados al inicio de esta investigación. Para ello, se clasifican a través de requerimientos funcionales y no funcionales, las acciones que el sistema deberá ser capaz de realizar.

2.3.1 Requisitos Funcionales

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer y explican las distintas funciones que presenta (Cañas, 2016). A continuación, se presentan los requisitos funcionales del sistema para determinar respuesta exacta en la plataforma C.U.B.A.

Tabla 4: Requisitos funcionales

Nº	Nombre	Descripción	Prioridad	Complejidad
RF 1	Extraer texto del servicio Web	Extraer texto recibido por el servicio web y enviado por la plataforma	Baja	Baja
RF 2	Extraer tipo de pregunta	Extracción del tipo de pregunta para determinar el prototipo de respuesta que devuelve	Baja	Baja
RF 3	Eliminar palabras irrelevantes	Eliminar palabras irrelevantes que no aportan información al resultado	Baja	Baja
RF 4	Procesar lenguaje natural	Realizar el procesamiento del lenguaje natural y clasificación de palabras en *tokens*.	Alta	Alta
RF 5	Construir consulta semántica	Construir consulta SPARQL en forma de texto	Alta	Alta
RF 6	Ejecutar consulta en el sistema	Realizar consulta en base de datos del sistema	Baja	Media
RF 7	Realizar consulta en la base de conocimiento	Realizar consulta SPARQL en la base de conocimiento	Alta	Alta
RF 8	Recibir información de la base de conocimiento	Recibir la información de la base de conocimiento DBpedia cuando se ejecuta la consulta SPARQL	Alta	Media
RF 9	Realizar filtro a los resultados	Realizar filtro de resultados de la base de conocimiento	Media	Alta
RF 10	Guardar información en base de datos	Guardar información de la consulta en base de datos del sistema	Baja	Media
RF 11	Normalizar los resultados	Normalizar los datos de la información devuelta por la base de conocimiento	Media	Media

RF 12	Devolver los resultados	Devolver los resultados a la plataforma	Alta	Baja

2.3.2 Requisitos no funcionales

Los requisitos no funcionales, como su nombre sugieren, son aquellos requisitos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este (Bolloju, 2017). A continuación, se presentan los requisitos no funcionales del subsistema para determinar respuesta exacta.

Requisitos de Usabilidad:

- **RnF1.** El sistema debe mantener las pautas de diseño que establece la UCI para productos comerciales.
- **RnF2.** Debe ser intuitivo, agrupando los requisitos funcionales por responsabilidad, lo que permite a los administradores de poca experiencia hacer uso del mismo.

Requisitos de Hardware:

- **RnF3.** El ordenador donde se ejecute el sistema debe tener una memoria *RAM* de 4GB o superior, un procesador a una velocidad 3 GHz o superior, 4 o más núcleos lógicos, al menos 4Mb de memoria cache y 200Mb libres de almacenamiento.

Requisitos de Software:

- **RnF4.** El sistema debe estar conectado con el endpoint SPARQL de DBpedia para las consultas semánticas.
- **RnF5.** El sistema debe recibir y responder a las peticiones mediante una API RESTful.

Requisitos de Seguridad:

- **RnF6.** La información manejada en el sistema debe estar protegida de accesos no autorizados mediante el cortafuego.
- **RnF7.** Los datos sensibles como las respuestas a las preguntas en la base de dato deben estar encriptadas en Base64[18].

2.4 Modelo de Casos de Uso del Sistema

[18] Método de codificación muy utilizado para cadenas de caracteres (Siahaan, 2017)

El modelo de casos de uso describe la funcionalidad propuesta del nuevo sistema. Un Caso de Uso representa una unidad discreta de interacción entre un usuario (humano o máquina) y el sistema (Sparks, 2013).

2.4.1 Diagrama del Caso de Uso del Sistema

Para favorecer la organización y el entendimiento del caso de uso, se muestra en la figura 4, inicializado por el autor correspondiente.

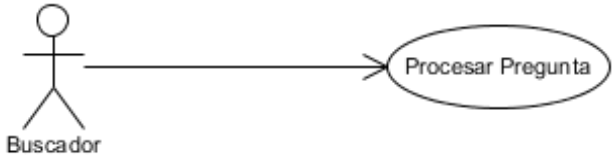


Figura 4: Diagrama del caso de uso Procesar pregunta

En el diagrama se encuentra representado un total de 1 caso de uso y el actor que interactúa con él. A continuación, se presenta una breve descripción de este caso de uso.

- **CU1** Procesar pregunta: Engloba todo el proceso desde la captura de la pregunta por el sistema, el análisis del lenguaje natural, la conversión a lenguaje de consulta semántico, la búsqueda en una base de conocimiento y devolver los resultados.

2.4.2 Especificación del caso de uso

Tabla 5: Caso de uso Procesar Pregunta

Objetivo	Con este caso de uso el sistema debe responder a una pregunta en lenguaje natural	
Actores	Buscador	
Resumen	El buscador envía una pregunta en lenguaje natural, el sistema la procesa y devuelve la respuesta en lenguaje natural también.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones	El buscador debe solicitar una pregunta en lenguaje natural	
Postcondiciones	Debe enviarse una respuesta en lenguaje natural	
Flujo de eventos		
Flujo básico Manipular mapa		
	Actor	**Sistema**
1.	El buscador envía una pregunta en lenguaje natural	
2.		Extrae la pregunta en lenguaje natural

3.		Extrae la forma de la pregunta
4.		Elimina palabras irrelevantes
5.		Efectúa el procesamiento del lenguaje natural
6.		Construye la sentencia SPARQL
7.		Realizar consulta en la base de datos del sistema
8.		Conectar el sistema con la base de conocimiento
9.		Realizar consulta en la base de conocimiento
10.		Devolver información de la base de conocimiento
11.		Normalizar los datos
12.		Guardar los datos en la base de datos del sistema
13.		Enviar respuesta
14.	El buscador recibe la respuesta	
Relaciones	**CU incluidos**	
	CU extendidos	

2.5 Modelo de Diseño

El objetivo fundamental del modelo de diseño es transmitir a través de diagramas una comprensión en profundidad de los aspectos relacionados con los requerimientos no funcionales y restricciones que conciernen a los lenguajes de programación. Es aquel que se encarga de la realización de los casos de uso del sistema, y se utiliza como medio de abstracción entre el modelo de la implementación y el código fuente. Los modelos de diseño pueden ser estructurales o dinámicos. Los primeros describen la estructura estática del sistema utilizando las relaciones de los objetos y las clases. Las dinámicas muestran las interacciones entre los objetos del sistema (Somerville, 2011).

2.5.1 Diagrama de clases del diseño

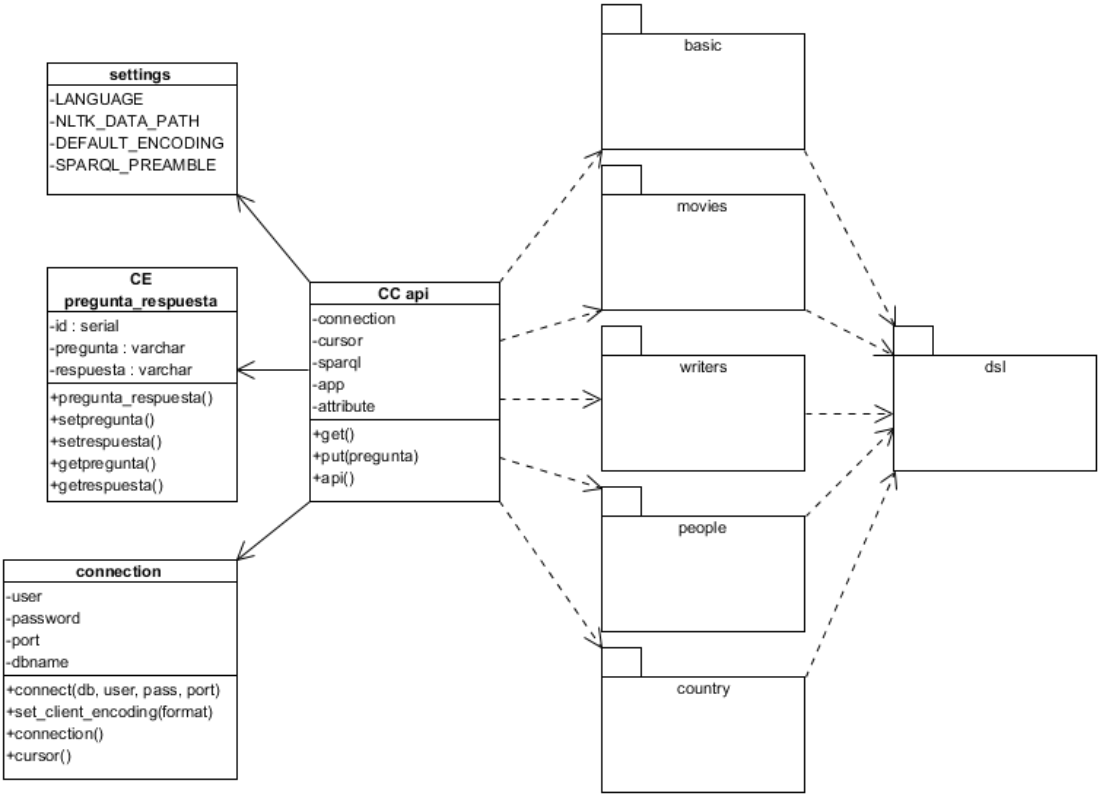


Figura 5: Diagrama de clases del caso de uso “Procesar pregunta”

En la figura 5 se muestra como la clase controladora “api” es la que presenta más interacciones con otras clases, paquetes y el modelo de la base de datos. Esta clase es la que se encarga de cargar el servicio RESTful, realizar la conexión con la base de datos del sistema, maneja el *matching* de las preguntas con los paquetes y de configurar la conexión con la base de conocimiento. La clase “*settings*” se encargará de cargar la configuración de la biblioteca para el análisis semántico “NLTK”. La clase “*connection*” es la encargada de realizar la conexión con la base de datos del sistema y la clase “pregunta_respuesta” es la clase de la base de datos con los atributos y métodos para realizar las consultas necesarias. Los paquetes “*basic*”, “*movies*”, “*writers*”, “*people*” y “*country*” representan una porción de las plantillas Python que se encargaran de representar cada tipo de pregunta para que la clase controladora pueda realizar el *matching* utilizando expresiones regulares. El paquete “dsl” se encargará de definir todas las características que presenta un recurso de la base de conocimiento DBpedia y este es utilizado por las plantillas Python.

2.6 Modelos de interacción

Un modelo de interacción expone visualmente las interacciones existentes entre las instancias y las clases del modelo de estas. Estos pueden tener dos enfoques: modelado de colaboración de caso de uso y diagrama de secuencia. Ambos son importantes debido a que ayudan a identificar los requerimientos del usuario. El modelado de la interacción sistema a sistema destaca los problemas de comunicación que pudieran llegar a presentarse (Somerville, 2011). Los diagramas de secuencia representan el tiempo como una dimensión en el intercambio de mensajes. Debido a que el proceso del subsistema se ejecuta con un orden, para una mejor representación de las interacciones del sistema se decide utilizar los mismos.

2.6.1 Diagrama de secuencia

A continuación, se muestra el diagrama de secuencia del caso de uso “Procesar pregunta”.

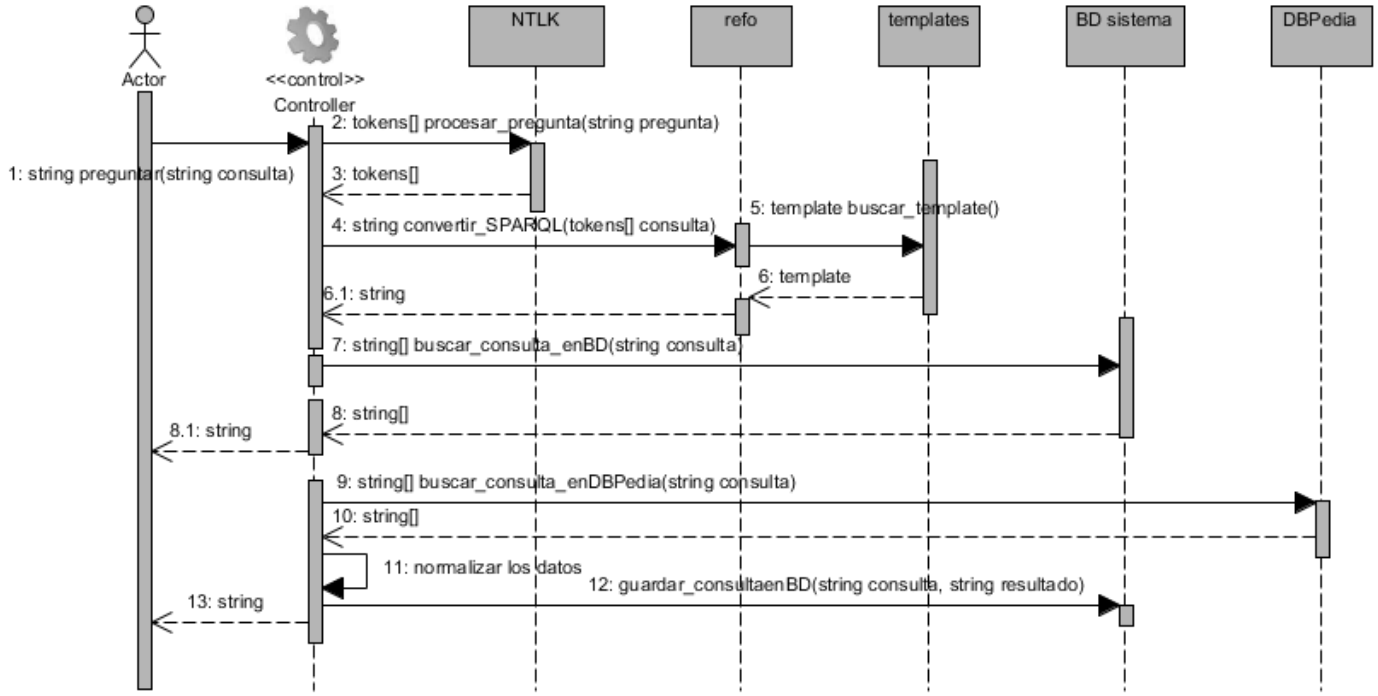


Figura 6: Diagrama de secuencia perteneciente al caso de uso “Procesar Pregunta”

El subsistema inicia su secuencia cuando interactúa con una petición del buscador que le solicita una respuesta en lenguaje natural a una pregunta en lenguaje natural, ambas guardadas en un tipo de datos de cadenas. El controlador llama a la clase que se encarga de realizar el procesado del lenguaje natural y

devuelve una lista con objetos etiquetados (*tokens*). Después el subsistema envía esa lista a la clase “refo” que se encarga de construir una consulta SPARQL siguiendo las plantillas integradas por defecto y devolviendo una cadena de caracteres al controlador con la consulta formulada. Se ejecuta la consulta en la Base de Datos del sistema, de no arrojar resultados se efectúa la consulta en la Base de conocimiento (DBpedia), se extraen los resultados, se normalizan aplicándole un formato de texto, se guardan en la base de datos del sistema y se devuelven los resultados al buscador.

2.7 Patrones de diseño utilizados en el desarrollo de software

Los patrones de diseño constituyen la descripción de un problema específico y recurrente, que aparece en contenidos determinados, y presenta un esquema genéricamente demostrado con éxito para su solución; este último se especifica mediante la descripción de los mecanismos que la constituyen, sus responsabilidades y desarrollos, así como también la forma como estos colaboran entre sí (Somerville, 2005).

En el diseño del sistema para determinar respuesta exacta se tuvieron en cuenta los siguientes patrones GRASP (Patrones Generales de Software para Asignación de Responsabilidades), que describen los principios fundamentales de la asignación de responsabilidades a objetos:

Experto: este patrón plantea que se debe asignar una responsabilidad al experto en información, en otras palabras, a la clase que cuenta con los datos necesarios para cumplir la responsabilidad. De esta forma, se conserva el encapsulamiento de la información, puesto que los objetos ejecutan las tareas que le corresponden de acuerdo a la información que poseen, lo que da lugar a sistemas más robustos y fáciles de mantener. En el marco de la presente investigación siguiendo el patrón Experto en Información se le asignaron responsabilidades determinadas solamente a las clases que cuentan con la información necesaria para dar cumplimiento a las mismas, esto se evidencia en la Tabla 6. De esta forma, se mantiene el encapsulamiento de la información, puesto que los objetos utilizan su propia información para llevar a cabo las tareas. Normalmente, esto conlleva un bajo acoplamiento, lo que da lugar a sistemas más robustos y más fáciles de mantener.

Tabla 6: Relación de asignación de responsabilidades

Clase Objeto	Responsabilidad
SPARQLWrapper.py	Se encarga de conectar el sistema con la base de conocimiento, ejecutar las consultas y devolver los resultados
Flask.py	Se encarga de ejecutar el servicio de la interfaz web y recibir todas las peticiones al subsistema

Creador: la instanciación de una clase es una de las actividades fundamentales en un sistema orientado a objetos. Este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos, con lo que se logra menos dependencia y mayores oportunidades de reutilización de código (Larman, 2004). La clase “API” es la encargada de ejecutar consultas y devolver los resultados para esto crea una instancia de la clase “SPARQLWrapper” en la librería “SPARQLWrapper-1.8.1-py2.7.egg” que realiza y configura la conexión con la base de conocimiento. En la figura 7 se muestra el flujo de creación de los objetos.

Figura 7: Creación de objeto en la clase API

Alta cohesión: en el diseño orientado a objetos, la cohesión es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un elemento (clase o subsistema). Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas, que colaboran entre sí y con otros objetos para simplificar su trabajo. Una clase con alta cohesión es relativamente fácil de mantener, entender y reutilizar. La utilización de este patrón de diseño se manifiesta en el diseño de la clase “API”. Esta clase en orden de cumplir la responsabilidad que le es asignada mediante el patrón Experto, colabora y a su vez delega responsabilidades en la clase “Flask” para ejecutar el servicio Web e interactúa a su vez con otras clases que utilizan el patrón Experto como “SPARQLWrapper”.

Controlador: este patrón tiene como objetivo asignar la responsabilidad a una clase de recibir o manejar un mensaje de evento del sistema generado por un actor externo, por lo general a través de una interfaz

gráfica de usuario a la que accede un usuario para realizar ciertas operaciones en el sistema. La utilización de este patrón se evidencia en la clase “API”, la misma se encarga de atender las preguntas formuladas por el usuario. La figura 6 que muestra el diagrama de secuencia perteneciente al CU “Procesar Pregunta”, muestra el flujo de peticiones recibidas por esta clase, así como las respuestas.

2.8 Diseño de la Base de Datos

Uno de los pasos cruciales en la construcción de una aplicación que requiera manejar la persistencia de la información, es sin duda, el diseño de la base de datos, la cual, correctamente diseñada, permite obtener acceso a información exacta y actualizada. El sistema que se propone hace uso de una base de datos ya que es necesario persistir una cantidad considerable de información relacionada con las preguntas y respuestas realizadas.

La base de datos del sistema para determinar respuesta exacta estará compuesta por cuatro variables: El id que almacena la llave primaria de la base de datos, la consulta SPARQL procesada por el sistema que describe una cadena de caracteres, la respuesta expresada en lenguaje natural lista para ser devuelta a la plataforma y la fecha en la que se guardó la consulta. En el caso de que la fecha esté caducada y la información que guarde la base de datos este desactualizada, se borrará automáticamente. En la Figura 8 se observa el diagrama de la base de datos “pregunta_respuesta”.

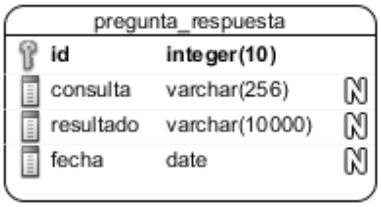


Figura 8: Diagrama de la base de datos

2.9 Modelo de despliegue

Para una mejor comprensión entre la correspondencia de la arquitectura de software y la arquitectura de hardware se realiza el modelo de despliegue. El modelo de despliegue que se presenta a continuación muestra un entorno de ejecución que alberga el “Servidor Buscador”, representado por un nodo como un ambiente de ejecución el cual envía preguntas al sistema para determinar respuesta exacta.

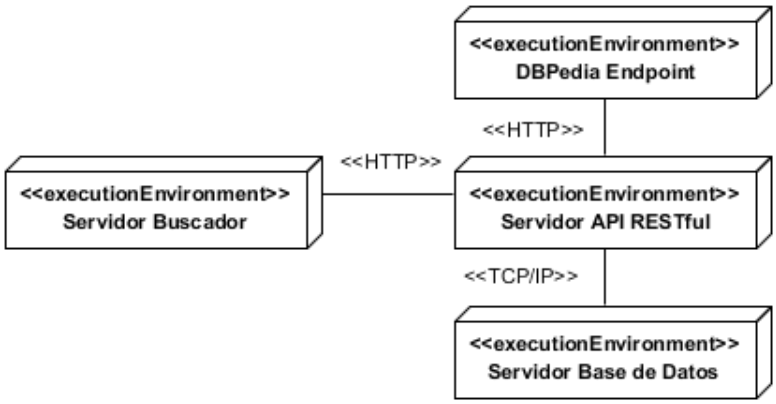


Figura 9: Diagrama de Despliegue

DBpedia endpoint se encuentra representado por un nodo el cual constituye un servidor para consultas semánticas que utiliza el protocolo HTTP. Por otro lado, se encuentra la herramienta Quepy la cual se encarga del procesamiento del lenguaje natural y la conversión de la pregunta a SPARQL la cual se encuentra en un servidor utilizado una interfaz API. El servidor de la Base de datos utiliza el conjunto de protocolos TCP/IP para la conexión con el sistema. El servicio corre mediante una interfaz de aplicaciones (API), utilizando el protocolo HTTP para comunicarse con los diferentes subsistemas.

2.10 Conclusiones parciales

En este capítulo se analiza y se diseña la propuesta de solución.

- La representación y descripción de los artefactos generados garantizaron un mejor entendimiento de los flujos de trabajos presentes en el proceso de búsqueda de una imagen.
- La especificación de los requisitos funcionales y no funcionales del sistema, dieron paso a una mejor comprensión, por parte de los autores, de los resultados que se pretenden obtener de una manera precisa y sirvieron de guía para la implementación del sistema.
- La definición de la arquitectura y los patrones de diseño a utilizar, permitieron establecer las bases para fomentar la reutilización y las buenas prácticas de programación entre los desarrolladores durante la fase de implementación, así como disminuir el impacto de los cambios futuros en el código fuente.
- La elaboración del diagrama de despliegue permitió identificar la disposición física de los artefactos de la herramienta informática a desarrollarse..

Capítulo 3: Implementación y validación del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.

La implementación del sistema es la fase importante dentro del ciclo de desarrollo de software. Esta fase comprende la objetivación, en forma de código, de todos los componentes, representaciones y arquitectura propuestos en la etapa de análisis y diseño; con el objetivo de conformar el producto final requerido por el cliente.

Junto al proceso de implementación, el software que se va construyendo debe ser sometido a determinadas pruebas que confirmen la correspondencia entre el producto y los requisitos definidos en las etapas anteriores. A esta fase se le conoce como validación y en ella, pueden realizarse diferentes tipos de pruebas en función de los objetivos de las mismas.

3.1 Modelo de componentes que integran el subsistema informático

El modelo de componentes representa la forma en que es estructurado un sistema informático atendiendo a las diferentes partes que lo componen. Puntualiza que cada componente debe ser tratado como una unidad de composición independiente e indispensable dentro de un sistema, y que puede contraer relaciones de dependencia con otros componentes. Algunos ejemplos de componentes físicos lo constituyen los archivos, módulos, librerías, ejecutables, binarios, entre otros (Pressman, 2005).

3.1.1 Diagrama de componentes

Un diagrama de componentes permite visualizar con facilidad la estructura general del sistema y el comportamiento de las funcionalidades que estos componentes proporcionan y utilizan a través de las interfaces. Además, muestra la organización y las dependencias entre un conjunto de componentes (Somerville, 2011).

A continuación, se describen los principales paquetes que componen el diagrama de componentes correspondiente a la interfaz web del subsistema implementado:

- **QAS:** Agrupa en su interior todos los componentes del servicio implementado, estableciendo de esta manera una estructura organizativa.
- **ServerAPI:** Contiene en su interior las clases y métodos básicos para lanzar un servicio API RESTful para la recepción del texto de la pregunta del buscador.

- **dbpedia:** Contiene clases y métodos básicos que ayudan al subsistema a la conexión con la base de datos, el ajuste de las opciones, hacer coincidir la forma de la pregunta con la plantilla Python.
- **templates:** Contiene disimiles plantillas para el análisis de la pregunta en forma SPARQL.

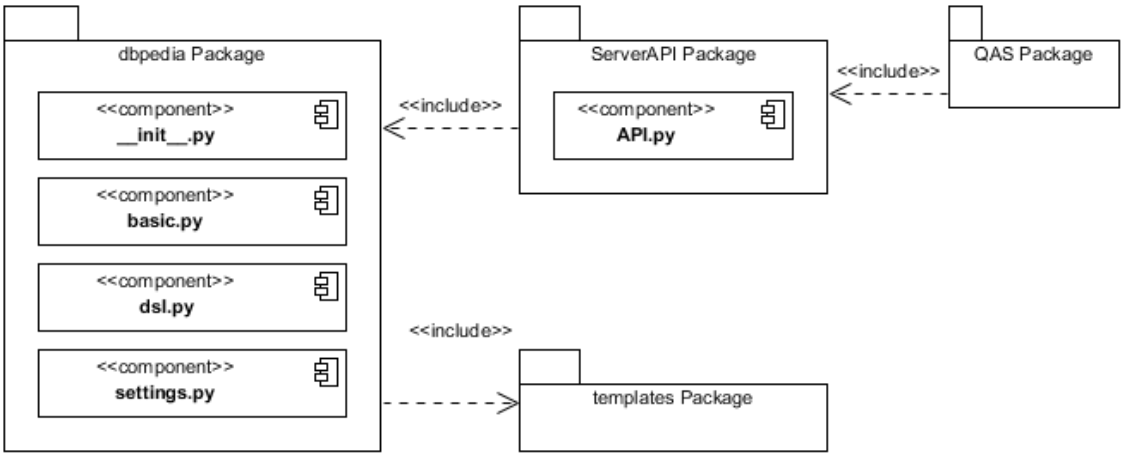


Figura 10: Diagrama de componentes del subsistema para determinar respuesta exacta

3.2 Estándares de codificación utilizados

Los estándares de codificación son especificaciones o estilos que establecen la forma de generar el código funcional de las aplicaciones informáticas. Debido a que, en muchas ocasiones, los sistemas de cómputo son implementados por varios programadores, la adopción inicial de un único estilo de codificación constituye uno de los factores de mayor peso en la calidad, rendimiento, legibilidad y capacidad de mantenimiento del producto final (Somerville, 2011).

Para lograr este objetivo se utilizó la Guía de estilo para el código Python (PEP 8) (Gómez, 2016). Esta guía posee una gran cantidad de convenciones para escribir código legible, dentro de las cuales se utilizan en el desarrollo del subsistema:

• Usar cuatro espacios por identación.

• Nunca mezclar tabulaciones y espacios.

• Limitar todas las líneas a un máximo de caracteres (120 en este proyecto).

• Separar funciones de alto nivel y definiciones de clase con dos líneas en blanco, mientras que las definiciones de métodos dentro de una clase son separadas por una línea en blanco.

• Codificación UTF-8 en todos los módulos.

• Las importaciones deben estar en líneas separadas.

• Evitar usar espacios en blanco innecesarios.

• Utilizar el estilo CamelCase para nombrar clases, y el lower_case_with_underscores para funciones y métodos.

```
# coding: utf-8

from flask import Flask
from flask_restful import Resource, Api, reqparse
import quepy
from SPARQLWrapper import SPARQLWrapper, JSON
import gzip
import io

app = Flask(__name__)
api = Api(app)

connection = psycopg2.connect("dbname=pregunta_respuesta user=postgres password=postgres")
connection.set_client_encoding('UTF8')

cursor = connection.cursor()

sparql = SPARQLWrapper("http://dbpedia.org/sparql")


def print_define(results, target, metadata=None):
    for result in results["results"]["bindings"]:
        if result[target]["xml:lang"] == "en":
            print result[target]["value"]


class api(Resource):
    def get(self):
        dbpedia = quepy.install("dbpedia")
        pregunta = args.get('pregunta')

        cursor.execute("SELECT respuesta FROM pregunta_respuesta WHERE consulta='"+pregunta+"';")
        if cursor.rowcount != 0:
            row = cursor.fetchone()
            print("The number of parts: ", cursor.rowcount)
            return row[0]
        return 0


if __name__ == '__main__':
```

Figura 11: Fragmento de código de la clase api.py

En la Figura 11 se ven reflejados todos los puntos del estándar de codificación destacados al principio del epígrafe. Se observa cómo se utiliza la codificación utf-8 al principio del fragmento de código, cuatro espacios para identar y las líneas no exceden los 120 caracteres. Los métodos son separados por dos espacios de distancia, no se utilizan espacios innecesarios ni tabulaciones, los métodos presentan el estilo adecuado como es el caso de “print_define” y las importaciones se encuentran en líneas separadas.

3.3 Validación del subsistema para determinar respuesta exacta en la plataforma C.U.B.A.

A continuación, se detallan los tipos de pruebas de software aplicadas al subsistema implementado. Las mismas persiguen como objetivo fundamental, la detección de las no conformidades respecto a las funcionalidades de la aplicación, la medición del grado de usabilidad de las funcionalidades implementadas, así como también la correcta integración entre los diferentes componentes de la arquitectura del sistema.

3.3.1 Pruebas funcionales

Las pruebas funcionales son aquellas que se aplican a un software determinado, con el objetivo de validar que las funcionalidades implementadas funcionen de acuerdo a las especificaciones de los requisitos definidos con anterioridad. Para la ejecución de este tipo de pruebas, suelen emplearse dos métodos fundamentales: el método de Caja Blanca y el método de Caja Negra (Somerville, 2011). El primero se centra en las pruebas al código de las aplicaciones; mientras que el segundo permite a los probadores enfocar su atención en el funcionamiento de la interfaz, a través del análisis de los datos de entrada y los de salida. En este epígrafe se exponen los aspectos concernientes a las pruebas funcionales realizadas utilizando el método de Caja Negra, a partir de los casos de prueba diseñados.

Diseño de los casos de prueba basados en casos de uso

A continuación, se muestra un fragmento del diseño del caso de prueba basado en caso de uso “Procesar Pregunta”.

Tabla 7: Caso de prueba correspondiente al CU “Procesar Pregunta”

EC	Descripción (Todas las peticiones se realizan utilizando una petición HTTP mediante el método GET)	V1	Respuesta del Sistema	Flujo central
1.1	La pregunta no se encuentra guardada en la Base de Datos del sistema	qué%20es%20un %20animal	Respuesta a la pregunta	1-La pregunta no se encuentra en la base de datos 2-Se genera la consulta SPARQL 3-Se realiza la consulta SPARQL y se devuelve el resultado 4-Se normalizan los datos y se guardan en la base de datos del sistema 5-Se devuelve el resultado
1.2	La pregunta se encuentra guardada en la Base de Datos del sistema	qué%20es%20un %20animal	Respuesta a la pregunta guardada en la base de datos del sistema	1-La pregunta se encuentra en la base de datos del sistema 2-Se devuelve el resultado
2.0	La pregunta no se encuentra en la base de datos del sistema y no se encuentra en la base de conocimiento	cúal%20es%20la%20 capital%20de%20Asd	“No se pudo responder a la consulta SPARQL de la pregunta: cúal%20es%20la%20 capital%20de%20Asd”	1-La pregunta no se encuentra en la base de datos 2-Se genera la consulta SPARQL

				3-Se realiza la consulta SPARQL y no se devuelve un resultado 4-El sistema responde un error
3.0	La pregunta no se encuentra en las plantillas Python	asd%20asd%20ads%20as%20de	"No se generó consulta SPARQL para la pregunta: asd%20asd%20ads%20as%20de"	1-La pregunta no se encuentra en la base de datos 2-No se genera la consulta SPARQL 3-El sistema responde un error

Resultados de las pruebas funcionales

Con el objetivo de probar la correcta ejecución de las funcionalidades del sistema se realizaron 4 iteraciones de pruebas al subsistema. En la Tabla 8 se muestran los resultados obtenidos en cada iteración de prueba al subsistema para determinar respuesta exacta, así como la corrección de cada uno de los errores.

Tabla 8: Resultados de la prueba funcional.

No conformidades	1era Iteración	2da Iteración	3era Iteración	4ta Iteración
Detectadas	12	2	1	0
Resueltas	10	1	1	0
Pendientes	0	0	0	0

En la Figura 12 se puede apreciar el comportamiento de las no conformidades de las pruebas funcionales ejecutadas.

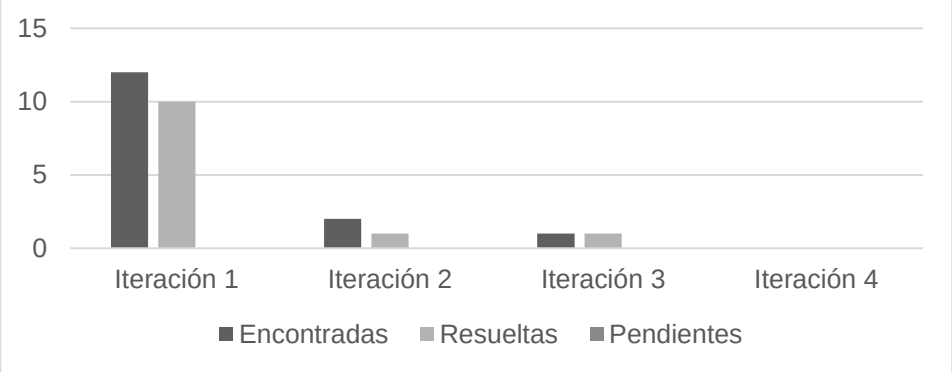


Figura 12: Comportamiento de las no conformidades por cada iteración de las pruebas.

3.3.2 Pruebas de integración

Las pruebas de integración son definidas para verificar el correcto ensamblaje entre los distintos módulos que conforman un sistema informático. Las mismas validan que estos componentes realmente funcionan juntos, son llamados correctamente y, además, transfieren los datos correctos en el tiempo preciso y por las vías de comunicación establecidas (Somerville, 2011).

Una vez realizadas las pruebas funcionales a cada componente interno de manera independiente, y verificado que las funcionalidades implementadas se corresponden de acuerdo a los requisitos funcionales y no funcionales establecidos; se pudo comprobar el correcto funcionamiento de los componentes mediante el estudio del flujo de datos entre ellos. Posterior a estas pruebas, se hace necesaria la realización de pruebas de integración, con la finalidad de validar la compatibilidad y el funcionamiento de las interfaces que comunican las diferentes partes que componen la solución informática.

Para la realización de las pruebas de integración se llevaron a cabo diferentes acciones, a continuación, se mencionan las fundamentales:

- Verificación de la unión entre el leguaje natural procesado y la generación de consulta SPARQL mediante plantillas Python.
- Verificación de la conexión entre el lenguaje natural procesado a lenguaje de consulta SPARQL y la ejecución de la consulta en la base de conocimiento.
- Verificar la integración de la consulta con la base de datos del sistema.
- Comprobar que el servicio API RESTful está correctamente integrado con los demás componentes.

La ejecución de las pruebas de integración permitió verificar el trabajo conjunto de los componentes del subsistema en cuestión. Se hizo énfasis en la integración de la conversión de lenguaje natural procesado a SPARQL, la conexión con la base de datos del sistema, la base de conocimiento y la integración del servicio RESTful con todos los demás componentes, para detectar incoherencias en el funcionamiento de la aplicación. No se encontró ninguna no conformidad, llegándose a la conclusión que existe una correcta integración entre los componentes internos del subsistema.

3.3.3 Pruebas de carga y estrés

Las pruebas de carga y estrés consisten en probar el funcionamiento del software bajo condiciones extremas. Estudia la especificación del software, las funciones que debe realizar, las entradas y las salidas, analizando los valores límites. Las pruebas de estrés están diseñadas para enfrentar al programa a condiciones inverosímiles. Las pruebas ejecutan un sistema, de manera que demande recursos en cantidad, frecuencia o grandes volúmenes (López, 2010).

Para la validación del subsistema para determinar respuesta exacta se utilizó la herramienta JMeter en su versión 3.0 para establecer un mínimo de recursos que deberá tener el sistema para ejecutar la aplicación y un límite de conexiones que soporta el servicio web realizando un gran número de peticiones.

La prueba que se realizó consistió en solicitar 200 y 300 peticiones en forma de pregunta en dos momentos diferentes al subsistema y en dos ordenadores con diferentes características, las cuales simulan peticiones de los usuarios de manera concurrente. La primera prueba se realizó en un sistema con un procesador Intel(R) Pentium(R) Dual-Core @1.9Ghz, una memoria RAM de 4GB a 1600Mhz de frecuencia y un ancho de banda Ethernet de 100Mb/s, y a las 14:00 horas para 200 hilos cada 1 segundo, y en un segundo momento a las 16:34 horas se realizó la misma prueba después de que el sistema guardara los datos en la base de datos interna.

La Segunda prueba fue realizada en un procesador AMD(R) Quad-Core Ryzen 3 1300X @3.9Ghz, una memoria RAM de 8GB a 3200Mhz de frecuencia y un ancho de banda Ethernet de 1000Mb/s, y a las 18:00 horas para 300 hilos cada 1 segundo, y en un segundo momento se realizó la misma prueba después de que el sistema guardara los datos en la base de datos interna.

Para mejorar el entendimiento de los componentes “Reporte Resumen” que se verán a continuación, se explica cada parámetro que la compone.

- **#Muestras:** cantidad de hilos utilizados por la URL
- **Media:** tiempo promedio en milisegundos para un conjunto de resultados.
- **Min:** tiempo mínimo que demora un hilo en acceder a una página.
- **Max:** tiempo máximo que demora un hilo en acceder a una página.
- **Rendimiento:** rendimiento medido en los requerimientos por segundo / minuto / hora.
- **Kb/seg:** rendimiento medido en Kbytes por segundo.
- **%Error**: Porcentaje de peticiones que no fueron respondidas

Los valores totales obtenidos por la componente “Reporte resumen” para 200 hilos se muestran en la Tabla 9.

Tabla 9: Resultados de prueba de carga y estrés con el acceso de 200 usuarios en el primer ordenador.

	#Muestras	Media	Min	Max	%Error	Rendimiento	Kb/seg
1ra Prueba	200	3753	974	9221	0.13	3.79	3.58
2da Prueba	200	3300	272	9221	0.02	1.58	1.45

Como se muestra en la Tabla 9, el subsistema desarrollado, para 200 usuarios conectados de forma concurrente en un primer momento respondió 200 peticiones al servidor en un promedio de 3.753 segundos, lo que equivale a 3.79 peticiones por segundo. Una segunda prueba que se le realizó al sistema después de guardar los resultados obtenidos en la base de conocimiento en la base de datos del sistema, este arrojó una media de 3.3 segundos, con un promedio de 1.58 peticiones por segundo.

Tabla 10: Resultados de prueba de carga y estrés con el acceso de 300 usuarios en el segundo ordenador.

	#Muestras	Media	Min	Max	%Error	Rendimiento	Kb/seg
1ra Prueba	300	7	1	31	0.00	50.76	32.24
2da Prueba	300	5	0	31	0.00	62.98	45.40

Como se muestra en la Tabla 10, el subsistema desarrollado, para 300 usuarios conectados de forma concurrente en un primer momento respondió 300 peticiones al servidor en un promedio de 0.007 segundos,

lo que equivale a 50.76 peticiones por segundo. Una segunda prueba que se le realizó al sistema después de guardar los resultados obtenidos en la base de conocimiento en la base de datos del sistema, este arrojó una media de 0.005 segundos, con un promedio de 62.98 peticiones por segundo.

3.3.4 Pruebas de seguridad

La seguridad informática comprende la puesta en práctica de un conjunto de medidas preventivas y reactivas en los sistemas informáticos y tecnológicos, que posibilitan la protección de la información, persiguiendo como objetivo principal la integridad, confidencialidad y disponibilidad de la misma (Somerville, 2011).

Las realizaciones de pruebas de seguridad contribuyen a la detección temprana de vulnerabilidades y la toma de medidas para la disminución de amenazas de ataque, y con ello proveer sistemas de cómputo más seguros y confiables. Al subsistema desarrollado se le realizaron una serie de pruebas de seguridad mediante el software Acunetix, las cuales se presentan a continuación:

- Ataques de inyección.
- Detección de ficheros y directorios.

En la siguiente tabla se verá un resumen de las principales vulnerabilidades encontradas en el subsistema.

Tabla 11: Vulnerabilidades del sistema

Tipo	Cantidad	Descripción	Recomendaciones
Software(Flask)	1	La versión del entorno de trabajo Flask contiene una vulnerabilidad en el método safe_join cuando el sistema este ejecutándose en Windows server.	Actualizar Flask a una versión más reciente.
Software(Python)	2	La versión de Python contiene una vulnerabilidad de CRLF [19] en la	Actualizar Python 2.7.9 a la versión 3.5.

[19] CRLF se refiere a un tipo de inyección vía HTTP (Kshirsagar, 2016)

		biblioteca urllib2 permitiendo ataques de cabecera HTTP, también contiene una vulnerabilidad en la función “scanstring” del módulo _json que permite leer a los atacantes procesos arbitrarios en memoria.	

Observaciones: Todas las vulnerabilidades fueron resueltas en el servidor y se recomienda que se tenga en cuenta la nueva configuración en el servidor para un futuro despliegue del sistema.

Todos los resultados que se corrigieron fueron satisfactorios llegando a la conclusión de que el subsistema para determinar respuesta exacta es seguro y se encuentra en disposición de ser utilizado por el cliente.

3.4 Validación de la hipótesis de la investigación

Tabla 12. Operacionalización de las variables.

Variable	Dimensión	Definición	Indicadores	Unidades de medida
Mejorar la precisión de los resultados	Mejorar precisión	Mejorar la calidad de los resultados en las búsquedas realizadas por los usuarios en el buscador cubano	Precisión	Cualitativa

Con el objetivo de evaluar el indicador asociado a precisión en el subsistema para determinar respuesta exacta, correspondiente a la variable dependiente determinada como parte de la hipótesis de investigación. Se realizó un cuasi-experimento donde se compararon los resultados obtenidos en la plataforma C.U.B.A. actual y el subsistema implementado.

Según Dubey (2016) existen cinco posibles tipos de respuestas en un sistema de pregunta-respuesta. La metodología para validar la veracidad del sistema deberá responder correctamente a cinco tipos de preguntas formuladas con diferentes resultados. Estos pueden ser booleano, ranking, número, conjunto o valor de propiedad. Para ello se realizó una prueba con cinco preguntas en total, cada una con una

respuesta diferente. Para el indicador de precisión se midió en cuanto a valores cualitativos en Alto, Medio o Bajo el índice obtenido por la siguiente fórmula donde el valor P corresponde al índice de precisión:

$$P = \frac{\#\ docRelevantesRecuperados}{\#totalDocRecuperados}$$

Figura 13: Índice de precisión en sistemas de recuperación de información (Romá, 2014)

Para la asignación del valor cualitativo del indicador “Precisión”, se determinó un intervalo del índice de precisión de 0,67 a 1,0 para el valor Alto, de 0,34 a 0,66 para el valor Medio y de 0,0 a 0,33 para el valor Bajo. Los resultados del cuasi-experimentos se muestran en la tabla siguiente.

Tabla 13: Resultados de la medición del indicador “Precisión”

Pregunta/Sistema	**Booleano**	**Ranking**	**Número**	**Conjunto**	**Valor de Propiedad**
Plataforma C.U.B.A.	Bajo	Bajo	Bajo	Bajo	Bajo
Subsistema de respuesta exacta	Alto	Alto	Alto	Alto	Alto

Al interpretar la tabla 13, se observa que los valores alcanzados por el indicador precisión en el subsistema para determinar respuesta exacta son altos para todos los tipos de pregunta. A partir del análisis de la comparación que se evidencia en la tabla, utilizando el subsistema desarrollado se obtienen resultados con mayor calidad para el usuario con respecto a la plataforma C.U.B.A, concluyendo esto como un efecto satisfactorio que apoya la hipótesis de la presente investigación.

3.5 Conclusiones parciales

En este capítulo se abordaron una serie de aspectos correspondientes a la implementación y validación del subsistema para determinar respuesta exacta.

- La representación y descripción del diagrama de componentes permitió visualizar con más facilidad la estructura general del subsistema.
- La ejecución de pruebas al subsistema permitió detectar las deficiencias presentes, subsanarlas en el menor tiempo posible y ofrecer una aplicación con mayor calidad, seguridad y usabilidad.
- La aplicación del método experimental y la realización de cálculos estadísticos, aportaron elementos sustanciales que permitieron validar la hipótesis de investigación planteada con anterioridad, y con ello la contribución del subsistema implementado.

CONCLUSIONES

Una vez concluida la presente investigación, se puede definir que:

- A partir del estudio realizado de los fundamentos teóricos relacionados con la recuperación de información y los sistemas pregunta-respuesta se determinó que existen una serie de Sistemas de Recuperación de Información los cuales brindan funcionalidades de un sistema de pregunta-respuesta. Ninguno de ellos tiene licencia libre para su uso, ni responden a las necesidades de los sistemas de recuperación de información cubanos, definiéndose una propuesta de solución de acuerdo a las necesidades existentes.
- El enfoque ágil propuesto por la metodología AUP-UCI y el uso de las tecnologías y herramientas seleccionadas, permitieron analizar y describir los subprocesos que se debían ejecutar, concretando así, en concordancia con las especificaciones del cliente, las características que debía tener el subsistema a desarrollarse.
- Una vez estudiados los elementos que intervienen en el proceso para determinar respuesta exacta, fue posible la modelación de los artefactos que contribuyeron al diseño de la propuesta de solución posibilitando un mayor soporte a la implementación de los requisitos previamente expresados por el cliente; garantizando la estructura base para la organización lógica del código fuente y la disminución del impacto ante futuras modificaciones en la aplicación.
- La evaluación de las pruebas de software realizadas permitió erradicar las insuficiencias detectadas en el subsistema desarrollado logrando así un producto más seguro y funcional conforme a las necesidades de los usuarios finales.
- La implementación del subsistema informático para determinar respuesta exacta utilizando las herramientas y tecnologías estudiadas y orientada por las tres fases de la metodología AUP-UCI permitió solucionar los problemas existentes planteados en la problemática de la presente investigación.
- La validación de la hipótesis a través de un cuasi-experimento demostró la calidad del subsistema desarrollado, resultado que fue satisfactorio para los autores quedando demostrada la hipótesis de la investigación.

RECOMENDACIONES

Una vez concluida la investigación y el desarrollo de la propuesta de solución, el autor de la presente investigación recomienda:

- Implementar una base de conocimiento ajustada a los contenidos cubanos para facilitar la búsqueda del subsistema para determinar respuesta exacta.
- Añadir funcionalidades al subsistema que permitan mostrar los resultados de las consultas a través de imágenes, sonido y videos.
- Integrar el subsistema para determinar respuesta exacta en el entorno real de la plataforma C.U.B.A.

REFERENCIAS BIBLIOGRÁFICAS

ABRAMOVICH, SERGEI. Educating teachers to pose mathematical problems in the digital age: Toward alternative ways of curriculum design. *IMVI Open Mathematical Education Notes*, 2015, vol. 5, no 2, p. 115-136.

ADDATI, ANGEL; ROGER, SANDRA. Agentes inteligentes y web semántica: preprocesamiento de texto de redes sociales. En XIX Workshop de Investigadores en Ciencias de la Computación (WICC 2017, ITBA, Buenos Aires), 2017.

AGGARWAL, SHALABH. Flask Framework Cookbook. Packt Publishing Ltd, 2014.

ALMOHAIMEED, ABDULAZIZ. Using Tweets Sentiment Analysis to Predict Stock Market Movement, 2017.

ALUPULUI, MARIANA, ET AL. Question-answering system. U.S. Patent Application No 15/851,769, 10 Mayo 2018.

ALVARADO-URIBE, JOANNA, ET AL. Una herramienta visual para la búsqueda semántica RDF. *Research in Computing Science*, 2015, vol. 95, p. 9-22.

BAEZA-YATES, R.; RIBEIRO-NETO, B. Modern information retrieval. New York: ACM Press; Harlow [entre otros.]: Addison-Wesley, 1999. ISBN 0-201-39829-X.

BANSAL, RITIKA; CHAWLA, SONAL. An approach for semantic information retrieval from ontology in computer science domain. International Journal of Engineering and Advanced Technology (IJEAT), 2014, vol. 4, no 2.

BARCO, PATRICIO MARTÍNEZ, ET AL. Sistemas de pregunta-respuesta, 2007.

BIRD, Steven; LOPER, Edward. NLTK: the natural language toolkit. En Proceedings of the ACL 2004 on Interactive poster and demonstration sessions. Association for Computational Linguistics, 2004. p. 31.

BLOMQVIST, EVA. The use of Semantic Web technologies for decision support–a survey. Semantic Web, 2014, vol. 5, no 3, p. 177-201.

BOLLOJU, NARASIMHA; CHAKRABARTI, SUJIT KUMAR. Designing Software Engineering Courses for Effective Teaching and Learning. En Proceedings of the 10th Innovations in Software Engineering Conference. ACM, 2017. p. 220-220.

CAÑAS, C., ET AL. Model of requirements for technological renewal of energy market management systems in Colombia. Revista Ingenierías Universidad de Medellín, 2016, vol. 15, no 28, p. 83-102.

CHÁVEZ ARCE, JAVIER ALEJANDRO. *Buscador inteligente basado en el comportamiento semántico y lenguaje natural en la Web*. 2016. Tesis Doctoral.

CHEN, YING; ARGENTINIS, JD ELENEE; WEBER, GRIFF. IBM Watson: how cognitive computing can be applied to big data challenges in life sciences research. *Clinical therapeutics*, 2016, vol. 38, no 4, p. 688-701.

CHEN, ZHIXIONG; MARX, DELIA. Experiences with Eclipse IDE in programming courses. Journal of Computing Sciences in Colleges, 2005, vol. 21, no 2, p. 104-112.

CHOLLET, FRANCOIS. *Deep learning with python*. Manning Publications Co., 2017.

COSTA, ROSALINA PISCO. Knockin'on Digital Doors: Dealing with Online [Dis] Credit in an Era of Digital Scientific Inquiry. En Establishing and Evaluating Digital Ethos and Online Credibility. IGI Global, 2017. p. 46-65.

DACONTA, MICHAEL C.; OBRST, LEO J.; SMITH, KEVIN T. *The Semantic Web: a guide to the future of XML, Web services, and knowledge management*. John Wiley & Sons, 2003.

DUBEY, MOHNISH, ET AL. Asknow: A framework for natural language query formalization in sparql. En International Semantic Web Conference. Springer, Cham, 2016. p. 300-316.

EMIR, BURAK; ODERSKY, MARTIN; WILLIAMS, JOHN. Matching objects with patterns. European Conference on Object-Oriented Programming. Springer, Berlin, Heidelberg, 2007. p. 273-298.

FROSINI, RICCARDO, ET AL. Flexible query processing for SPARQL. *Semantic Web*, 2017, vol. 8, no 4, p. 533-563.

FUENTES, JOSÉ RUBÉN LAÍNEZ. Desarrollo de Software Ágil: Extremme Programming y Scrum. IT Campus Academy, 2015.

GARCÍA, ESPERANZA ALBACETE. *An ontology for human-like interaction systems*. 2016. Tesis Doctoral. Universidad Carlos III de Madrid.

GÓMEZ, BÁRBARO GUILLERMO BARROSO. Plataforma para la creación de boletines informativos para el centro Telemática de la UCI, 2016.

HEFFELFINGER, DAVID R. *Java EE 7 Development with NetBeans 8*. Packt Publishing Ltd, 2015.

HEIMLICH, ORI; EZRA TSUR, ELISHAI. openVX-Based Python Framework for Real-time Cross-Platform Acceleration of embedded Computer Vision Applications. Frontiers in ICT, 2016, vol. 3, p. 28.

HÖFFNER, KONRAD, ET AL. Survey on challenges of question answering in the semantic web. Semantic Web, 2017, vol. 8, no 6, p. 895-920.

HOLOVATY, ADRIAN; KAPLAN-MOSS, JACOB. The definitive guide to Django: Web development done right. Apress, 2009.

BEATI, HERNÁN. HTML5 y CSS3-Para diseñadores. Alfaomega Grupo Editor, 2016.

ISLAM, **QUAZI NAFIUL.** *Mastering PyCharm*. Packt Publishing Ltd, 2015.

JENA, APACHE. the Apache Jena project. 2014.

JOLION, JEAN-MICHEL; ROSENFELD, AZRIEL. A pyramid framework for early vision: multiresolutional computer vision. Springer Science & Business Media, 2012.

KADAMBI, SHREESHA, ET AL. *Automatically generating question-answer pairs during content ingestion by a question answering computing system*. U.S. Patent No 9,330,084, 3 mayo 2016.

KANG, YONG-BIN; HAGHIGHI, PARI DELIR; BURSTEIN, FRADA. CFinder: An intelligent key concept finder from text for ontology development. *Expert Systems with Applications*, 2014, vol. 41, no 9, p. 4494-4504.

KEFALAKIS, NIKOS, ET AL. A visual paradigm for IoT solutions development. En *Interoperability and Open-Source Solutions for the Internet of Things*. Springer, Cham, 2015. p. 26-45.

KNOWLTON, JIM. *Python*. tr: Fernández Vélez, María Jesús (1 edición). Anaya Multimedia-Anaya Interactiva. ISBN 978-84-415-2513-9, 2009.

KNUDSEN, JONATHAN; DIRECTOR-ZUKOWSKI, JOHN. *Wireless Java: developing with J2ME*. APress LP, 2003.

KNUTH, MAGNUS, ET AL. The DBpedia Events Dataset. En International Semantic Web Conference (Posters & Demos). 2015.

KOIVUNEN, MARJA-RIITTA; MILLER, ERIC. W3c semantic web activity, 2001. Acesso em, 2015, vol. 25.

KOWALSKI, G. Information retrieval systems: theory and implementation, Computers and Mathematics with Applications, vol. 5, No. 35, pp. 133,1998.

KSHIRSAGAR, DEEPAK; KUMAR, SANDEEP. HTTP Flood Attack Detection using Ontology. Proceedings of the International Conference on Advances in Information Communication Technology & Computing. ACM, 2016. p. 15.

KULKARNI, PRIYANKA; JOGLEKAR, YASHADA. Generating and analyzing test cases from software requirements using nlp and hadoop. *International Journal of Current Engineering and Technology (INPRESSCO)*, 2014.

KUNA, HORACIO DANIEL, ET AL. Expansión de consultas basada en ontologías para un sistema de recuperación de información. En XVI Workshop de Investigadores en Ciencias de la Computación, 2014.

LARMAN, C. UML y Patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado. Segunda. s.l.: Prentice Hall, 2004. pág. 520.

LEHMANN, JENS, ET AL. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia**.** *Semantic Web*, 2015, vol. 6, no 2, p. 167-195.

LÓPEZ, E., S. Ecología y libertad. [En línea]. ATIX, Revista Digital, 2010. [Citado el: 14 de mayo de 2018.]. Disponible en: [http://osl.ugr.es/descargas/atix16.pdf].

LIBHUNT. A python framework to transform natural language questions to queries in a database query language. Disponible: https://python.libhunt.com/quepy-alternatives [Último acceso: 22 mayo 2018], 2018.

LÓPEZ, JOSÉ ANTONIO AMARO; ACEVES, LÁZARO MARCOS CHÁVEZ; NAVARRO, GERARDO ALBERTO VARELA. La Web Oculta y cómo los buscadores encuentran la información. *PAAKAT: Revista de Tecnología y Sociedad*, 2014, no 7.

LUMNIFY. BETA: Code Quality Rank. Disponible: http://www.lumnify.com [Último acceso: 22 mayo 2018], 2018.

MACIÁ DOMENE, FERNANDO; GOSENDE GRELA, JAVIER. Posicionamiento en buscadores. Madrid, España: Ediciones Anaya Multimedia, 2009.

MACK, CHRIS. The multiple lives of Moore's law. IEEE Spectrum, 2015, vol. 52, no 4, p. 31-31.

MARTÍN. Free Active Directory Manager (fadmanager). 3C TIC, 2016, vol. 5, no 1, p. 39.

MARKOFF, JOHN. Start-up aims for database to automate web searching. *The New York Times*, 2007, p. 03-09.

MENDOZA, YOEDUSVANY HERNÁNDEZ; GONZÁLEZ, MAITÉ MARTÍNEZ; JAIME, ELVIS MANUEL

MERIÑO, JOSE. Simulador para el protocolo de comunicación industrial AB Ethernet. 2015.

MORENO, MELO; JESNETH, LEIDY. Propuesta de diseño para la transición del protocolo de internet versión 4 (IPV4) al protocolo de internet versión 6 (IPV6) en la empresa MARKET MIX SAS. 2015.

NAGDEVE, MR PRATIK V.; KARALE, SHIVKUMAR J. Natural Language Interface for Casual User to Retrieving the Information from Ontologies. 2016.

NAVARRO MARSET, RAFAEL. Modelado, Diseño e Implementación de Servicios Web. (2007). [Consultado en noviembre del 2017]. Disponible en: http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf

NIETO, ABDIEL MATOS; RUEDA, EYERIS RODRÍGUEZ. ORION: UN MOTOR DE BÚSQUEDAS PARA LA WEB DE LA UCI. 2010.

PELIKÁNOVÁ, ZUZANA. Google Knowledge Graph. 2014.

PRESSMAN ROGER, S. Ingeniaría del Software, 5ta Edición, Edit. 2005.

PYDEV-PYTHON, I. D. E. for Eclipse. 2018.

RAO, CHINTA SOMESWARA; RAJU, K. BUTCHI. Single and Multiple Pattern String Matching Algorithm. Indian Journal of Science and Technology, 2017, vol. 10, no 3.

REDDY, K. SIVA PRASAD. Spring Boot Essentials. En *Beginning Spring Boot 2*. Apress, Berkeley, CA, 2017. p. 47-53.

REESE, RICHARD M. Natural language processing with Java. Packt Publishing Ltd, 2015.

REYNOSO, C. (2004). Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. 2004.

RODRÍGUEZ LEYVA, PAÚL, ET AL. Componentes y funcionalidades de un sistema de recuperación de la información. Revista Cubana de Ciencias Informáticas, 2016, vol. 10, p. 150-162.

ROMÁ, TERESA. Los sistemas de recuperación de la información (SRI) de las bases de datos documentales y la calidad de los resultados obtenidos, 2014.

SÁNCHEZ, EVELÍN PERDOMO, ET AL. ANÁLISIS DE LOS PROCESOS DE LEMATIZACIÓN Y ESTEMIZADO EN LINGÜÍSTICA COMPUTACIONAL. 2017.

SANTOVENIA DÍAZ, JAVIER; CAÑEDO ANDALIA, RUBÉN. 2x3: el primer buscador cubano en Internet. *Acimed*, 2007, vol. 15, no 5, p. 0-0.

SARDA, Deepak. Python for the Busy Java Developer, 2017.

SIAHAAN, ANDYSAH PUTERA UTAMA. Base64 Character Encoding and Decoding Modeling. 2017.

STALLMAN, RICHARD M. GNU coding standards. 2015.

SULÉ, ANDREU, ET AL. Aplicación del modelo de datos RDF en las colecciones digitales de bibliotecas, archivos y museos de España. Revista española de Documentación Científica, 2016, vol. 39, no 1, p. 121.

SUN, HUAN, ET AL. Open domain question answering via semantic enrichment. En Proceedings of the 24th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2015. p. 1045-1055.

UJHELYI, ZOLTÁN, ET AL. EMF-INCQUERY: An integrated development environment for live model queries. Science of Computer Programming, 2015, vol. 98, p. 80-99.

UNGER, CHRISTINA, ET AL. Question answering over linked data (QALD-4). En Working Notes for CLEF 2014 Conference. 2014.

URIBE, ÁNGELA CAMARGO; MARTÍNEZ, CHRISTIAN HEDERICH. Psicología cognitiva en la idea del procesamiento de la información. *Revista Folios*, 2017, no 8.

VALENCIA, FAUSTO R.; ARCOS, HUGO N. Unified Process Applied to the Implementation of the Finite Difference Time Domain Method in Python through Object Oriented Programming. Revista Técnica Energía, 2018, no 14.

WARREN, JOSEPH, ET AL. Messaging over http protocol for data exchange. U.S. Patent Application No 13/955,863, 5 Feb. 2015.

WATANABE, MIZUKI; KOBAYASHI, RYOTARO; KATO, MASAHIKO. HTTP-GET Flood Prevention Method by Dynamically Controlling Multiple Types of Virtual Machine Resources. Journal of Information Processing, 2015, vol. 23, no 5, p. 655-663.

WITBROCK, MICHAEL J., ET AL. Cyc and the Big C: Reading that Produces and Uses Hypotheses about Complex Molecular Biology Mechanisms. En *AAAI Workshop: Scholarly Big Data*. 2015.

WONG, JC CHEANG. Ley de Moore, nanotecnología y nanociencias: síntesis y modificación de nanopartículas mediante la implantación de iones. Rev. Digit. Univ, 2005, vol. 6, no 7, p. 10.

WORLD WIDE WEB CONSORTIUM, ET AL. RDF 1.1 concepts and abstract syntax. 2014.

YAO, XUCHEN; BERANT, JONATHAN; VAN DURME, BENJAMIN. Freebase QA: Information extraction or semantic parsing. En *Proceedings of ACL*. 2014.

YATES, ANDREW, ET AL. The Ensembl REST API: Ensembl data for any language. Bioinformatics, 2014, vol. 31, no 1, p. 143-145.

ZHANG, LEI; RETTINGER, ACHIM; THOMA, STEFFEN. Bridging the gap between cross-lingual NLP and DBpedia by exploiting Wikipedia. *NLP & DBpedia*, 2014.