ENRIQUECIMIENTO SEMÁNTICO DEL CONTENIDO DE UN LMS PARA CONSUMIR DATOS ABIERTOS DISPONIBLES EN LA WEB DE DATOS

GÓMEZ HERNÁNDEZ PABLO LUIS

Cód. 201255393

TAMAYO VALENCIA MARIA ALEJANDRA

Cód.200956134

UNIVERSIDAD DEL VALLE SEDE TULUÁ

PROGRAMA DE INGENIERÍA DE SISTEMAS

TULUÁ - VALLE

MAYO, 2015

ENRIQUECIMIENTO SEMÁNTICO DEL CONTENIDO DE UN LMS PARA CONSUMIR DATOS ABIERTOS DISPONIBLES EN LA WEB DE DATOS

GÓMEZ HERNÁNDEZ PABLO LUIS

Cód. 201255393

TAMAYO VALENCIA MARIA ALEJANDRA

Cód.200956134

Trabajo de grado para optar al título de

Ingeniero de Sistemas

Director: OSCAR ORLANDO CEBALLOS ARGOTE, M.Sc.

Co-director: ALBEIRO APONTE VARGAS, M.Sc.

UNIVERSIDAD DEL VALLE SEDE TULUÁ

PROGRAMA DE INGENIERÍA DE SISTEMAS

TULUÁ - VALLE

MAYO, 2015

Nota de Aceptación

Presidente del jurado

Jurado 1

Jurado 2

Tuluá, Mayo de 2015

Resumen

Actualmente, un gran número de sitios Web enfocados al ámbito educativo, como el campus virtual de instituciones educativas o de plataformas educativas gubernamentales, se gestionan a través de los *Learning Management Systems* (LMS). Tradicionalmente los datos publicados en estos sistemas no se modelan con una estructura y semántica apropiada para permitir que entidades individuales como documentos de texto, imágenes, videos y archivos de audio, entre otros, puedan ser conectados o enlazadas a otras entidades que se encuentran publicados en la Web de Datos. En los últimos años, la evolución de la Web ha considerado un conjunto de mejores prácticas para publicar y enlazar datos de forma estructurada; estas mejores prácticas se conocen como *Linked Data*. Esta propuesta aborda el problema sobre cómo enriquecer semánticamente los contenidos publicados en un LMS, extraer los temas centrales alrededor de los cuales giran dichos documentos para consumir datos abiertos disponibles en la Web de Datos que sean relevantes a los contenidos mencionados. Finalmente se generó un prototipo que consigue integrar el proceso de Extracción de Información y de consulta a la Web de Datos en la plataforma Moodle, que genera resultados de hasta un 72,62% de concordancia en los resultados de la consulta.

Palabras clave: Web de Datos, Sistemas de Gestión de Contenidos, Datos Enlazados Abiertos, Reconocimiento de Entidades Nombradas, Moodle.

Abstract

Nowadays, a large amount of educational environment-driven web sites, such as virtual campuses of educative institutions or government educational platforms, are managed by Learning Managing Systems or LMS. Published data in these systems is usually not modeled with correct semantic and structure to allow individual entities as text files, images, videos and audio files among others be connected or linked to other entities already published on the Data Web. In the last few years, Web evolution has considered a set of better practices known as Linked Data. This initiative takes on the problem of semantic enrichment for published contents from a LMS, extracting the central topics regarding those documents to consume open available data from Data Web related and relevant to the mentioned contents. Finally, a prototype that manages to put the Information Extraction and Data Web query together was generated and integrated into Moodle's platform. This prototype is able to achieve up to a 72,62% matching rate on the results from the query.

Keywords: Web Data, Content Management System, Linked Open Data, Named Entity Recognition, Moodle.

Dedicatoria

A mis padres José Edier Gómez Espinal e Hilda Hernández Ruiz que hicieron todo en la vida para que yo pudiera lograr mis sueños, por motivarme y darme la mano cuando sentía que el camino se terminaba, a ustedes por siempre mi agradecimiento.

Pablo Luis Gómez Hernández

A Dios el ser que hace posible mi vida y que me da fortaleza para continuar cada día. Con todo mi cariño y mi amor para mis padres Samuel Antonio Tamayo Orrego y Luz Enidt Valencia Orrego, las personas que hicieron todo en la vida para que yo pudiera lograr mis sueños, a ustedes les agradezco por siempre y les dedico este trabajo que con sacrificio hemos logrado culminar juntos.

A mi hermana Luz Adriana Tamayo Valencia, por acompañarme y apoyarme en aquellos momentos de mucho trabajo y por ser mi ejemplo de tenacidad y lucha cada día.

A mi prometido Jorge Alejandro Narváez Giraldo que con su paciencia y comprensión, me ha apoyado y enseñado a lo largo de este camino. Gracias por estar siempre a mi lado.

A mi abuelo José Tamayo que siempre me ha dado un gran ejemplo de sabiduría y de fortaleza, gracias por todas tus enseñanzas. A todos mis tíos y primos, que han estado siempre apoyándome y animándome para seguir adelante siendo un gran ejemplo de vida y de superación para mí.

Maria Alejandra Tamayo Valencia

Agradecimientos

Agradecemos a nuestras familias que con su cariño y comprensión siempre estuvieron motivándonos y apoyándonos para culminar nuestros estudios profesionales con éxito.

Agradecemos especialmente al Profesor Oscar Orlando Ceballos que como director nos ha orientado, apoyado y corregido, con un gran interés y una gran entrega. También agradecemos al Profesor Albeiro Aponte que como co-director nos brindo su apoyo y acompañamiento durante este proceso.

A la Universidad del Valle Sede Tuluá por brindarnos la oportunidad de estudiar y de ser profesionales con una excelente calidad académica.

También agradecemos a nuestros profesores que durante toda la carrera nos han aportado su conocimiento en pro de nuestra formación profesional, y que nos han compartido sus experiencias profesionales.

A todas las personas que han formado parte de nuestra vida profesional, agradecemos su amistad, consejos, apoyo, ánimo y compañía a lo largo de esta gran experiencia.

Tabla de Contenido

Listado de Tablas

Lista de Figuras

Lista de Anexos

Capítulo 1

Introducción

1.1 Descripción General

En los últimos años se han incorporado nuevos conceptos como el de *Linked Open Data* (LOD) [13], el cual ha generado una evolución en la Web de datos, este término surgió del uso de *Linked Data*[1](LD) con datos abiertos, cuyo objetivo es que las personas puedan compartir datos y que a su vez estos datos puedan relacionarse con otros sin importar su tipo, logrando una redistribución y reutilización de los mismos de forma libre. LOD permite ver la Web como un espacio global de datos que se conectan entre sí, sin importar de donde provenga esta información [4].

Por otra parte se encuentran los *Content Management Systems* (CMS) [5] que son un software que permite la gestión de contenidos Web y que partiendo de sus formas de aplicación surgen los sistemas *e-learning* el cual se centra en la educación a distancia, dando lugar a los *Learning Management Systems* (LMS) [5] (e.g., recursos, documentos y pruebas evaluadoras, entre otros), que permiten la publicación y administración de contenido educativo.

En la actualidad diversas plataformas educativas se gestionan a través de un LMS (e.g., Blackboard [2], Moodle [3], A Tutor [4]) pero los datos publicados en estas plataformas no tienen la estructura semántica adecuada para poder relacionarse con los datos que se encuentran publicados en la Web de Datos. Por ejemplo, el contenido de un curso de programación no podría relacionarse con otros recursos como videos tutoriales, ejer-

[1] http://linkeddata.org/home

[2] www.blackboard.com/

[3] https://moodle.org/

[4] atutor.ca/

cicios o wikis que se encuentran disponibles en la Web de datos. Por lo anterior, esta propuesta se centra en enriquecer semánticamente los contenidos publicados en un LMS con los datos abiertos disponibles en la Web de Datos aplicando los principios de LOD, para facilitar el proceso de aprendizaje y ayudar a la comprensión de los temas que se encuentran publicados por este medio.

Para el desarrollo de dicho estudio se plantea una metodología de investigación cuantitativa evaluativa, con la cual se pretende solucionar los objetivos planteados, en busca de dar solución a una situación problemática. Esta metodología se caracteriza por ser de tipo exploratorio, es decir, de recopilación de información teórico práctica, ya que no se cuenta con modelos guías para dar una solución.

1.2 Problema

1.2.1 Descripción del Problema

En los últimos años, la evolución de la Web ha considerado un conjunto de mejores prácticas para publicar y enlazar datos de forma estructurada; estas mejores prácticas se conocen como *Linked Data*[5](LD). El objetivo que persigue *Linked Data* es que las personas puedan compartir datos estructurados y distribuidos en la Web, de tal manera que, se forme una relación o vínculo entre estos. La importancia de *Linked Data*, precisamente, se centra en que el valor de los datos aumenta a medida que se relacionan con otros. Sin embargo, *Linked Data* no significa que los datos involucrados sean datos abiertos o que puedan reutilizarse y redistribuirse libremente, sino que están relacionados con otros datos sin importar su tipo. De la misma manera, el hecho de que existan datos abiertos en la Web, no significa que están necesariamente conectados entre sí. Dentro de *Linked Data* los datos pueden no ser completamente abiertos. El uso exclusivo de *Linked Data* con datos abiertos se denomina *Linked Open Data* (LOD) [13].

La adopción del término LOD ha generado la extensión de la Web como un espacio global de datos que conecta datos de dominios diferentes (e.g., gente, compañías, libros, publicaciones científicas, música) provenientes de diversas fuentes [4]. Importantes organizaciones se han sumado a la idea de publicar datos abiertos en la Web siguiendo las buenas prácticas de LOD, como por ejemplo, la *BBC*[6] *, Thomson Reuters*[7]

[5]http://linkeddata.org/home

[6]http://www.bbc.co.uk/nature/feedsanddata

[7]https://customers.reuters.com/rmds/CZRDP/ECOPages/DataSources.aspx

y la Library of Congress[8], la *Biblioteca Nacional de España*[9], entre otras; aportando a la construcción de un espacio global de datos sobre personas, compañias, libros, publicaciones científicas, películas, música, programas de radio y televisión, genes, proteinas, fármacos y ensayos clínicos, comunidades en línea, datos estadísiticos y científicos, que en el año de 2014, según la *W3C*[10], se estimaban en 1014 conjuntos de datos con más de 32.000 millones de tripletas en *Resource Description Framework* [11] (RDF) con 570 millones de enlaces entre ellas [12] .

Por otra parte, los *Content Management Systems*(CMS) [5] son software cuya principal función es la de facilitar la gestión sitios de web, ya sea en Internet o en una intranet, y por esa razón es posible referirse a ellos también como *Web Content Management* (WCM) [5]. A partir de los CMS y sus formas de aplicación surgen los sistemas *e-learning*; en estos sistemas, la gestión se centra en los contenidos educativos (e.g., gráficos, mapas mentales, videos, diapositivas, audios, pruebas evaluadoras) y no propiamente en la Web. El enfoque de los sistemas *e-learning* va dirigido a cubrir ciertas necesidades concretas (e.g., mayor velocidad de creación de contenidos académicos con menor coste, personalización del aprendizaje, incremento de la calidad en la atención del estudiante, generación de ventajas competitivas) que un CMS general no siempre cubre, o si lo hace, no da las mismas facilidades que una herramienta creada para realizar esta función; esto da lugar a los *Learning Management Systems* (LMS) [5] como sistemas de *e-learning* orientados a gestionar contenidos educativos, brindando herramientas que permiten la distribución de cursos, recursos, noticias y contenidos relacionados con la formación, cuyo fin es mejorar las competencias de los usuarios relacionados con los cursos.

Actualmente, un gran número de sitios Web enfocados al ámbito educativo, como el campus virtual de instituciones educativas (e.g., campus virtual de la Universidad del Valle[13]) o de plataformas educativas gubernamentales (e.g., Sena virtual[14]), se gestionan a través de los LMS. Tradicionalmente los datos publicados en estos sistemas no se modelan con una estructura y semántica apropiada (i.e., en un formato RDF) para permitir que entidades individuales como documentos de texto, imágenes, videos y archivos de audio, entre otros, puedan ser conectadas o relacionadas a otras entidades que se encuentran publicadas en la Web [4]. La figura 1.1 muestra un ejemplo esquemático del problema, en el lado (a) muestra el funcionamiento actual de Moodle y el lado (b)

[8]http://www.loc.gov/standards/mads/rdf/

[9]http://datos.bne.es/

[10] http://lod-cloud.net/state/

[11]http://www.w3c.org/RDF

[12]http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/

[13]http://campusvirtual.univalle.edu.co

[14] http://sena.blackboard.com/webapps/portal/frameset.jsp

muestra el grafo de la Web de Datos al año 2014 http://lod-cloud.net/

Por ejemplo, un profesor publica contenido de un curso de programación en diversos formatos (i.e., pdf, .doc, jpg, .rar, entre otros) y de diversas fuentes (i.e., URL, libros, articulos, etc.), cuando un estudiante obtiene esta información y desea ampliarla debe consultar otros materiales de estudio (e.g., videos tutoriales, ejercicios, ejemplos o wikis que se encuentran disponibles en la web), y esto debe hacerlo mediante una búsqueda en la Web por medio de buscadores (e.g., Google[15], Yahoo[16], Bing[17], entre otros), obteniendo como resultado información que usualmente no está relacionada con su consulta (ver figura 1.1).

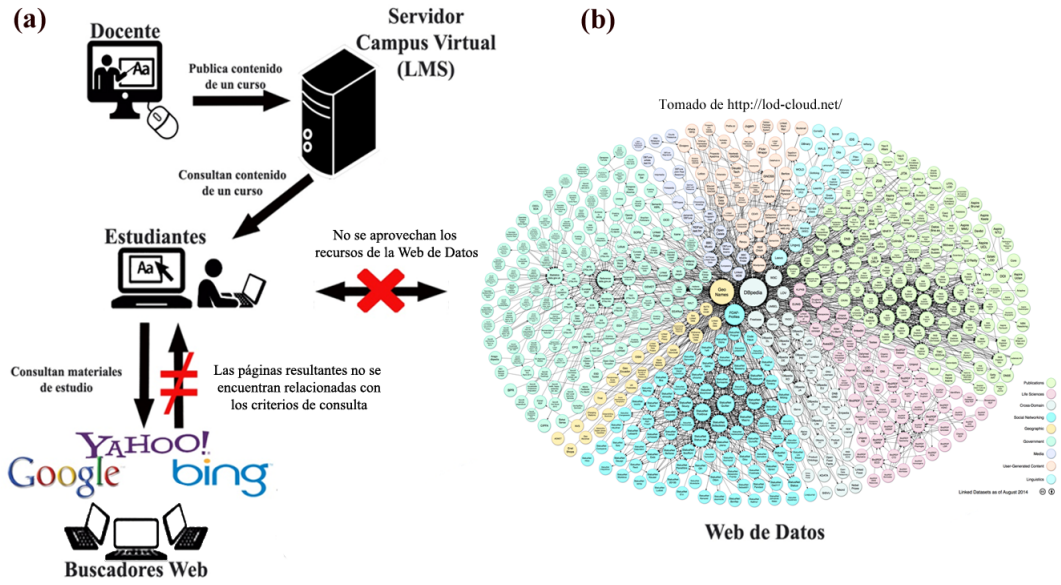


Figura 1.1: Problema objeto de estudio.

1.2.2 Formulación del Problema

¿Cómo enriquecer semánticamente los contenidos publicados en un *Learning Management System* para consumir datos abiertos disponibles en la Web de Datos?

[15]https://www.google.com.co/

[16]http://co.yahoo.com/

[17]https://www.bing.com/

1.3 Objetivos

1.3.1 Objetivo General

Enriquecer semánticamente los contenidos publicados en un *Learning Management Systems* (LMS) para consumir datos abiertos disponibles en la Web de Datos.

1.3.2 Objetivos Específicos

Objetivo	Sección
Usar las técnicas de NER (Named Entity Recognitions) para la clasificación y extracción de entidades de los LMSs.	3.1
Modelar en formato RDF las entidades individuales publicadas en un LMS para que puedan ser relacionadas a otras entidades disponibles en la Web de Datos por medio de una consulta en SPARQL.	3.2
Extender un LMS existente, a manera de prototipo, para acceder, consultar, buscar y combinar recursos aplicando los principios de LOD para consumir datos de la Web de Datos.	3.3
Adaptar la arquitectura de un LMS existente a la arquitectura de un SCMS para la gestión e integración de contenidos enriquecidos semánticamente.	3.3.1

Tabla 1.1: Objetivos específicos con los resultados obtenidos.

1.3.3 Objetivo Adicional

Objetivo	Sección
Ampliar la detección y la consulta para incluir temas relacionados con Física y añadir soporte para el idioma Inglés.	3.4

Tabla 1.2: Objetivo adicional.

1.4 Estructura del Documento

En el capítulo 2 se presenta el Marco de Referencia, donde se explican conceptos como: *Content Management System*, *Learning Management System*, *Linked Open Data*, *Extracción de Información*, entre otros; requeridos para entender el tema planteado. Además se presenta la estructura y composición actual de la plataforma Moodle, explicando la arquitectura, las herramientas con las que cuenta la plataforma, la estructura de la base de datos y el estándar de programación.

En el capítulo 3 se presenta lo relacionado con el enriquecimiento semántico del contenido de un LMS para consumir datos abiertos disponibles en la Web de Datos; es decir, el proceso de Extracción de Información, la transformación semántica de las entidades extraidas, la consulta a DBpedia, la adptación de la plataforma Moodle para su extensión y los resultados obtenidos en la implementación.

Finaliza con el Capítulo 4, donde se dan algunas conclusiones y se describen los trabajos futuros.

1.5 Publicación

Parte de este trabajo de grado ha sido publicado en:

M. Tamayo y O. Ceballos. Enriquecimiento Semántico del contenido de un LMS para consumir datos abiertos disponibles en la Web de Datos. Póster en III Encuentro Departamental de Semilleros de Investigación Nodo Valle del Cauca. Red Regional de Semilleros de Investigación - RREDSI. Mayo de 2014. Cartago, Colombia.

También sometido a revisión de resumen como:

M. Tamayo, P. Gómez y O. Ceballos. Extensión semántica de Moodle aplicando técnicas de NER y consumiendo datos de la Web de Datos. 10CCC Congreso Colombiano de Computación. Septiembre de 2015. Bogotá, Colombia.

Capítulo 2

Marco Referencial

2.1 Marco Teórico

En la actualidad existen herramientas que ayudan a construir y administrar sitios Web de manera fácil y rápida, permitiendo realizar actividades que, normalmente, las realizarían profesionales especializados en diseño, desarrollo, administración y mantenimiento de sitios Web [2]. Estas herramientas se conocen como *Content Management Systems* (CMS) y tienen por objetivo principal la creación y administración dinámica de información (e.g., texto, imágenes, videos, sonido, etc.) en una página Web [5]. La Figura 2.1 muestra la arquitectura de un CMS.

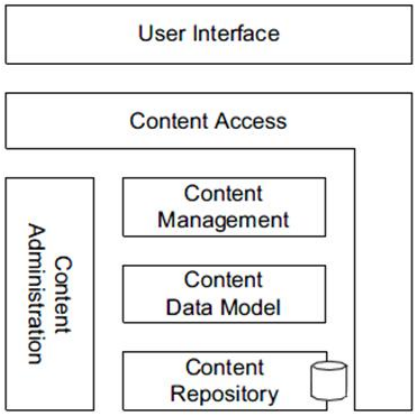


Figura 2.1: Arquitectura de un CMS.
Tomada de [7]

La arquitectura de un CMS está construida sobre la base de seis capas: *i*) *User Interface* a través de la cual se presenta el contenido para su modificación, *ii*) *Control Access* para que la capa anterior pueda tener acceso al contenido, *iii*) *Content Management* que permite definir el dominio o la aplicación de un modelo de datos específico, *iv*) *Content Data Model* que especifica el modelo sobre el cual se basa el almacenamiento del contenido, *v*) *Content Repository* que define el mecanismos de persistencia para el modelo de datos de contenido y *vi*) *Content Administration* que permite administrar la pila del CMS [7]. Ejemplos de CMS son: Joomla! [1], Drupal [2], Snews [3], Plone [4].

Los CMS también se caracterizan por no tener herramientas elaboradas de colaboración (e.g., foros, chats, diarios) ni apoyo en tiempo real. Su aplicación no solo se limita a la gestión dinámica de la información, con la aparición del concepto de *e-learning* [5] basado en Web, los CMS tomaron una distinta especialización, se orientaron a la gestión de contenidos para el aprendizaje a distancia. La evolución de los CMS hacia el *e-learning* tiene unas necesidades concretas (e.g., mayor velocidad de creación de cotenidos académicos con menor coste, personalización, incremento de la calidad en la atención del estudiante, generación de ventajas competitivas) que no siempre se cubre, o si la cubre no otorga las mismas facilidades que una herramienta creada específicamente para realizar esta función [5]. Por lo tanto, surgen los *Learning Management System* (LMS) [5] para facilitar la gestión de contenidos académicos. Estos proporcionan entornos donde es posible adaptar la formación a las necesidades de una institución o empresa y, al propio desarrollo profesional. Un LMS dispone de herramientas que permiten la creación, administración y distribución de contenidos relacionados con la formación académica. A partir de estas herramientas, surgen los *Learning Content Management Systems* (LCMS) [5] como la integración de las funcionalidades de un CMS y un LMS. La caraterística principal de los LCMS es que proporciona contenidos para el aprendizaje con soporte para la evaluación dinámica y cumplimiento de tareas. La Tabla 2.1 muestra una comparación de las principales características de uso entre los sistemas LMS y LCMS.

[1] http://www.joomla.org/

[2] https://drupal.org/

[3] http://snewscms.com/

[4] http://plone.org/

[5] http://www.e-abclearning.com/definicione-learning

Usos	LMS	LCMS
Usuarios a los que va dirigido	Responsables de los cursos, administradores de formación, profesores o instructores	Diseñadores de contenidos, diseñadores instruccionales, directotres de proyectos
Proporciona	Cursos, eventos de capacitación y está dirigido a estudiantes	Contenidos para el aprendizaje, soporte en el cumplimiento y usuarios
Manejo de clases, formación centrada en el profesor	Si (pero no siempre)	No
Administración	Cursos, eventos de capacitación y estudiantes	Contenidos para el aprendizaje, soporte en el cumplimiento y usuarios
Análisis de competencias-habilidades	Si	Si (en algunos casos)
Informe del rendimiento de los participantes en el seguimiento de la formación	Enfoque principal	Enfoque secundario
Colaboración entre usuarios	Si	Si
Mantiene una base de datos de los usuarios y sus perfiles	No siempre	No siempre
Agenda de eventos	Si	No
Herramientas para la cración de contenidos	No	Si
Organización de contenidos reutilizable	No siempre	Si
Herramientas para la evaluación integrada para hacer exámenes	Si (la mayoría de los LMS tienen esta capacidad)	Si (la gran mayoría tiene esta capacidad)
Herramientas de flujo de trabajo	No	Si (en algunas ocasiones)
Comparte datos del estudiante con un sistema ERP (enterprise requerirment planning)	Si	No
Evaluación dinámica y aprendizaje adaptativo	No	Si
Distribución de contenido, control de navegación e interfaz del estudiante	No	Si

Tabla 2.1: Comparación de características de uso entre LMS y LCMS. Tomada de [5]

Los sistemas LMS y LCMS, aunque complementarios, tienen distintos propósitos. Un LMS permite planificar y administrar los cursos y eventos de capacitación dirigidos por los profesores o instructores; además, ofrecen herramientas para gestionar usuarios, recursos, actividades, módulos, permisos, evaluaciones, calificaciones, comunicación de foros, videoconferencias, chats, entre otros. Por el contrario, un LCMS permite administrar contenidos de aprendizaje de diferentes programas de capacitación que se configuran en el desarrollo en toda una organización. Además, proporcionan a los diseñadores de contenido, diseñadores institucionales y directores de proyecto, los medios para crear y reutilizar el contenido de aprendizaje (i.e., crear, almacenar, ensamblar y entregar de forma personalizada el contenido en forma de objetos de aprendizaje). Un LCMS es similiar a un LMS pero con una particularidad propia de los CMS, el hecho de poder administrar todos los contenidos del sistema.

A pesar que existen un gran número de aplicaciones en el contexto de los LMS (e.g., propietarios: *Blackboard, Desire2Learn, eCollege, Saba Learning, Docebo, Claroline, entre otros*; de código abierto: *Moodle, Sakai, Atutor, Ilias, entre otros*) y LCMS (e.g., propietarios: *Itelligent Web Teacher, SumTotal TotalCMS*; de código abierto: *Atutor*) para instituciones educativas (e.g., campus virtual de la Universidad del Valle[6]) o plataformas educativas gubernamentales (e.g., sena virtual[7]), los datos publicados en estas aplicaciones no se modelan con una estructura y semántica apropiada (i.e., en un formato adecuado) para permitir que entidades individuales como documentos de texto, imágenes, videos y archivos de audio, entre otros, puedan ser conectadas o relacionadas a otras entidades que se encuentran publicadas en la Web [6].

Por otra parte, la visión de una Web, tal y como lo planteó *Tim Berners-Lee* incluye la posibilidad de razonar y sacar conclusiones lógicas de forma automatizada a partir de los datos publicados en la Web. En el desarrollo de las tecnologías para la Web se han definido múltiples lenguajes para representar información y conocimiento, tales como, *Resource Description Framework* [8] (RDF), *RDF Schema* [9] (RDFS) y *Web Ontology Language*[10] (OWL). En particular, RDF es un estándar para el intercambio de datos en la Web que tiene características que facilitan la interconexion de datos, incluso si los esquemas que subyacen a dichos datos difieren unos de otros (e.g. bases de datos relacionales, XML[11]). El lenguaje RDF es útil para describir recursos en la Web, pero se queda corto en la definición de metadatos del esquema en el que se describen estos recursos. Para la definición de esquemas, es necesario hacer uso de RDFS, un conjunto de clases y propiedades definidas usando lenguaje RDF, que proporciona elementos básicos para la descripción de ontologías. Estos elementos permiten describir propiedades y clases de recursos RDF. Además, contiene la semántica necesaria para definir relaciones de herencia o generalización entre clases y propiedades [12].

La especificación de la W3C (*World Wide Web Consortium* - http://www.w3c.org) de RDF define un modelo de datos basado en grafos cuya unidad básica de información se conoce como tripleta. Una tripleta consta de un sujeto, un predicado y un objeto. Para evitar la ambigüedad de la información establecida por una tripleta en particular, el sujeto y el predicado de la tripleta deben ser *Universal Resource Identificator* (URI), el objeto puede ser un valor numérico, un valor literal o una URI. Una URI tiene como finalidad identificar de manera única un recurso particular. Una tripleta puede ser representada por un grafo dirigido, donde los nodos son el sujeto y el objeto, y el arco entre dichos nodos es el predicado.

[6] http://campusvirtual.univalle.edu.co

[7] http://sena.blackboard.com/webapps/portal/frameset.jsp/

[8] http://www.w3c.org/RDF

[9] http://www.w3c.org/TR/rdf-schema

[10] http://www.w3c.org/TR/owl-features

[11] http://www.w3schools.com/xml/

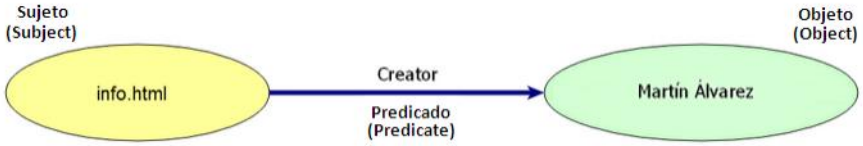


Figura 2.2: Ejemplo de tripleta.
Tomado de http://lod-cloud.net/state/

Figura 2.3: Ejemplo de tripleta con URIs.
Tomado de http://lod-cloud.net/state/

Las tripletas se almacenan en diferentes repositorios de datos, por ejemplo, en *Triple Store* (e.g., Fuseki[12], Virtuoso[13], 4Store[14]), en bases de datos basadas en grafos (e.g., Neo4j[15],) o en la Nube de Datos (i.e., *Linked Data*[16]). Después de almacenar las tripletas se hace necesario un lenguaje de consulta para las mismas. *SPARQL Protocol* and *RDF Query Language* (SPARQL) [12] es un lenguaje estandarizado para la consulta de grafos RDF, normalizado por la W3C. SPARQL se usa para realizar consultas sobre diversas fuentes de datos, sin importar si los datos están almacenados nativamente como RDF o son vistos como RDF a través de un *middleware*(e.g., CORBA[17] para objetos distribuidos). SPARQL consulta patrones, obligatorios u opcionales de grafos con sus conjunciones y disyunciones. Los resultados de las consultas SPARQL pueden ser conjuntos de resultados o grafos RDF [12].

[12]http://jena.apache.org/
[13]http://www.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSRDFWP
[14]www.4store.com/
[15]www.neo4j.org/
[16]linkeddata.org/
[17]http://www.corba.org/

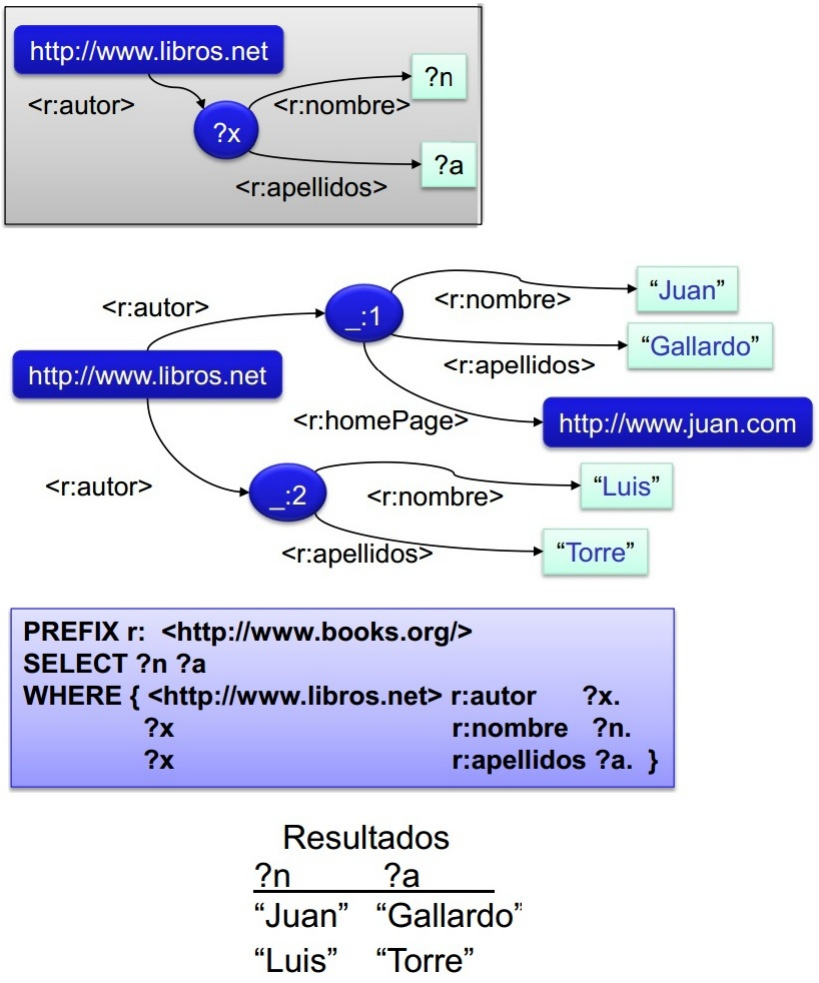


Figura 2.4: Ejemplo de consulta SPARQL.
Tomado de http://di002.edv.uniovi.es/ labra/cursos/XML/SPARQL.pdf

Tim Bernes-Lee estableció un conjunto de reglas para publicar datos estructurados de forma que estos hagan parte de un mismo espacio global, la Web de Datos; a este conjunto de reglas se conoce como principios de *Linked Data* [9]. Estos prinicipios son cuatro: *i*) usar el modelo de datos RDF para publicar datos estructurados en la Web de Datos, *ii*) utilizar enlaces RDF para interconectar datos desde distintas fuentes de datos, *iii*) usar identificadores globales (URIs) para identificar las cosas y *iv*) hacer accesible la información mediante el protocolo de comunicación HTTP URI [3].

El objetivo que persigue *Linked Data* es que las personas puedan compartir los datos estructurados (i.e., datos en RDF) y distribuidos en la web de tal manera que se forme una relación o vinculo. La importancia de *Linked Data* se centra en que el valor de los

datos aumenta a medida que se relacionan con otros. *Linked Data* no significa que los datos involucrados sean datos abiertos o que puedan reutilizarse y redistribuirse libremente, sino que están relacionados con otros datos sin importar su tipo. De la misma manera, el hecho de que existan datos abiertos en la Web, no significa que están necesariamente conectados entre sí. Dentro de *Linked Data* los datos pueden no ser completamente abiertos. El uso exclusivo de *Linked Data* con datos abiertos se denomina Datos Abiertos Enlazados (*Linked Open Data* - LOD) [4]. La Figura 2.5 muestra la Web de Datos en forma de grafo al año 2014.

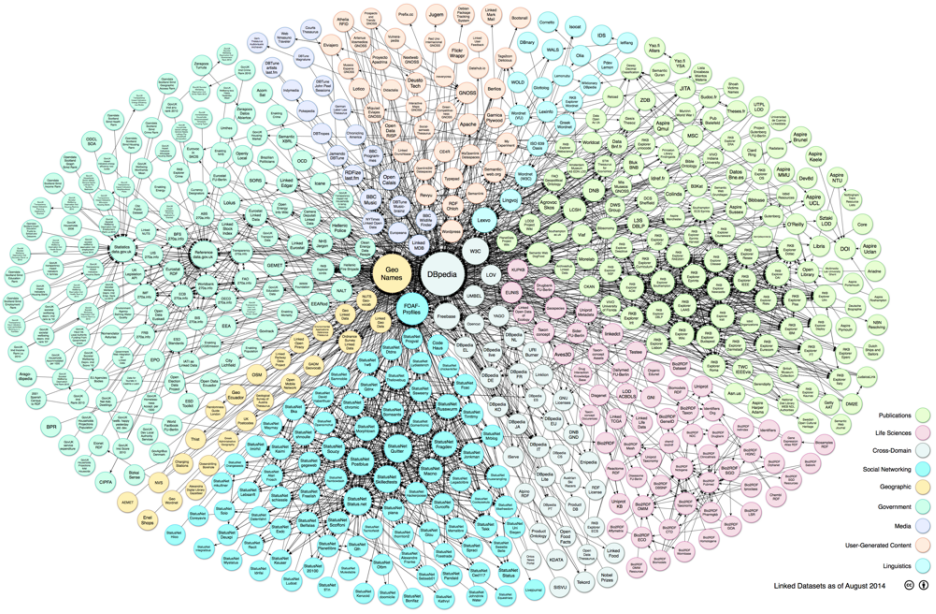


Figura 2.5: Nube de Datos (LOD), al año 2014.
Tomado de http://lod-cloud.net/

Una red neuronal artificial es un modelo de procesamiento inspirado en las redes neuronales biológicas que busca simular el funcionamiento del cerebro en lo que a procesamiento de información respecta. Tiene como ventajas la flexibilidad y el aprendizaje por medio de la experiencia [14].

Una red neuronal artificial se compone de una o varias neuronas y las conexiones entre ellas también denominadas sinapsis. Las conexiones pueden ser de entrada o de salida y no existe límite para la cantidad de conexiones de entrada o salida para una determinada neurona. Existen neuronas de capa de entrada que reciben la información de manera directa y sin modificación, esta información puede ser lineal o no lineal y en el paso a la siguiente neurona esta información se multiplica por el peso sináptico de la

conexión, los pesos sinápticos son valores asignados a las sinapsis según sea la importancia de la misma. La neurona que recibe la información de las neuronas de entrada inicia su proceso verificando su función de activación que consiste en la sumatoria de los valores de entrada tras ser multiplicados por su peso sináptico. Dicha función debe retornar un valor superior al umbral de la neurona, el umbral es un valor asignado a cada neurona para decidir con qué facilidad esta se activa. Si el valor retornado por la función de activación es menor al umbral, la neurona no se activa. Lo que significa que generalmente retornaría 0, o sencillamente no sabría que responder [14].

Una vez que se comprueba que la función de activación sea mayor al umbral, la neurona se activa retornando el resultado de la función de activación, esta función puede ser bipolar o binaria. Bipolar significa que su retorno está entre -1 y 1 y binaria que está entre 0 y 1. Estos valores pueden ser lineales o no lineales ya que tanto las entradas como la salida deben ser la representación numérica de ”algo”.

Un perceptrón simple es una red neuronal unicapa especializada en la detección de patrones, una red unicapa es una red con una capa de entrada, es decir compuesta únicamente por neuronas de entrada, y una capa de salida en donde todas las neuronas de entrada se conectan a todas las neuronas de salida. Las neuronas de salida son no lineales con función de activación tipo escalón y de naturaleza bipolar [14].

El entrenamiento de una red neuronal artificial es el proceso mediante el cual se asignan los valores de los pesos sinápticos que permiten a la red devolver una respuesta correcta. El aprendizaje puede ser supervisado o no supervisado, en el aprendizaje supervisado se le indica a la red en el momento de su entrenamiento cuál debe ser la respuesta correcta para un conjunto de entrada dado. Los pesos sinápticos se asignan inicialmente de manera aleatoria y por medio del entrenamiento se ajustan para lograr la solución correcta, esto consiste en asignar pesos mayores a las entradas que tienen un mayor impacto en la solución y pesos menores a los que tienen menor impacto o son casi irrelevantes [14].

Para lograr la detección de patrones del perceptrón, se entrena para clasificar una entrada o señal en categorías, el número de categorías y los conjuntos de entrada que representan se decide al momento del entrenamiento del perceptrón de manera que su respuesta sea un código que represente la categoría a la que la señal de entrada pertenece, o a la que más parece pertenecer. Si la neurona retorna 0, es decir si no se activa, quiere decir que no sabe a qué categoría pertenece la señal de entrada, generalmente porque la señal no es lo suficientemente similar a alguna de las señales de entrada con las que el perceptrón fue entrenado [14].

La fuente más abundante de información son los textos escritos (e.g., Archivos de Texto, Libros, Páginas Web) y en ellos, por medio del lenguaje natural escrito se transmite la información. Cada lenguaje natural posee elementos sintácticos que forman una estructura gramatical necesaria para la comprensión de los textos como artículos, conectores, preposiciones etc. sin embargo, no todos estos elementos poseen en sí mismos información a diferencia de los sustantivos, adjetivos y adverbios que suelen poseer información como nombres, cantidades y fechas.

La Extracción de Información es un proceso mediante el cual se obtienen, de un texto de entrada, valores numéricos, palabras claves, nombres, terminologías, ideas principales o cualquier otra información que pudiera ser de utilidad. Esta extracción requiere de un proceso iterativo que implica diferentes niveles de reconocimiento según sea el propósito, primero palabra por palabra, luego agrupaciones, relaciones cruzadas entre oraciones diferentes y finalmente ideas o propósito de cada párrafo.

Para conseguir extraer la información necesaria es preciso detectar los patrones que sigue la gramática (i.e., sujeto, verbo, complemento), siguiendo las pistas que ofrecen los elementos de la sintaxis (i.e., artículos y preposiciones) para encontrar dónde puede estar la información (e.g., ”El Banco X, con 500 sedes en Europa...” el artículo ”el” indica que probablemente la siguiente palabra o palabras son un nombre, de la misma forma las preposiciones ”con” y ”en” indican la probabilidad de que en frente de ellas haya información). Generalmente, después de detectadas las partes del texto que contienen la información a ser extraída, se obvian dichos elementos sintácticos ya que, aunque son útiles a la hora de comprender un texto, resultan irrelevantes en lo que a información sustantiva respecta. Por ejemplo de la frase La Compañía X logró 25.000 ventas en el 2013 se etiquetaría Compañía X como nombre de organización, 25.000 como una cantidad numérica y 2013 como una expresión temporal.

Inicialmente, la técnica más sencilla de extracción de información es el *POSTagging* o etiquetado de partes de discurso. En este se evalúan una a una las oraciones del texto y para cada oración se reconocen las palabras para definir si son nombres, artículos, adjetivos o verbos y deducir de éstos el tiempo de la oración y su estructura sintáctica. Suele utilizarse como el primer paso para extraer información de un texto. Implica reconocer la categoría de cada palabra, es decir, si es un verbo, un sustantivo, un artículo, adjetivo, adverbio, etc. Para esto es necesario consultar un corpus[18] de palabras del lenguaje en el que está escrito el texto. Estos suelen estar organizados en forma de árbol donde las hojas son todas las palabras y al ir avanzando hacia el nodo principal se encuentra la palabra raíz de cada hoja y su categoría gramatical(e.g., ”vio” sería una hoja, su palabra raíz es ”ver” y su categoría es verbo). En esta etapa de la extracción son de interés principalmente las palabras raíces y sus categorías. Al requerir recorrer un árbol con cientos o hasta miles de millones de palabras, este proceso es muy costoso computa-

[18]http://www.tei-c.org/release/doc/tei-p5-doc/en/html/CC.html

cionalmente y suele reemplazarse con algún método de reconocimiento de patrones que sea más eficiente.

Tras reconocer la estructura sintáctica de la oración, se extraen los valores nominales. Para esto es necesario reconocer grupos de palabras por su categoría y formar sintagmas[19], por ejemplo, en español: un artículo y un nombre propio forman un sintagma nominal 'el Albatros', aunque este sintagma puede ser mucho más complejo (e.g., algunas de las tardes silenciosas color lapislázuli del mes de Junio que pasé contigo). Estos sintagmas nos interesan porque las entidades suelen tener forma de sintagma nominal (e.g., La Armada Imperial Japonesa). Luego se comparan las entidades mencionadas con un conjunto de entidades conocidas para reconocer lugares, corporaciones o nombres famosos, si la entidad no está listada en el corpus entonces se etiqueta como desconocida. Esta técnica se conoce como Extracción de Entidades Nombradas, o NER [15] por sus siglas en inglés. Debido a que se debe consultar y realizar una búsqueda de palabras en una lista por cada sintagma nominal que se haya etiquetado es muy costoso computacionalmente y por este motivo, la detección se implementa de varias formas buscando una mayor eficiencia. Debido a que este proceso ocurre sobre las oraciones, las oraciones que no contienen información, como interjecciones[20] deben ser ignoradas tras reconocer las entidades nombradas.

En caso de hallar anáforas[21], es preciso relacionarlas con el nombre al que se refieren siempre y cuando dicho nombre haya sido previamente mencionado. Si estas entidades no están definidas en la Web Semántica es entonces necesario reconocerlas y tratarlas de manera que puedan relacionarse con entidades afines.

A la aplicación combinada y coordinada de estas técnicas se conoce como extracción de información. Su implementación varía según sea el propósito de la misma, ya que para lograr un sistema que responda preguntas a partir de la información de un texto escrito en lenguaje natural se requiere un mayor grado de entendimiento que para extraer valores nominales del mismo.

Tras detectar las entidades nombradas, es posible asignarles una definición en el ámbito del texto escrito buscando conectores que las vinculen con definiciones que pueden o no estar presentes en el texto, a partir de las definiciones encontradas, se genera una terminología.

Un ejemplo de cómo se extraerían entidades usando *POSTagging* y NER requeriría leer el texto palabra por palabra, y consultar cada una en el corpus para decidir cuál es su categoría más probable, ya que una palabra puede significar varias cosas y tener difer-

[19] http://de?nicion.de/sintagma/

[20] http://www.retoricas.com/2012/03/ejemplos-de-interjecciones.html

[21] http://www.retoricas.com/2009/06/10-ejemplos-de-anafora.html

entes categorías posibles. Luego, se agrupan los sintagmas nominales para consultarlos como entidades nombradas conocidas, y etiquetar los desconocidos como tal, como se muestra en la Figura 2.6.

Figura 2.6: Ejemplo de NER.

Cabe resaltar que dicho procedimiento no es óptimo, ya que por cada palabra es necesario realizar una búsqueda entre alrededor de quinientos millones de palabras del corpus y por cada sintagma nominal, es preciso realizar otra búsqueda entre los sintag-

mas conocidos que debe ser un conjunto amplio para ser útil de manera flexible y debe mantenerse actualizado para que consiga detectar las entidades nuevas.

Una alternativa más eficiente requeriría obviar el proceso de POSTagging por completo y usar un reconocedor de patrones para definir si una palabra o grupo de palabras parece ser una entidad. Como reconocedor de patrones para realizar un ejemplo concreto de esta alternativa se va a utilizar un perceptrón entrenado para reconocer nombres propios. Esta aproximación añade más flexibilidad al proceso, siendo capaz en casos de detectar entidades nuevas (i.e., nombres de empresas o personas, siglas o acrónimos) por su parecido a entidades existentes, lo que le permite ser vigente sin que sea necesario realizar actualizaciones manuales. Por otro lado, debido a su naturaleza de detección de patrones, es posible que algunos de sus resultados no sean correctos y sea necesario algún filtrado sobre las entidades detectadas, como se muestra en la Figura 2.7

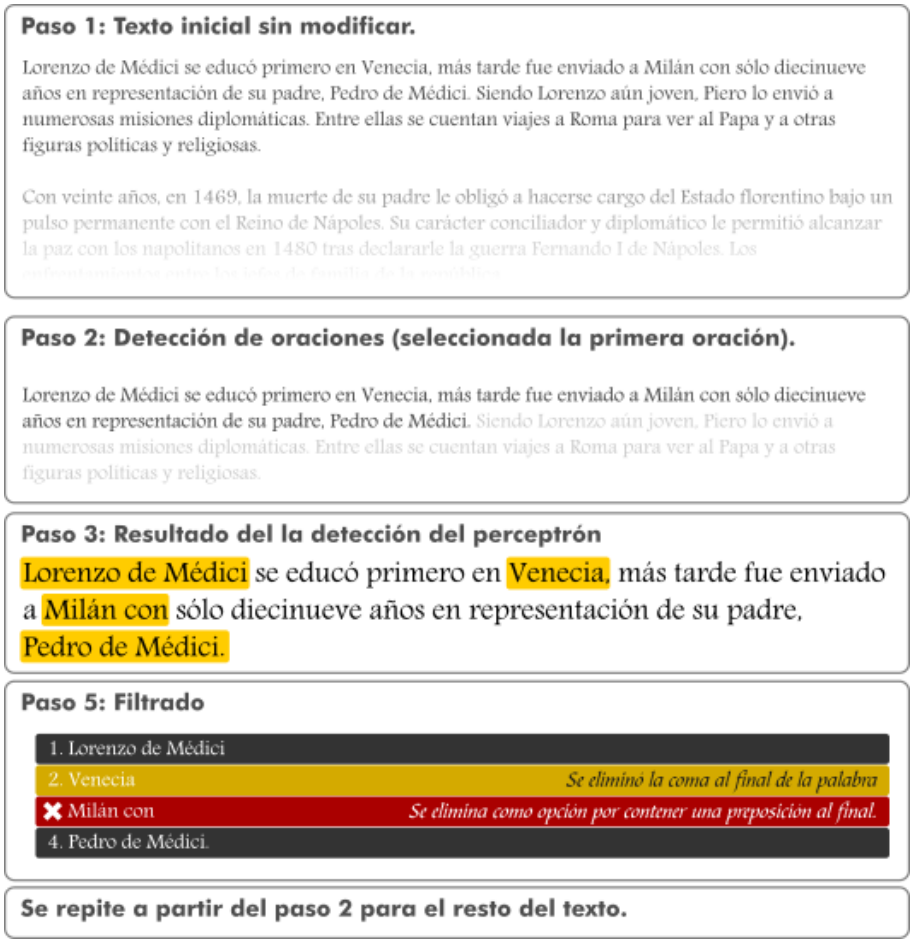


Figura 2.7: Ejemplo de NER funcionando con un Perceptrón.

Inicialmente es necesario separar el texto por oraciones para no sobrecargar al perceptrón, lo que perjudicaría su correcto funcionamiento. El perceptrón está entrenado para detectar nombres propios en un contexto normal de una oración, lo que lo lleva a preferir conjuntos de palabras que inician con mayúsculas. Si embargo, la segunda palabra puede ir en minúscula como en los casos Francisco de Paula Santander y ser tan corta como una o dos palabras (e.g., Einstein o John jr.) debido a esto, el perceptrón puede detectar a "Venecia,", por su parecido con nombres como "Lucrecia", y "Milán con" por contener dos palabras, una con mayúscula inicial y por ser la segunda de tres caracteres. La palabra "Venecia," posee una coma al final que la convertiría en una palabra no válida, esto se puede evitar si se usa un algoritmo para separar estos caracteres que añada espacios antes y después de toda coma o signo de puntuación. Finalmente, se usa un filtro en el que se elimina la coma y se guarda a Venecia unicamente y se descarta "Milán con" ya que al tener la preposición "con" al final, se toma como un error del perceptrón, los demás resultados son correctos, por lo que se los deja tal como están.

2.2 Antecedentes

2.2.1 Modelos de Arquitectura

Entre las aproximaciones reportadas para enriquecer semánticamente la información de un sitio Web, se encuentran entre estos, cambios arquitectónicos por la necesidad de gestionar contenidos semánticos asociados a los recursos, mecanismos de obtención de metadatos, formas de consulta e integración con LOD y, la visualización de los resultados para el usuario final. Abordando cada uno de estos aspectos, en relación a los cambios arquitectónicos, el que un CMS requiera gestionar contenidos semánticos o metadatos representa un impacto en la arquitectura, con el fin de generar un *Semantic Content Management System* (SCMS), otorgando no sólo un espacio de almacenamiento para los contenidos y metadatos, sino también proporcionando mecanismos de extracción, asociación, recuperación de conocimiento y raciocinio. Para esto se proponen dos modelos arquitectónicos; el primero de ellos propone una arquitectura conceptual con las siguientes capas de referencia que se presentan en la Figura 2.8.

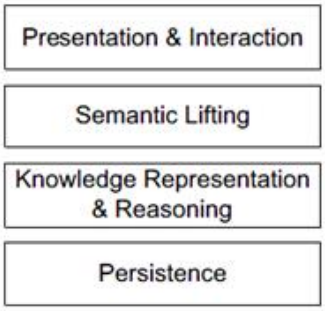


Figura 2.8: Capas de referencia para un SCMS.
Tomada de [7].

Las capas de referencia para un SCMS son cuatro: *i) Presentation & Interaction* que presenta al usuario final el conocimiento y apoya su interacción directa con este, *ii) Semantic Lifting* a través de la cual se extrae el conocimiento, en esta capa un contenido dado se levanta a nivel semántico mediante la extracción de conocimiento de ella, *iii) Knowledge Representation Reasoning* para crear nuevo conocimiento con base en conocimiento existente y para representar este conocimiento dentro de un SCMS y *iv) Persistence* para almacenar el nuevo conocimiento.

La segunda propuesta arquitectónica es en relación a la arquitectura de un CMS y las capas de referencia para un SCMS, la Figura 2.9 muestra esta segunda arquitectura para un SCMS.

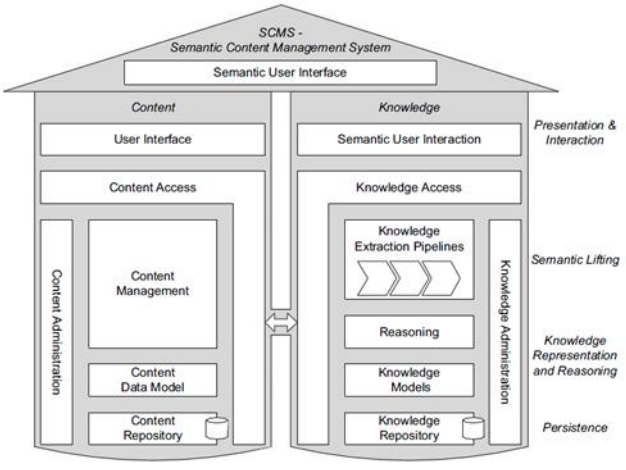


Figura 2.9: Arquitectura de referencia para un SCMS.
Tomada de [7].

Esta arquitectura está diseñada de tal manera que cualquier CMS que contenga la misma estructura mostrada en la Figura 2.1, pueda extenderse para convertirse en un SCMS. Para establecer un SCMS partiendo de un CMS se debe agregar una segunda columna de conocimientos para las características semánticas en paralelo. La interacción entre estas dos columnas se realiza si están conectadas a nivel de *Content Access* y *Knowledge Access* [7]. En un SCMS el contenido es almacenado en la columna de contenido, el cual se envía a la columna de conocimiento por medio de la capa *Content/Knowledge Access*. Este contenido puede analizarse con los elementos de la columna de conocimiento y a su vez se almacena este conocimiento en la misma columna [7].

Teniendo en cuenta en esta segunda propuesta la arquitectura de referencia de un SCMS mostrada en la Figura 2.8, la descripción de las cuatro capas se encuentran relacionadas de la siguiente manera:*i) Presentation Interaccion:* En un CMS, el usuario tiene la capacidad de modificar y consultar contenido por medio de la interfaz de usuario. Pero en el caso del conocimiento en un SCMS se requiere una capa adicional en el nivel de interfaz de usuario llamada *Semantic User Interaction* que conceda la interacción entre el usuario y el contenido. *ii) Semantic Lifting:* Un CMS no tiene la capacidad de extraer conocimiento de los contenidos almacenados en cuento a metadatos semánticos se refiere. Por tanto, un SCMS determina una capa llamada *Knowledge Extraction Pipelines* la cual contiene los algoritmos para la extracción de los metadatos semánticos. *iii) Knowledge Representation Reasoning:* Luego de construir el contenido a un nivel semántico, la información extraída puede emplearse como recursos en las técnicas de razonamiento utilizadas en la capa *Reasoning*. Para la gestión del conocimiento en el sistema se emplea *Knowledge Models* que determina la semántica de metadatos usada para expresar el conocimiento. La columna de contenido proporciona una capa adicional *Content Administration* esta es necesaría para la construcción de la administración de conocimiento. *Knowledge Administration* va desde la gestión de las plantillas de *Semantic User Interaction*, sobre *Knowledge Extraction Pipeline* y la gestión de *Reasoning* a la administración de *Knowledge Models* y *Repositories*. *iv) Persistence:* El conocimiento es almacenado en *Knowledge Repository* el cual define la estructura de datos para el conocimiento [7].

Por otra parte, en lo que respecta a la obtención de los metadatos, las anotaciones semánticas son una pieza clave para la estructuración de los recursos y posibilitar su relación con otros; ya que las anotaciones facilitan el procesamiento automático de los recursos en la web de datos y su relación con LOD, dando lugar al enriquecimiento de la información. Dentro de este proceso de anotación podemos encontrar varias opciones que van desde los gestores de contenido como es el caso que consiste en integrar, por ejemplo, la información del sitio Web con una wiki semántica. Las wikis permiten crear y mantener diversos contenidos entre diversos usuarios, principalmente

contenidos textuales, con el fin de que todos aporten información que pueda ser útil. Cada usuario proporciona contenido a la wiki de manera fácil, en el sentido de que, el único requisito para hacerlo es conocer el marcado wiki (i.e., hacer anotaciones sobre el contenido publicado). Una aplicación bajo este enfoque es *Semantic Media Wiki* que permite realizar anotaciones sobre el contenido publicado para luego enriquecerlo con información proveniente de la Web de Datos, a través de *Linked Open Data*. La forma en que se anotan los contenidos en *Semantic Media Wiki* se realiza por medio de plantillas y formularios que restringen al usuario a un conjunto de anotaciones predefinido que permiten y garantizan una estructuración de la información; además se cuenta con la posibilidad de agregar otras anotaciones abiertas que den flexibilidad y escalabilidad a los datos, favoreciendo que se puedan relacionar nuevos conjuntos de datos y por ende se incremente el enriquecimiento [13].

Para aquellos CMS que carecen de mecanismos de anotación, en *Rhizomer*[11], se presenta un enfoque que integra el contenido del CMS con un repositiorio de metadatos, a modo de brindar significado semántico a los contenidos. La Figura 2.10 muestra la arquitectura de *Rhizomer*. Este enfoque también lo implementan otras plataformas como *Drupal RDF Modules* o *Wordpress RDF*. Como se mencionó *Rhizomer* es una plataforma que combina un CMS multimedia con un depósito de metadatos a modo de estructurar la información y brindar significado semántico a los recursos; con las características de integrar de forma transparente tanto el componente CMS, como el repositorio semántico para los usuarios finales. Junto con esto, el componente CMS que implementa *Rhizomer* integra los principios de *Linked Open Data* para gestionar recursos y metadatos semánticos relacionados a dichos recursos; de ahí que, es posible clasificar a *Rhizomer* como un *Semantic Content Management System* (SCMS). Cada vez que un nuevo contenido se almacena en el SCMS, un plugin de extracción de metadatos se activa para extraer automáticamente dichos metadatos y almacenarlos en forma semántica a modo de anotaciones, con la posibilidad de ser editados para potenciar los resultados obtenidos por los plugin de extracción.

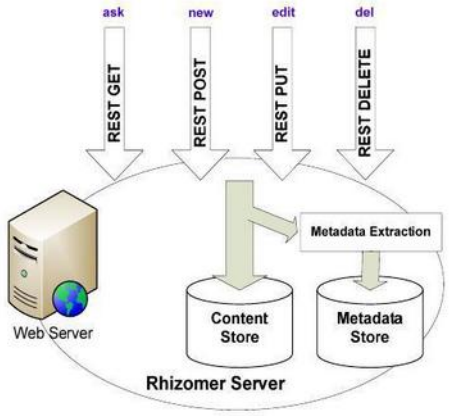


Figura 2.10: Arquitectura General de OntoWiki-CMS.

Tomada de [11].

Otros dos enfoques que se aproximan más a la integración de LOD a un CMS se presentan en [12] y [7]. Por una parte en [7] se describe un SCMS que integra conceptos de Extracción de Información (*Information Extraction* - IE) [7] y de Recuperación de Información (*Information Recuperation* - IR) [7] con tecnologías de la Web Semántica. Por otra parte, en [12] se propone *OntoWiki-CMS*, un SCMS para la generación de contenidos Web semánticamente enriquecidos. *OntoWiki-CMS* integra componentes como *OntoWiki* (i.e., una wiki semántica) y *Erfurt Framework* (como *backend* de RDF), para la colaboración y autoría de contenido Web enriquecido semánticamente y relacionado con datos provenientes de *Linked Data*, vocabularios (e.g., FOAF, DOAP) y taxonomías (e.g., SKOS); también se compone de *Site Extension*, una plantilla para representar el contenido enriquecido como un sitio Web. La Figura 2.11 muestra la arquitectura general de *OntoWiki-CMS*. Dentro de esto, es relevante el mencionar los diferentes lenguajes desarrollados para las tecnologías de la Web Semántica y puestos en práctica en estos trabajos, ya que permiten representar información y conocimiento, tales como, *Resource Description Framework*[22] (RDF), *RDF Schema*[23] (RDFS), *Web Ontology Language*[24] (OWL) y SPARQL. Siendo los lenguajes de la Web de datos, permitiendo el enriquecimiento por medio de *Linked Open Data*, al estandarizar la información de los CMS con la información presente en la Web, dando lugar al enlazamiento de anotaciones y contenidos, a conjuntos de datos por medio de consultas dinámicas, llamadas consultas *inline*, generando relaciones y ontologías que permiten un crecimiento continuo, reutilizando los datos, evitando su redundancia y actualizando en todo momento los resultados del enriquecimiento con LOD.

[22] http://www.w3c.org/RDF

[23] http://www.w3c.org/TR/rdf-schema

[24] http://www.w3c.org/TR/owl-features

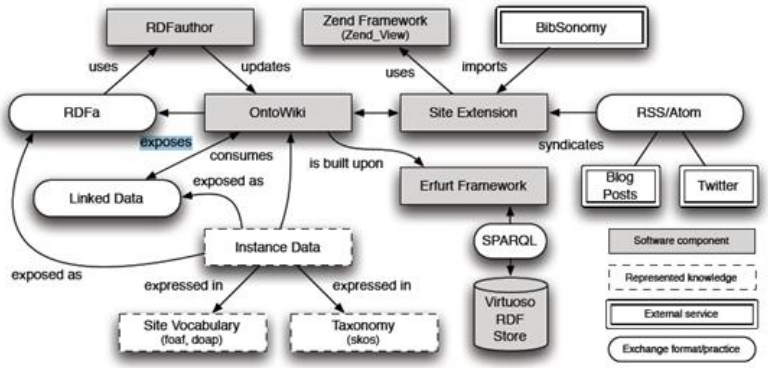


Figura 2.11: Arquitectura General de OntoWiki-CMS.

Tomada de [12].

2.2.2 Extensiones de LMS

Otro caso relacionado con el enriquecimiento de un LMS, se presenta con el gestor de contenidos de aprendizaje Moodle en [15], el cual permite que los recursos que publica un profesor únicamente se encuentren disponibles para los estudiantes inscritos al curso, esto limitando la propagación, enriquecimiento y enlace de los recursos, y conocimiento almacenado. Sumado a esto, las características que poseen los contenidos no cuentan con suficientes metadatos para llevar a cabo una búsqueda tradicional por las limitaciones impuestas por la plataforma. Por lo que el trabajo presentado en [15], propone una extensión de Moddle para que genere búsquedas de contenidos semánticamente, por medio de una representación estructural jerárquica en la que están organizados los recursos y los cursos mediante una ontología; llevando a cabo el proceso que se presenta a continuación:

1. La búsqueda se generará a partir de un conjunto de palabras clave, como el caso presentado en Ontowiki-CMS con las anotaciones.

2. Identificar el contexto o concepto que las relaciona semánticamente.

3. Finalmente, la recuperación de los recursos digitales asociados a ese contexto independientemente de la información sintáctica (metadatos) que inherentemente contiene el recurso.

La Figura 2.12 muestra la arquitectura propuesta para la extensión de Moddle en busca de generar el enriquecimiento de sus contenidos.

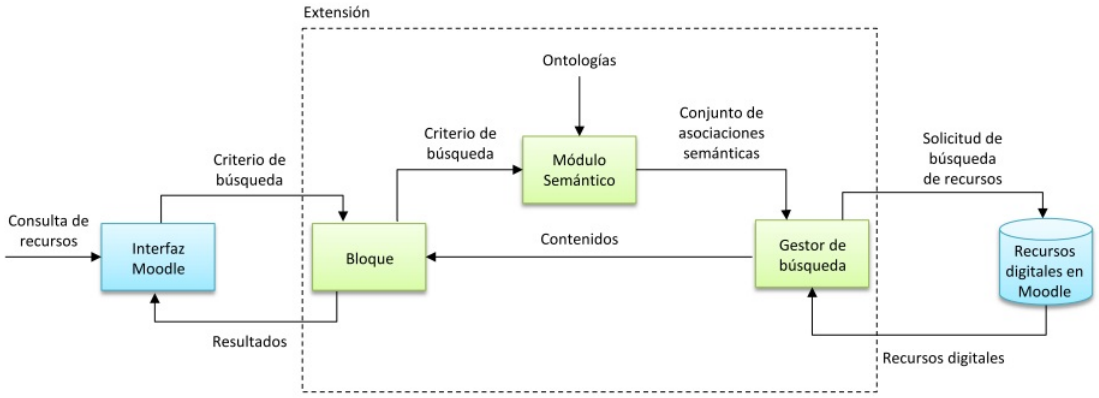


Figura 2.12: Arquitectura propuesta para diseño de la extensión de búsqueda semántica.

Tomada de [15].

La propuesta, plantea una arquitectura modular, iniciado con el módulo Interfaz Moodle, el cual tiene por función el ingreso de las consultas de recursos y la disponibilidad con la que se pueda ofrecer la herramienta, así como la visibilidad que tenga en el LMS Moodle y los resultados. Un módulo bloque [16], es un elemento que ofrece Moodle para visualizar información en el espacio de trabajo del usuario sin afectar su interacción, con el beneficio de ser configurable, este permitirá al estudiante o profesor ingresar el criterio de búsqueda y servirá de contenedor para presentar los resultados. Un módulo semántico, para implantar un mecanismo que establezca el contexto o la relación semántica que guardan entre si los conceptos proporcionados en el criterio de búsqueda, los cuales se utilizan en la localización de los recursos almacenados en Moodle, obteniendo un dominio representado por cada curso y un conjunto de conceptos relacionados. Un módulo gestor de búsqueda, que encapsula la lógica necesaria para hacer las gestiones ante la base de datos inherente en Moodle, cuyo propósito es interpretar el conjunto de cursos obtenido tras el proceso ontológico y gestionar los recursos digitales asociados para posteriormente presentarlos de forma apropiada al estudiante o profesor [15].

Los resultados de esta extensión, se ven reflejados en un prototipo que hizo uso de las tecnologías y herramientas asociadas a la construcción de aplicaciones para la Web Semántica [1] y de las recomendaciones proporcionadas por Moodle para implementar sus componentes [17]. En la Figura 2.13 se presenta un caso de estudio del prototipo de extensión, donde el estudiante proporciona el concepto de Web 2.0. La búsqueda inicia obteniendo el conjunto de cursos con los cuales se relaciona el concepto, en este ejemplo, se establece una relación indirecta con los conceptos XML, HTML y diseño

de arquitecturas semánticas, por tanto devuelve como resultados los recursos asociados a estos cursos y subcursos. Posteriormente el usuario puede identificar su procedencia y conocer información relacionada referente al recurso [15].

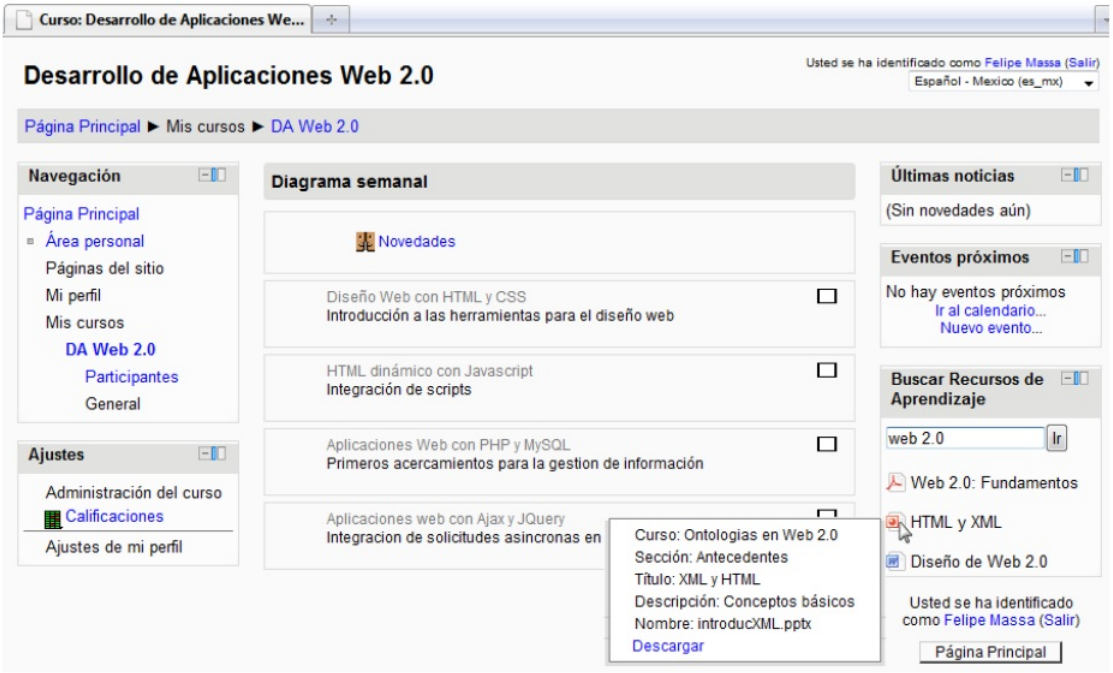


Figura 2.13: Prototipo de extensión de búsqueda de recursos de aprendizaje. Tomada de [15].

En lo referente a la presentación de la información, se presentan dos posibilidades en los trabajos Semantic Media Wiki y Rhizomer. En el primero de ellos, Semantic Media Wiki y en relación a las consultas inline se propone un proceso de refinamiento de las consultas inline bajo un interfaz gráfica, con el fin de no sobrecargar al usuario con demasiada información, tal como se muestra en la Figura 2.14. El proceso se divide en tres pasos:

1. Ingresar palabras claves o anotaciones.
2. Seleccionar la interpretación o conceptos relacionados de interés, según las palabras claves ingresadas.
3. visualizar los resultados y navegar por los diferentes conceptos hasta llegar a la información deseada.

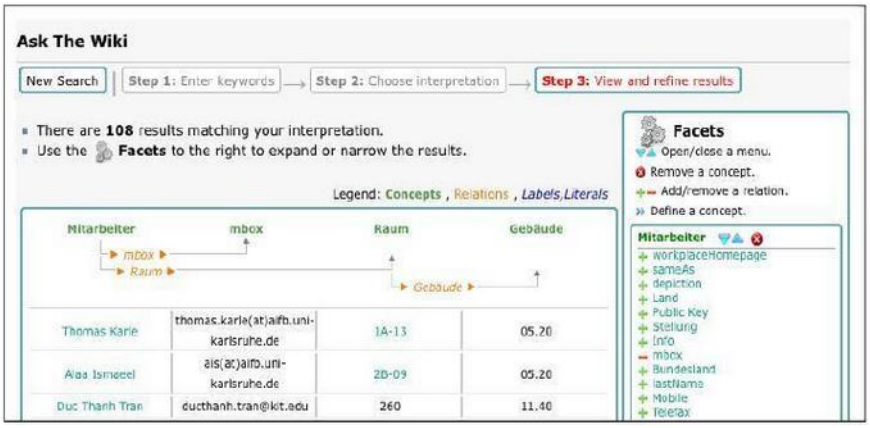


Figura 2.14: Ejemplo de consulta inline e interfaz.

Tomada de [13].

Por otro lado, en lo que respecta la plataforma *Rhizomer*, esta emplea un algoritmo para la renderización de la información en RDF a un formato más sencillo de entender por el usuario final como lo es HTML. Junto con esto es importante tener en cuenta el proceso interno que realiza el algoritmo previamente a la interpretación que consiste en descomponer las ontologías o los grafos RDF en fragmentos, con el fin de no sobrecargar al usuario final con una cantidad enorme de información como lo hace *Semantic Media Wiki*; estos fragmentos de grafo son conocidos dentro de la plataforma como pasos de navegación y determinan si el usuario así lo desea acceder a más información asociada. En la Figura 2.15 se presenta un ejemplo que permite evidenciar los pasos de navegación y cómo a partir de un metadato se puede formar otro fragmento de grafo para ser presentado al usuario.

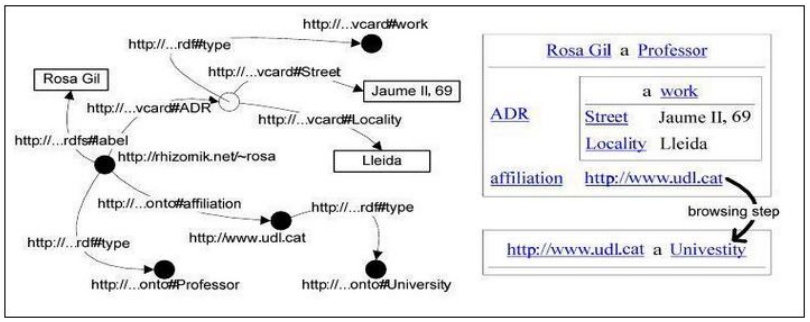


Figura 2.15: Ejemplo de fragmentación de grafo RDF y su posterior interpretación en HTML.

Tomada de [11].

En relación a las aproximaciones mencionadas para enriquecer semánticamente la información de un sitio Web. Se puede concluir que un cambio arquitectónico es necesario para permitir el almacenamiento y gestión de contenidos semánticos asociados a los recursos; dentro de esto se cuenta con la posibilidad de utilizar las arquitecturas de extensión de CMS tradicionales o hacer uso de herramientas existentes como son las wikis. Sin embargo, en estas últimas se hace necesario conocer con profundidad el marcado wiki para la generación de nuevos contenidos y procesamiento de anotaciones, como herramientas fundamentales para el estructuramiento de información y enlace con la Web de datos, empleando lenguajes como RDF, RDFS, OWL, SPARQL para el almacenamiento y consulta. Siendo más complejo el mantenimiento de las aplicaciones y de la información que gestionan, restringiendo esta tarea a personas especializadas en este tipo de marcado wiki.

En lo que respecta a la visualización de los resultados se cuenta con dos opciones, lo que son los pasos de navegación y los refinadores de consulta o navegadores; siendo estos últimos más sencillos de elaborar por motivos de interfaz gráfica de usuario y manejo de la información, debido a que existe un único espacio donde se evidencian los resultados de integración y enriquecimiento con LOD y no dentro del contenido del sitio web como lo serian los pasos de navegación, generando no solo consultas dinámicas, sino también interfaces con contenidos dinámicos. En la Figura 2.16 se presenta un mapa conceptual a modo de resumen de las propuestas de enriquecimiento.

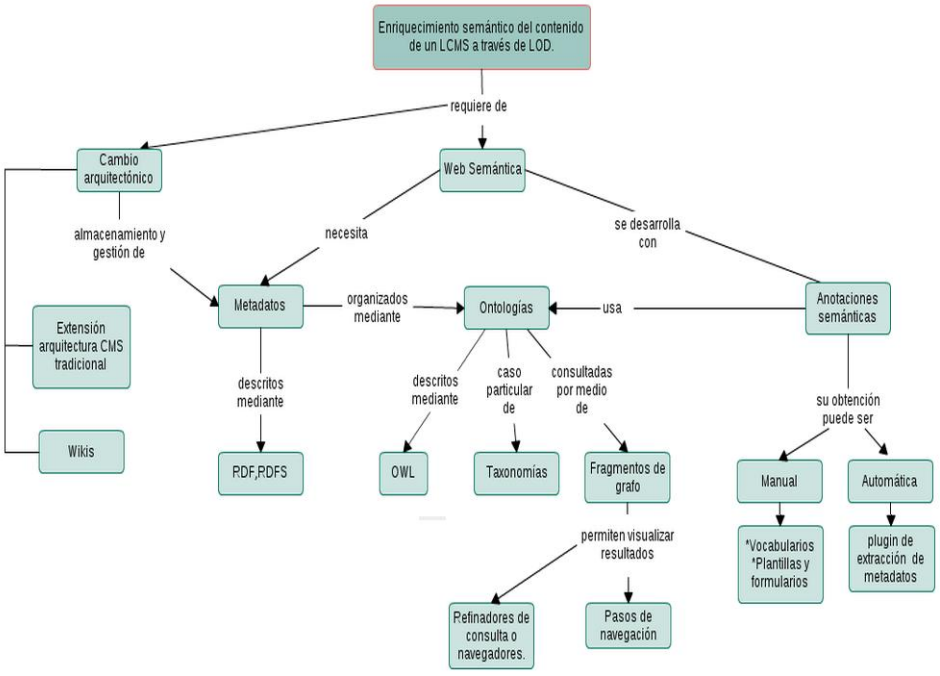


Figura 2.16: Propuestas de enriquecimiento

2.2.3 Implementaciones que hacen uso de Extracción de Información

Por otra parte, en la Extracción de Información existen diversas implementaciones de etiquetadores y reconocedores de entidades, la más común de ellas es *FreeLing*[25] que ofrece una amplia gama de operaciones de Extracción de Información para una extensa lista de lenguajes naturales. *Freeling* está desarrollado en C++, pero también posee una API para posteriores desarrollos en JAVA y una interfaz básica por línea de comandos llamada *Analyzer*, que sirve para probar las funciones de *Freeling* sin necesidad de desarrollar un proyecto sobre el mismo. Su mayor desventaja es la instalación, que es muy sencilla para sistemas Windows pero inestable en Linux en cuanto a insytalación y ejecución, lo que es un grave inconveniente teniendo en cuenta que los servidores en los que se suele hospedar Moodle (y en general los servidores de internet) usan Linux.

Una de las aplicaciones desarrolladas a partir de *Freeling* es Apache Stanbol[26], un potente framework que integra el reconocimiento de entidades nombradas (personas, organizaciones y lugares específicamente) con la Web de Datos, permitiendo visualizar las entidades detectadas en un texto, localizarlas en un mapa si se trata de lugares, mostrar su foto si son personas o su logotipo si son organizaciones. El objetivo es integrar los LMS con la Web de Datos, enlazando las entidades presentes en el contenido escrito de los LMS con información disponible en la Web de Datos.

Apache Stanbol, utiliza además de las aplicaciones de detección de lenguaje de *Freeling*, una copia parcial del contenido de DBpedia[27] para garantizar la consistencia de las búsquedas sin importar que DBpedia se encuentre en mantenimiento y utiliza a su vez un framework para el reconocimiento de entidades llamado Apache OpenNLP[28].

El reconocimiento de entidades requiere de un corpus de palabras en cada lenguaje con alrededor de quinientos millones de palabras, sobre el cual es preciso realizar una búsqueda por cada palabra del texto analizado. Esto es computacionalmente costoso e ineficiente para textos extensos y más aún si se pretende analizar diversos lenguajes al tiempo. Una alternativa para realizar este proceso con mayor agilidad es aprovechar la capacidad de reconocimiento de patrones de las Redes Neuronales[29]. Apache OpenNLP es un framework de procesamiento de lenguaje natural escrito en JAVA, que utiliza este concepto al incluir Perceptrones[30] que pueden ser entrenados con archivos de texto plano para generar un reconocedor de entidades que puede usarse posteriormente.

[25] http://nlp.lsi.upc.edu/freeling/

[26] https://stanbol.apache.org/

[27] http://dbpedia.org/About

[28] https://opennlp.apache.org/

[29] http://www.redes-neuronales.com.es/tutorial-redes-neuronales/tutorial-redes.htm

[30] http://www.redes-neuronales.com.es/tutorial-redes-neuronales/el-perceptron-simple.htm

Apache OpenNLP viene por defecto sin reconocedores de entidades y estos pueden ser descargados de su sitio web http://opennlp.sourceforge.net/models-1.5/ para diversos lenguajes. Los reconocedores están entrenados para identificar nombres, lugares, organizaciones u otras entidades pero su aplicación para usos específicos es limitada. El entrenamiento de los Perceptrones se realiza dando a este un archivo de texto plano donde cada palabra o *token* (i.e., signos de puntuación o etiquetas) debe estar separado por espacios y las entidades presentes deben estar etiquetadas de la siguiente manera: <START:tipo> entidad <END>. En lugar de la palabra tipo, se escribe lo que es la entidad (e.g., persona, lugar, organización, animal, etc). Una vez se tiene el texto para el entrenamiento en el formato correcto, se realiza una llamada al método *train* propio de OpenNLP que recibe el archivo de texto que se usará para el entrenamiento, el tipo de entidades a entrenar (debido a que sólo se entrena un tipo de entidad a la vez) y el nombre del archivo .bin resultante del entrenamiento que podrá ser utilizado para reconocer entidades.

2.2.4 LMS

En la actualidad, existen un gran número de aplicaciones en el contexto de los LMS para la gestión de diversas plataformas educativas; éstos se pueden dividir en dos grupos, los LMSs propietarios Blackboard[31], Desire2Learn[32], eCollege[33], Saba Learning[34], Docebo[35], Claroline[36], entre otros y los LMSs de código abierto Moodle[37], Sakai[38], Atutor[39], Ilias[40], entre otros.

Los LMS de popietario ofrecen diversas funcionalidades útiles para el aprendizaje. Sin embargo, por ser software propietario no se puede modificar el código, los de código abierto por su parte permiten la modificación y distribución de su código para generar aplicaciones diferentes partiendo del código inicial, es más, los desarrolladores de estos animan a los usuarios a generar plugins y módulos nuevos para compartirlos en próximas entregas de software y así mejorar su LMS. Para explicar la estructura de un LMS de código abierto, citaremos como ejemplo a la plataforma Moodle (*Modular Object Oriented Dynamic Learning Environment*) explicando la arquitectura, las

[31] www.blackboard.com/

[32] http://www.d2l.com/

[33] http://www.ecollege.com/index.php

[34] https://www.saba.com/us/apps/learning-work/

[35] https://www.docebo.com/

[36] http://www.claroline.net/

[37] https://moodle.org/

[38] https://sakaiproject.org/

[39] http://www.atutor.ca/

[40] http://www.ilias.de/

herramientas con las que cuenta la plataforma, la estructura de la base de datos y el estándar de programación.

1. Arquitectura de Moodle

   Esta plataforma es de distribución libre y se encuentra diseñada de forma modular permitiendo que sus módulos sean independientes, configurables, además de poder ser habilitados o deshabilitados según sea conveniente; así mismo brinda una mayor flexibilidad para la modificación de sus funcionalidades (administración de cursos, administración de usuarios, administración de archivos, chat, foros, entre otros) [8].

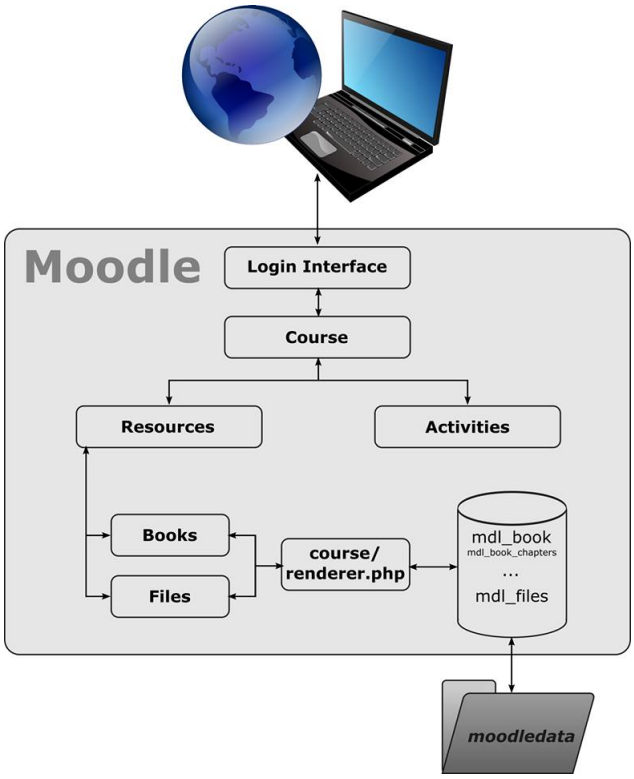


Figura 2.17: Arquitectura de Moodle.

La Figura 2.17 muestra la arquitectura de Moodle, la cual se encuentra divida en dos tipos de herramientas los Recursos y las Actividades, que serán explicadas más adelante. Nos centraremos en los Recursos ya que estos son lo que serán usados como insumo para el enriquecimiento semántico del contenido. Entre los

recursos existentes se encuentran: archivo, carpeta, etiqueta, libro, página, paquete de contenido IMS, URL.

De estos recursos se usará Libro, que se encuentra almacenado en la base de datos de Moodle y Archivo, que se encuentra guardado en un directorio llamado ”moodledata”.

El directorio ”moodledata” es donde se guardan los archivos con extensión .pdf, .doc, .docx, .png, .rar, entre otros que son publicados por medio de la interfaz de Moodle. Este directorio es creado una sola vez en el momento de la instalación indicando un nombre y una localización (por defecto ”moodledata”) de la carpeta, esta información puede ser consultada en el archivo config.php.

En la carpeta ”moodledata” hay otras carpetas llamadas *cache, filedir, lang, temp, trashdir*[41] . En la carpeta llamada *filder* se guardan los archivos que son subidos a Moodle. Los archivos que son almacenados en este directorio se guardan en la Base de Datos de Moodle de allí la relación existente entre el directorio y la Base de Datos.

La Tabla 2.2 muestra los requisitos mínimos para llevar a cabo la instalación de Moodle. Este se encuentra escrito en PHP, y la versión actual (versión 2.8) requiere la versión de PHP 5.4.4 o superior para funcionar[42].

Navegadores	Google Chrome (30.0), Mozilla Firefox (25.0), Safari 6, Internet Explorer 9.
Moodle	Actualización de Moodle (2.x)
Base de Datos	PostgreSQL (9.1), MySQL (5.5.31), MariaDB (5.5.31), MSSQL (2008) u Oracle (10.2)
PHP	PHP 5.4.4

Tabla 2.2: Requisitos mínimos para instalación de Moodle.

Tomado de https://docs.moodle.org/all/es/Notas_de_Moodle_2.8#Requisitos

[41] https://docs.moodle.org/all/es/Directorio_Moodledata

[42] https://docs.moodle.org/all/es/Notas_de_Moodle_2.8#Requisito

2. Herramientas de Moodle

Moodle cuenta con dos tipos de herramientas para el aprendizaje, como son los Recursos y las Actividades.

(a) Actividades: *Una actividad es un nombre general para un grupo de características en un curso Moodle*[43]. En estas actividades se refleja la interacción entre los estudiantes y los profesores de los cursos, con el fin de obtener calificaciones y de generar debates o situaciones de intercambio de conocimiento. Moodle tiene 14 tipos de actividades para su versión 2.x, las cuales se muestran en la Tabla 2.3.

ACTIVIDADES	
Base de Datos	Herramienta externa
Chat	Lección
Consulta	Retroalimentación
Cuestionarios	SCORM
Encuesta predefinida	Taller
Foro	Tareas
Glosario	Wiki

Tabla 2.3: Actividades de Moodle.
Tomado de https://docs.moodle.org/all/es/Actividades

(b) Recursos:*Un recurso es un objeto que un profesor puede usar para asistir el aprendizaje, como un archivo o un enlace*[44].

Moodle cuenta con 7 tipos de recursos para su versión 2.x, los cuales se muestran en la Tabla 2.4.

[43] https://docs.moodle.org/all/es/Actividades

[44]https://docs.moodle.org/all/es/Recursos

RECURSOS
Archivo
Carpeta
Etiqueta
Libro
Página
Paquete de contenido IMS
URL

Tabla 2.4: Recursos de Moodle.
Tomado de https://docs.moodle.org/all/es/Recursos

Estos recursos son modificados y actualizados por los docentes encargados de los cursos, por ende, se hará uso de dos de estos recursos como insumo para el enriquecimiento semántico del contenido, estos son: ”Recurso archivo”, el cual permite subir y mostrar una variedad de recursos en un curso; de los distintos formatos que tiene este recurso, los formatos seleccionados comprenderán archivos planos como lo son los documentos PDF, las hojas de cálculo, documentos en formatos .doc, .docx, .txt, .odt. y ”Recurso Libro” el cual es un recurso multi-página que es construido por el docente del curso.

3. Modelo de Base de Datos de Moodle

En la actualidad, la base de datos de Moodle se encuentra conformada por aproximadamente doscientas cincuenta tablas (para la versión 2.8); las cuales son creadas durante el proceso de instalación de Moodle; el nombre para estas tablas se crea por defecto anteponiéndole el prefijo *mdl_* seguido de su módulo correspondiente.

Cada módulo que contiene Moodle posee una o más tablas en su base de datos, por ende cada tabla tiene la siguiente estructura para su nombre: *mdl_nombreModulo* (si es la tabla para el módulo) o *mdl_nombreMódulo_nombreSección* (si es una tabla que hace parte de un módulo); por ejemplo, para el módulo book se encuentran las siguientes tablas relacionadas *mdl_book* y *mdl_book_chapters* [10].

Para modificar la base de datos de Moodle basta con usar el lenguaje de consulta SQL, usar una herramienta como por ejemplo PhpMyAdmin y seguir la estructura de los nombres para las tablas que se requieran crear.

4. Código de Moodle

Moodle se divide por módulos, donde la vista, las clases y la base de datos se comunican de forma indirecta en la mayoría de los casos.

Cuando es necesario acceder a la base de datos, se llama a la variable global *$DB* que es una instancia de la clase *moodle_database*. En ella se definen métodos para las consultas de manera que, para la mayoría de los casos, no es necesario escribir consultas SQL en el código de Moodle, salvo que se requiera una consulta con requerimientos muy particulares.

Las clases que modelan los objetos que contienen la información de los módulos contienen métodos para la gestión de dicha información: creación, edición y eliminación, así como métodos propios del módulo y algunos métodos orientados a la visualización.

Para comunicar las clases de los módulos con el curso para su visualización, se utiliza la clase *cm_info* en las que se mapea la información del módulo relevante a visualización y usando instancias de esta clase, el *renderer* del curso puede mostrar la información. De igual forma, la clase *cm_info* contiene llamados a métodos que retornan punteros a la clase del módulo original para usar métodos específicos del módulo que no son mapeados en el *cm_info*.

En resumen, la información de los módulos se almacena en la base de datos, luego, se llama a la base de datos en PHP por medio de la instancia de la clase de base de datos almacenada en la variable global *$DB* y con un método *get_records()* obtiene la información requerida del módulo y la almacena en la clase de dicho módulo. Esta genera su propia instancia de *cm_info* en el que se almacenan los datos que son necesarios para ser visualizados por el curso. El curso llama a todos los módulos que comparten su id y por cada uno de ellos requiere una instancia de *cm_info*. Con cada instancia, el *renderer* del curso genera las secciones del curso por semanas y visualiza cada módulo.

2.3 Análisis del Marco Referencial

La mayoría de los datos publicados en la Web, no tienen significado si se consideran individualmente, pero se vuelven útiles en el momento en que se enlazan a otros datos. En el caso de los datos que sí poseen significado a nivel individual, pueden enriquecerse y volverse más significativos una vez se relacionan a más datos. Este es el principio de *Linked Data*, ya que permite crear enlaces entre datos de diferentes fuentes, las cuales pueden provenir de diversas organizaciones con ubicaciones geográficas diferentes, para poder compartir datos estructurados (i.e., datos en RDF) y distribuidos en la Web, de tal forma que, el valor en esos datos aumente a medida que se relacionan con otros.

De ahí que, importantes organizaciones como por ejemplo, la *BBC*[45], *Thomson Reuters*[46] y la *Library of Congress*[47], entre otras, se han sumado a la idea de publicar datos abiertos en la Web siguiendo las buenas prácticas de Datos Abiertos Enlazados *Linked Open Data* (LOD), permitiendo aportar a la construcción de un espacio global de datos sobre personas, compañías, libros, publicaciones científicas, películas, música, programas de radio y televisión, genes, proteínas, fármacos y ensayos clínicos, comunidades en línea, datos estadísticos y científicos, que en el año de 2014, según la *W3C*[48], se estimaban en 1014 conjuntos de datos con más de 32.000 millones de tripletas en *Resource Description Framework* [49] (RDF) con 570 millones de enlaces entre ellas [50].

Por lo anterior, es importante investigar cómo enriquecer semánticamente los contenidos publicados en un *Learning Management Systems* para consumir los datos abiertos disponibles en la Web de Datos aplicando los principios *Linked Open Data*. Para alcanzar dicho objetivo se requiere hacer uso de las técnicas de Extracción de Información, el cual es un proceso vital y complejo en el que se requiere adaptar sistemas existentes de Extracción de Información en español para que funcionen de la mejor manera posible sobre los diferentes tipos de documentos de texto que puedan subir los profesores al LMS y adaptar su funcionamiento de manera que sea transparente al usuario del nuevo sistema gestor.

La implementación de Extracción de Información varía según sea el propósito del análisis del texto en Lenguaje Natural. Para nuestro caso, la técnica principal de extracción de información a ser utilizada es el Reconocimiento de Entidades Nombradas o NER. Por desgracia, las implementaciones existentes de esta técnica se enfocan en detectar nombres propios, lugares, organizaciones y valores tales como cifras numéricas,

[45] http://www.bbc.co.uk/nature/feedsanddata

[46] https://customers.reuters.com/rmds/CZRDP/ECOPages/DataSources.aspx

[47] http://id.loc.gov/authorities/subjects/sh2003010124.html

[48] http://lod-cloud.net/state/

[49] http://www.w3c.org/RDF

[50] http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/

porcentajes y fechas entre otros y no temas académicos que serían las palabras claves en los textos subidos a Moodle, que son objeto de análisis.

Al igual que la Extracción de Información, la implementación de NER es variable y en nuestro caso requerimos de una implementación flexible capaz de reconocer los patrones que siguen las palabras clave que buscamos y no que se limiten a comparar con una lista existente; es decir, una implementación que detecte todas las entidades que parezcan ser palabras clave en lugar que las busque de entre una lista estática. La necesidad de flexibilidad en la detección de entidades, responde a que las palabras clave en los textos que van a ser analizados son en su mayoría temas académicos y éstos cambian con el tiempo y se generan nuevos constantemente. Por lo anterior se decidió implementar métodos de Extracción de Información basados en una extensión de NER capaz de detectar temas académicos.

El resultado de este trabajo de investigación es una aplicación a modo de prototipo que permite la consulta de material adicional al que se encuentra publicado en un LMS, dicho material se obtiene de una consulta realizada a la Web de Datos; para lograrlo se requiere del apoyo de un modelo arquitectónico que facilite la integración del enriquecimiento semántico del contenido. La adaptación de la arquitectura de un LMS a la de un SCMS se hará con base a la arquitectura planteada en la sección 2.2, subsección 2.2.1, Figura 2.8: Capas de referencia para un SCMS, ya que ésta propone una arquitectura conceptual por capas que proporciona mecanismos de extracción y asociación de información.

Para el desarrollo del prototipo y posteriores pruebas, se elige como LMS a usar la pataforma Moodle, ya que es un Sistema de Gestión de Cursos que brinda herramientas para la creación, administración y distribución de contenidos relacionados con la formación académica. Esta plataforma es de distribución libre y se encuentra diseñada de forma modular permitiendo la independencia y configuración de sus módulos, además de poder ser habilitados o deshabilitados según sea conveniente; brinda una mayor flexibilidad para la modificación de sus funcionalidades (administración de cursos, administración de usuarios, chat, foros, administración de archivos, entre otros).

Como insumo para el enriquecimiento semántico del contenido, se hará uso de uno de los recursos de la plataforma *Moodle* denominado ”Recurso archivo”, el cual permite subir y mostrar una variedad de recursos en un curso. De los distintos formatos que tiene este recurso, los formatos seleccionados comprenderán archivos como lo son los documentos PDF, las hojas de cálculo, documentos en formatos .doc, .docx, .txt, .odt.

La información obtenida como resultado de la consulta a la Web de Datos estará relacionada con el contenido publicado en un LMS referente al recurso nombrado anteriormente, por medio del formato RDF.

En cuanto al proceso de Extracción de Información, debido a que el propósito de Apache Stanbol no es fácilmente modificable para hacerlo coincidir con los objetivos de nuestro trabajo, y que su instalación y puesta en marcha no es consistente, decidimos no utilizarlo directamente para construir nuestro proyecto a partir de él, pero sí tomar en cuenta los mecanismos de Extracción de Información utilizados en él como lo es Apache OpenNLP.

Una vez se alcancen los objetivos propuestos, los resultados obtenidos contribuirán de manera positiva en la capacidad de proporcionar servicios para acceder, consultar, buscar y enlazar la gran cantidad de datos que habita en la Web. Esto facilitaría, en parte, los procesos tanto de aprendizaje como de enseñanza a través de un LMS.

Capítulo 3

Enriquecimiento semántico del contenido publicado en Moodle

En el presente capítulo se da a conocer el proceso realizado para el enriquecimiento semántico del contenido publicado en Moodle y se explican las diferentes tecnologías usadas para la extensión y adaptación de la plataforma Moodle.

3.1 Técnicas para extracción de Información (NER)

Los LMS son usados para gestionar cursos virtuales o como soporte en línea para cursos presenciales, la forma más común en la que se utilizan en el proceso de aprendizaje es como un repositorio de documentos compartidos por el tutor, ya sea de su autoría o de alguien más. Estos documentos quedan almacenados en la base de datos del LMS, como en el caso del Sistema Gestor de Aprendizaje objeto de estudio, Moodle.

El primer paso para la Extración de Información con técnidas de NER, fue identificar la forma en la que se almacenan los archivos en Moodle. Moodle guarda las carpetas y archivos en la carpeta *filedir* que a su vez está dentro de la carpeta *moodledata*. Además Moodle, crea hasta cuatro registros en la tabla *mdl_files* de la base de datos por cada archivo y carpeta de forma encriptada. Cada registro en la tabla está compuesto por los cammpos *id*, *pathnamehash*, *contexid*, *component*, *filearea*, *itemid*, *filepath*, *filename*, entre otros, tal como se muestra en la figura 3.1.

Por ejemplo, la figura 3.1 muestra un extracto de cuatro registros en la tabla *mdl_files* que hacen referencia a un mismo archivo. Para identificar el registro correcto se tienen en cuenta los valores de los campos *contextid*, *filearea* y *component*, los cuales deben ser el número identificador del archivo (en este caso 31), *mod_resource* y *content*, respectivamente. Una vez identificado el registro correcto, se tiene en cuenta el valor del campo *pathnamehash*, que para el ejemplo es *d427be...* Este valor contiene la ubicación del archivo en la caperta *filedir* y el nombre del archivo encriptado, es decir, que el nombre del archivo es *d427bebafb6e..*, que está en la carpeta *27*, que su vez está dentro de la carpeta *d4*.

id	pathnamehash	contextid	component	filearea	itemid	filepath	filename
28	d427bebafb6e752461710e8b845f87cee26b44a3	5	user	draft	903228759	/	clase2-WEBSUV.pptx
29	da39a3ee5e6b4b0d3255bfef95601890afd80709	5	user	draft	903228759	/	.
30	d427bebafb6e752461710e8b845f87cee26b44a3	31	mod_resource	content	0	/	clase2-WEBSUV.pptx
31	da39a3ee5e6b4b0d3255bfef95601890afd80709	31	mod_resource	content	0	/	.

Figura 3.1: Ejemplo Base de Datos de Moodle. Tabla mdl_files.

Una vez localizado el archivo y conociendo la extensión que poseía originalmente el segundo paso es extraer su texto, para lo cual se usaron las librerías de Java Apache POI[1], Apache PDFBox[2] y Apache ODF Toolkit[3] debido a que, a pesar de carecer el archivo de extensión aún son reconocidos, así como se manejan apropiadamente los saltos de líneas, separación con espacios y tíldes.

Con el texto extraído, el tercer paso es obtener las palabras claves, para lo que se necesita escoger una Técnica de Extracción de Información en lenguaje español especializada en temas relacionados con Ciencias de la Computación. Inicialmente se estudió la posibilidad de adaptar un *POSTagger* para que detectara las palabras clave, pero su complejidad computacional y la necesidad de construir un corpus abundante para su funcionamiento terminaron por descartarlo. La siguiente técnica en considerarse fue NER, sus resultados y la diversidad de formas en las que puede implementarse resultan afines a las necesidades de la tarea en cuestión, sin embargo, las aplicaciones existentes de NER se limitan a Nombres Propios, Lugares, Organizaciones y valores como cifras numéricas, fechas y porcentajes. Por lo que se decidió implementar una

[1] http://poi.apache.org/download.html

[2] https://pdfbox.apache.org/download.cgi

[3] https://incubator.apache.org/odftoolkit/

nueva aplicación de NER para detectar temas académicos, el común denominador de las palabras clave relevantes en los textos presentes en Moodle.

Se estudió *Freeling* [4] como primera alternativa para abordar la necesidad de procesamiento de lenguaje natural, más precisamente NER. Al poseer una herramienta de demostración pre-construida llamada *Analyzer*, fue posible evidenciar el funcionamiento del framework en el dominio de los nombres, organizaciones, lugares y demás previamente mencionados. Compilar y utilizar las librerías del mismo para crear una nueva aplicación no resultó ser sencillo y el proceso de instalación difícilmente reproducible; un cambio en la herramienta usada por los desarrolladores para ofrecer el soporte y las descargas de las librerías dejó temporalmente inaccesibles los archivos binarios necesarios para la instalación, por lo que exploramos otras opciones.

Se investigó Apache STANBOL[5], un framework en desarrollo con problemas de estabilidad similares a los de *Freeling* pues no todas sus partes son compatibles entre sí ya que el desarrollo sobre cada una es paralelo y por lo tanto su integración presentaba llamados a módulos inexistentes en el momento de la instalación. Una vez instalados los módulos de las versiones compatibles entre sí, se pudo probar su funcionamiento, la capacidad multilenguaje y velocidad del NER respecto la longitud del texto, la funcionalidad sin conexión de las consultas a DBpedia y la calidad de los resultados arrojados. El código como tal, debido a su extensión y las complicaciones de su compilación debido a las inconsistencias anteriormente mencionadas, no resultó de tanta utilidad como sí lo hicieron sus listas de librerías entre las que se encuentra Apache OpenNLP [6], por defecto, la librería no incluye modelos por lo que no puede realizar procesamiento natural hasta no descargarlos de la página oficial, estos modelos no se ajustan al propósito del presente trabajo, ya que son el resultado del entrenamiento de un perceptrón para reconocer entidades en el que se usó un conjunto de textos extraídos de noticias de una cadena española, por lo que incluso en el ya limitado dominio de NER, era insuficiente al carecer de entidades como Francia, Italia, Estados Unidos y otros.

En nuestro caso requerimos que el sistema reconozca tópicos académicos presentes en textos de la misma naturaleza. Afortunadamente, Apache OpenNLP posee métodos para generar modelos que reciben un texto. Dicho texto debe tener un formato específico, en el que cada *token* debe estar separado por espacios y donde las entidades deben estar etiquetadas de la forma: < START: (tipo de entidad)> entidad <END>

[4] http://nlp.lsi.upc.edu/freeling/

[5] https://stanbol.apache.org/

[6] https://opennlp.apache.org/

Primero se generó un conjunto de entidades representando tópicos de Ciencias de la Computación y se realizó un documento recopilando fragmentos de texto donde estas entidades eran mencionadas al menos una vez. Es relevante que en el texto aparezcan las entidades explícitamente porque si durante el entrenamiento no son detectadas, el perceptrón no aprende a encontrarlas. Igualmente es importante el contexto en el que estas están porque le ayudan al perceptrón a detectar entidades similares. Partiendo de esta lista de entidades que fueron seleccionadas como patrón y de los textos extraídos de sitios web en los que se mencionan las entidades, fue necesario crear una clase *Tagger* que contiene un método para convertir dichos textos en el formato de texto requerido etiquetando las entidades listadas previamente. Para realizar este proceso es necesaria una lista de palabras clave, que llamaremos *keywords*, cuyas apariciones en el texto deben ser etiquetadas y el texto de entrada que usualmente es una oración en la que una o más *keywords* aparecen.

Para explicar el funcionamiento de este método usaremos como ejemplo el texto *"Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización"*.

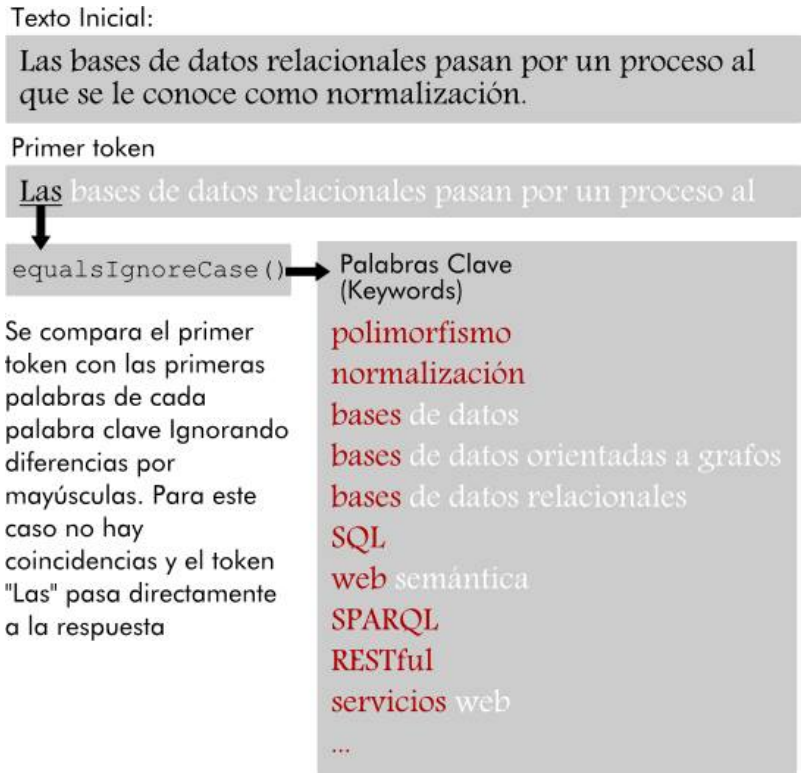


Figura 3.2: Ejemplo de Tagger.

Se inicia comparando palabra por palabra (figura 3.2), en este caso *"Las"* se compara con la primera palabra de todas las keywords ignorando mayúsculas. Como ninguna keyword inicia con *"Las"*, esta palabra pasa a ser parte de la respuesta, es decir que su aparición en el texto no es modificada. Para la siguiente palabra *"bases"* (figura 3.3) existen varias coincidencias que inician por esta palabra (bases de datos, bases de datos orientadas a grafos y bases de datos relacionales) todas estas *keywords* están compuestas por más de una palabra, así que, por ser coincidencias parciales, se añaden a la lista de *keywords* candidatas. La palabra *"bases"* aún no puede añadirse a la respuesta puesto que no sabemos aún si será etiquetada, por lo tanto la almacenamos en una respuesta acumulada mientras se decide si se etiqueta o no.

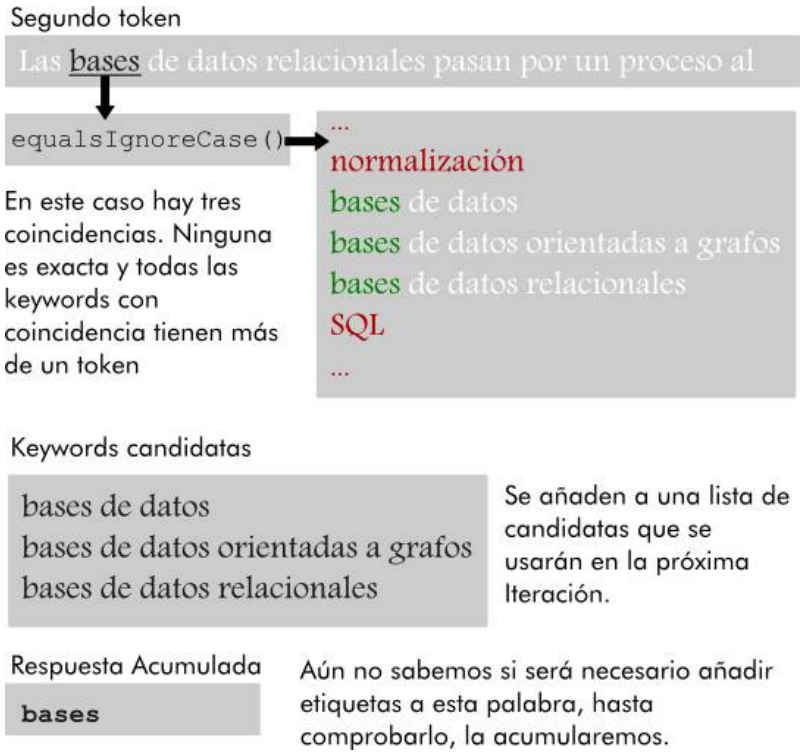


Figura 3.3: Continuación Ejemplo de Tagger.

La palabra *"de"* se compara con la segunda palabra de las candidatas, con los mismas tres coincidencias y para la palabra *"datos"*, que se compara con la tercera palabra de las candidatas, ocurre una coincidencia exacta, es decir, *"bases de datos"*. Si la siguiente palabra no tuviera coincidencias, se etiquetaría *"bases de datos"* pero, como aún no es seguro, se guarda en una variable y se continúa el proceso (figura 3.4). A demás como la *keyword "bases de datos"* no posee una cuarta palabra, se descarta como candidata.

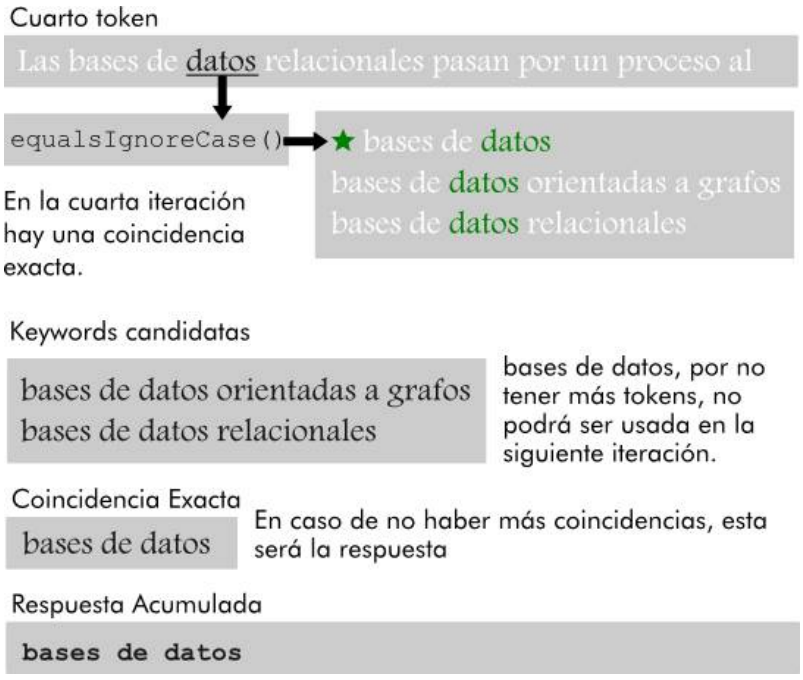


Figura 3.4: Continuación Ejemplo de Tagger.

En la siguiente palabra *"relacionales"* encontramos sólo una coincidencia y es exacta, por lo que se descarta la coincidencia exacta anterior. La lista de candidatas queda vacía y por lo tanto se etiqueta la palabra clave *"bases de datos relacionales"*. Quedando la respuesta acumulada como <START:systems> bases de datos relacionales <END>, esto se añade a la respuesta y se continúa el proceso para las demás palabras del texto (figura 3.5).

El resultado de este método, para cada una de las líneas de un texto constituye el contenido de archivo necesario para el entrenamiento del perceptrón. Por lo tanto es necesario guardarlo en un archivo con extensión "train" para proceder con el entrenamiento. Una vez entrenado el perceptrón, este genera un modelo en formato "bin" que puede ser utilizado por los métodos de OpenNLP para detectar entidades. Usando estos modelos, fue posible probar el funcionamiento de la detección de entidades para diversos datos de entrada.

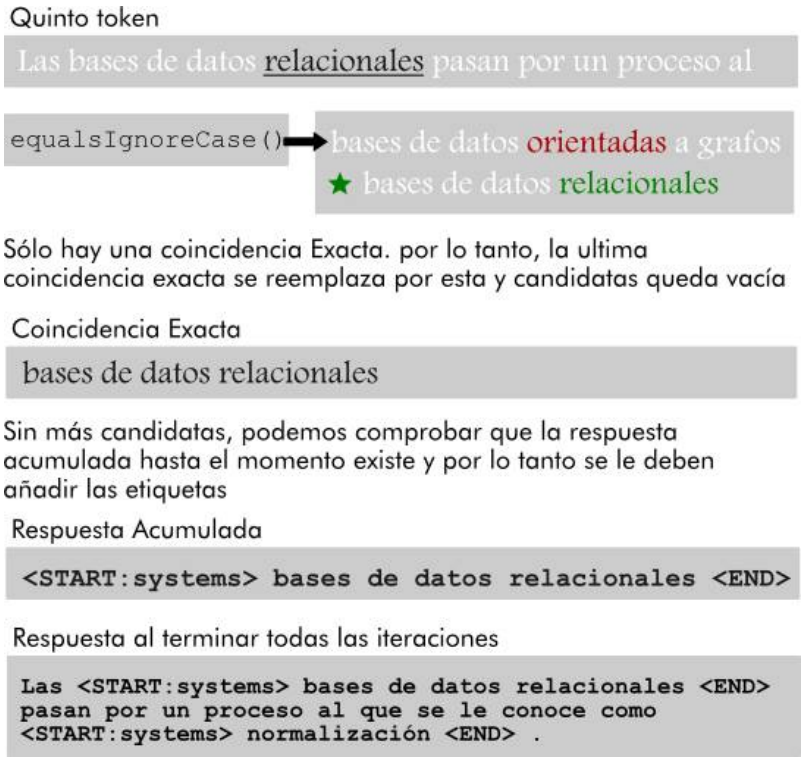


Figura 3.5: Continuación Ejemplo de Tagger.

El estudio de diversos entrenamientos con diversos conjuntos de entrada, es decir, entidades y textos nos condujo a deducir que el número de entidades debe ser mayor a veinte y menor que cien[7] para obtener los mejores resultados, ya que cuando se incluyen muy pocas entidades, el perceptrón se vuelve muy laxo para distinguir entidades similares, reconociendo casi todos los tokens como entidad, mientras que al incluir demasiadas entidades, el perceptrón se vuelve muy estricto, tomando casi exclusivamente las entidades listadas. Como los tópicos presentes en los textos académicos de un LMS pueden tratar de cualquier tema y teniendo en cuenta la restricción de cantidad de entidades anteriormente mencionada, es posible generar diferentes modelos para subtópicos académicos relacionados entre sí, (e.g., Química, Medicina, Física) y reducir el dominio de las entidades que el sistema reconoce con alta probabilidad a aquellas palabras dentro de dichos subtópicos.

En cuanto a los resultados de las palabras claves detectadas como entidades, se realizaron varios procedimientos para su posterior consulta en DBpedia como eliminar las repeticiones, ya que si una palabra clave aparece cientos de veces, esta será listada tantas veces como aparezca. También se notó que debido a que muchas palabras

[7]Estos valores se dedujeron de manera empírica por prueba y error

clave contienen preposiciones (i.e., ”programación con restricciones”) el perceptrón detecta palabras como ”problemas con” como una palabra clave por su similitud a las palabras con preposiciones. Debido a esto, se eliminaron las preposiciones finales de todas las palabras detectadas. Finalmente, para asegurar la relevancia de los resultados y evitar que las consultas sean muy numerosas, se seleccionaron del resultado las apariciones, exactas de las palabras clave que hicieron parte del proceso de entrenamiento y se pusieron al inicio de la lista de resultados; para las demás se dio una posibilidad del 50% de ser elegidas, esto con el fin de dar posibilidades a la palabras clave similares sin hacer la consulta a DBpedia demasiado copiosa.

3.2 Consultando a la Web de Datos

La Web de Datos es una colección de muchos conjuntos de datos que a su vez se encuentran conectados entre sí, en particular el conjunto de datos mas importantes es DBpedia, que también se encuentra conectado con otros conjuntos de datos, conformando alrededor de 3 mil millones de enlaces RDF, permitiendo enlazar cada vez más información y nutriendo esta gran red de datos[8]. Para el desarrollo de este proyecto se usó como fuente principal DBpedia, *ya que utiliza una gran ontología multi-dominio que se ha derivado de Wikipedia*[9].

DBpedia cuenta con un total de 125 versiones (125 idiomas diferentes) haciendo que contenga 38.3 millones de ”cosas” de las cuales 23,8 millones se encuentran relacionados con conceptos de DBpedia en Inglés. El conjunto de datos completo DBpedia cuenta con 38 millones de resúmenes en distintos idiomas, 25.2 millones de enlaces a imágenes y 29.8 millones de enlaces a páginas web externas; 80.9 millones de enlaces a las categorías de Wikipedia, y 41.2 millones de categorías YAGO [18].

En la Tabla 3.1 se muestran algunos datos estadísticos sobre el tipo de información que ofrece DBpedia como fuente de datos.

[8]http://dbpedia.org/Datasets

[9]http://dbpedia.org/Datasets

CLASES	INSTANCIAS (millones de enlaces)
Personas	1.445.000
Lugares	735.000
Obras en creación (álbumes de música, películas y videojuegos)	411.000
Organizaciones (empresas e instituciones educativas)	241.000
Especies	251.000
Enfermedades	6.000

Tabla 3.1: Estadística de la ontología DBpedia año 2014.

Tomado de http://dbpedia.org/Datasets

Esta gran cantidad de recursos hace de DBpedia la fuente de datos con mayor información y la mejor opción para realizar el enriquecimiento semántico del contenido de un LMS por medio del consumo de datos a la Web de Datos.

Para realizar una consulta a DBpedia, ésta debe de realizarse por medio de un SPARQL Endpoint, el cual es un servicio web para obtener información a través de consultas, usando el lenguaje SPARQL y especificando el formato en el que se espera ver los resultados (e.g., HTML, XML, JSON, JavaScript, Turtle, RDF/XML, N-Triples, CSV y TSV). El SPARQL Endpoint público de DBpedia es http://dbpedia.org/sparql.

La Figura 3.6 muestra una consulta en SPARQL en la que se obtienen los primeros 100 links de los recursos (páginas web o documentos) que se encuentren relacionados o que contengan el valor de la variable $term.

En la conuslta, las líneas de la cláusula WHERE contiene cuatro patrones de tripletas y dos filtros; la primer tripleta (figura 3.6, línea 1.), consta de ”sujeto - predicado - objeto” seguido de un punto para poner fin a la tripleta. En esta tripleta, el sujeto es ?sujeto, el predicado es *rdfs:label*, el objeto es ?lbl, en su forma abreviada. El objeto es lo que se trata de encontrar con esta consulta. Para explicar esta primer línea hace refencia a ”encontrar el valor del objeto de la tripleta que coincida con el patrón ?sujeto *rdfs:label* ?lbl”.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT DISTINCT  ?lbl ?topic ?abstract ?pageExternal
WHERE {
1. ?sujeto rdfs:label ?lbl .
2. ?sujeto foaf:isPrimaryTopicOf ?topic .
3. ?sujeto dbpedia-owl:abstract ?abstract.
4. ?sujeto dbpedia-owl:wikiPageExternalLink ?pageExternal.
5. FILTER ( regex(?lbl, '".$term."', 'i')).
6. FILTER (langMatches(lang(?abstract), 'en')||
   langMatches(lang(?abstract), 'es'))
} ORDER BY ?pageExternal
LIMIT 100
```

Figura 3.6: Primera versión de la consulta.

La segunda tripleta (figura 3.6, línea 2.), indica que el sujeto ?sujeto debe tener un predicado *foaf:isPrimaryTopicOf* con objeto ?topic. La tercera tripleta (figura 3.6, línea 3.), indica que el sujeto ?sujeto debe tener un predicado *dbpedia-owl:abstract* con sujeto ?abstract. La cuarta tripleta (figura 3.6, línea 4.), indica que el sujeto ?sujeto debe tener un predicado *dbpedia-owl:wikiPageExternalLink* con obejto ?pageExternal. Esta tripleta será el resultado principal, ya que indica los links que se pueden consultar y que están relacionados con el valor de la variable $term. Esta variable toma el valor de cada una de las entidades que fueron extraídas de los archivos publicados en Moodle luego de aplicar las técnicas de Extracción de Información (NER) especificadas en la sección 3.1. Para que las entidades puedan ser relacionadas con otros recursos disponibles en DBpedia, éstas deben modelarse en formato RDF al momento de realizar la consulta SAPRQL.

El primer filtro FILTER (regex(?lbl, '".$term."', 'i')) (figura 3.6, línea 5.), retorna los resultados que contengan el término de la variable $term; esto se hace para que el objeto ?lbl tome el valor de la variable $term. El parámetro 'i' indica que no importa si el valor de $term está en mayúscula o minúscula. El segundo filtro (figura 3.6, línea 6.) retorna los resultados que contengan resumenes tanto en inglés como en español; esto se hace con el fin de limitar los resultados para evitar que se obtengan demasiados links repetidos.

Apesar de que la consulta (figura 3.6) está bien formulada, se evidenció que presentaba problemas en cuanto a la conexión con DBpedia, es decir, no arrojaba resultados de forma consistente y no funcionaba de manera eficiente al realizar la búsqueda de palabras claves; para ello se investigó y se identificó que el principal factor que afectaba

la consulta era la complejidad de la misma, a raíz de los filtros aplicados, además por ser público el Endpoint, la consulta SPARQL era denegada en momentos de alto tráfico entregando resultados deficientes y en algunas casos inexistentes.

A raíz de lo anterior se planteó otra consulta (figura 3.7), teniendo en cuenta que para Enero de 2015 DBpedia ofreció el servicio del Endpoint para consultas en español denomidano esDBpedia[10]. Teniendo en cuenta que el Endpoint de esDBpedia arroja resultados tanto en Español como en Inglés, se decidió usarlo porque el proceso de extracción de palabras claves se realiza sobre archivos que contienen textos en Español.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?lbl ?pageExternal
WHERE {
?sujeto rdfs:label ?lbl .
?sujeto dbpedia-owl:abstract ?abstract.
?sujeto dbpedia-owl:wikiPageExternalLink ?pageExternal.
FILTER ( regex(?lbl, '".$term."', 'i') && regex(?abstract, '".$term."', 'i'))
}
```

Figura 3.7: Segunda versión de la consulta.

Esta cosulta funciona tanto para DBpedia como para esDBpedia, lo único que cambia es el Endpoint al que se está conectando.

En la segunda consulta, a diferencia de la primera, se solicitan los valores de ?lbl y ?pageExternal, sólo se aplica un filtro que requiere que ?lbl y ?abstract contengan el valor de la variable $term, y se elimina la segunda tripleta que buscaba los topic.

A pesar de que se mejoraron los resultados luego de aplicar la segunda consulta, aún se presentaban inconsisitencias como por ejemplo al consultar la palabra *"SPARQL"* no arrojaba resultados y se halló que esto era causado porque de acuerdo a la estructura de DBpedia la palabra *"SPARQL"* no contiene el predicado *"dbpedia-owl:abstract"* y además tampoco cumpliría con el filtro que se realiza a ?abstract; por tanto se determinó que la consulta no debería estar sujeta a este predicado, sino cumplir con tener los predicados *"rdfs:label"* y *"dbpedia-owl:wikiPageExternalLink"*; por esta razón se planteó una tercera versión de la consulta, tal como se observa en la figura 3.8.

[10]http://es.dbpedia.org/sparql

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?lbl ?pageExternal
WHERE {
?sujeto rdfs:label ?lbl .
?sujeto dbpedia-owl:wikiPageExternalLink ?pageExternal.
FILTER (regex (?lbl, '".$term."', 'i'))
}
```

Figura 3.8: Tercera versión de la consulta.

Esta tercer consulta, a diferencia de la segunda, se aplica un filtro que requiere que ?lbl contengan el valor de la variable $term y se elimina la segunda tripleta que buscaba los abstratc.

También en esta consulta se observó que existen palabras claves como por ejemplo ”Perceptrón” que no arrojaron resultados de páginas relacionadas, ya que no todas las tripletas en esDBpedia contienen el predicado *dbpedia-owl:wikiPageExternal*; por ende, se porpueso una cuarta consulta de carácter opcional, donde la segunda tripleta de la cláusula WHERE se reemplaza por *?sujeto dbpedia-owl:abstract ?abstract* (figura 3.9), la cual retorna la definición (?abstract) de la palabra consultada, sólo si ésta no arroja resultados de páginas relacionadas.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?lbl ?abstract
WHERE {
?sujeto rdfs:label ?lbl .
?sujeto dbpedia-owl:abstract ?abstract.
FILTER (regex (?lbl, '".$term."', 'i'))
}
```

Figura 3.9: Consulta Opcional.

La tercer consulta que se convierte en la consulta principal, la cual entrega como resultado URIs que en algunos casos están duplicadas, no están relacionadas con la palabra clave o no están disponibles en la Web. Puesto que no hay un criterio para identificar cuales de los resultados son los más relevantes para mostrar al usuario, sólo se optó por eliminar los duplicados. De esta manera el usuario tendrá acceso al resto de URIs que arroje la consulta. La forma en cómo se presentan estos resultados se explica en la sección 3.3

3.3 Adpatción de Moodle a un SCMS

En esta sección se explica las modificaciones realizadas a la plataforma Moodle para llevar a cabo el enriquecimiento semántico del contenido publicado. Para ello se tiene en cuenta el esquema general de la arquitectura, el código y la base de datos; además de las herramientas empleadas para conectar Moodle, DBpedia y NER.

3.3.1 Esquema General

La arquitectura de Moodle requirió de algunas modificaciones para realizar el enriquecimiento semántico de su contenido, con el fin de lograr la adaptación de la arquitectura de un LMS existente a la arquitectura de un SCMS (descrita en la sección 2.2, subsección 2.2.1, figura 2.8), para así gestionar e integrar los contenidos que sean enriquecidos semánticamente.

La adaptación de la arquitectura se basó en tres de las capas de referencia para un SCMS: *i) Presentation & Interaction* que presenta al usuario final el conocimiento, en este caso la forma en como se visualizan los resultados de la consulta a DBpedia (sección 3.3.4), y su interacción directa con este, *ii) Semantic Lifting* a través de la cual se extrae el conocimiento, en esta capa se realiza el proceso de Extracción de Información haciendo uso de NER y se crea un contenido a nivel semántico mediante la consulta a la Web de Datos (en este caso DBpedia) usando el formato RDF, *iii) Persistence* para almacenar el nuevo conocimiento, es decir donde se guardan los resultados obtenidos después del proceso de extracción y consulta. La firuga 3.10 muestra la arquitectura de Moodle después del enriquecimiento semántico del contenido.

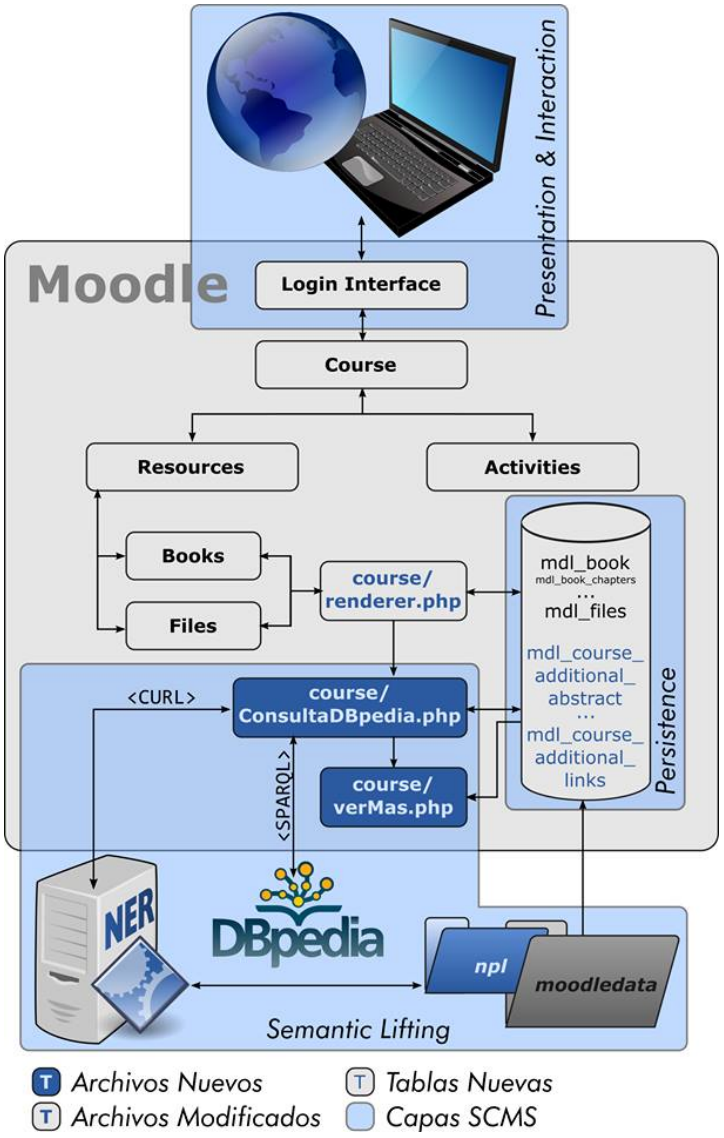


Figura 3.10: Arquitectura de Moodle extentida.

La figura 3.11 muestra el proceso de consulta a *Material Adicional* luego de dar click en el enlace; este proceso inicia con el llamado al archivo consultaDBpedia.php desde el archivo renderer.php para su ejecución; paso (1) de la figura 3.11. Este archivo tiene en cuenta la estructura de Moodle para programar sus archivos; la ubicación de éste yace en la ruta *moodle/course/consultaDBpedia.php*.Â´Además, la arquitectura contiene un servicio de NER externo que es el encargado de la Extracción de Información, el cual se encuentra desarrollado en Java y se comunica con php por medio de un servicio RESTful (sección 3.3.4). En la carpeta *moodledata* se encuentra almacenada la carpeta *nlp*, la cual contien los modelos entrenados y los datos de entrenamiento para el perceptrón. El

archivo consultaDBpedia.php recibe del servicio de NER las palabras clave extraídas, paso (2). Igualmente contiene la consulta y la conexión a la fuente de datos DBpedia, y los resultados que retorna de la misma, paso (3). También envia todos los resultados de la consulta a la base de datos de Moodle para su posterior almacenamiento (sección 3.3.2), paso (4). Y finaliza enviando y mostrando los resultados al usuario, paso (5). Una vez visualizados los resultados de la consulta al link *Material Adicional*, se puede visualizar un nuevo link llamado *Ver Todos*, el cual hace un llamado al archivo verMas.php desde el archivo consultaDBpedia.php, paso (6); quién realiza la consulta a la base de datos de Moodle para obtener los resultados de la consulta a DBpedia de una palabra clave específica.

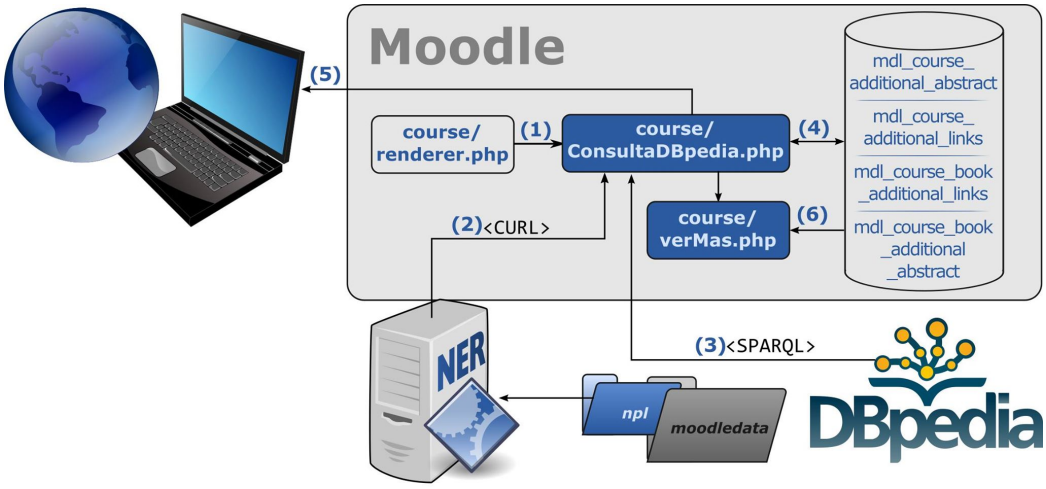


Figura 3.11: Proceso de consulta a *Material Adicional*.

Estos cambios se realizaron con el fin de brindar además de un espacio de almacenamiento para los contenidos; también ofrecer mecanismos de extracción, asociación y visualización de información.

3.3.2 Base de Datos de Moodle

En la base de datos de Moodle se agregaron 4 nuevas tablas para almacenar los resultados de la consulta a DBpedia de cada uno de los recursos a los que se les realizó el proceso de Extracción de Información. Estas nuevas tablas están distribuidas de la siguiente manera:

- Se agregaron 2 tablas para el recurso Libro, una tabla para guardar las URI's como resultado de la consulta a DBpedia llamada *mdl_course_book_additional_links* y la

otra para guardar las definiciones como resultado de la consulta opcional llamada *mdl_course_book_additional_abstract*. Estas tablas están conformadas por un id automático, el id del libro, la palabra clave, y el resultado de la consulta (ya sea una URI o una definición).

- Se agregaron 2 tablas para el recurso Archivo, una tabla para guardar las URI's como resultado de la consulta a DBpedia llamada *mdl_course_additional_links* y la otra para guardar las definiciones como resultado de la consulta opcional llamada *mdl_course_additional_abstract*. Estas tablas están conformadas por un id automático, el id del archivo, la palabra clave, y el resultado de la consulta (ya sea una URI o una definición). Cabe aclarar que los recursos que conformaran estas tablas son los que tienen extensión .pdf, .ppt, .pptx, .xls, .xlsx, .doc, .docx, .odt y .txt.

3.3.3 Código de Moodle

Tras la modificación de la base de datos, se procedió a modificar la clase que modela los archivos subidos a moodle, es decir *stored_file* en la que, siguiendo el paradigma orientado a objetos usado en esta clase, se añadió una variable privada *$add_content* y métodos *set_add_content()* y *get_add_content()*.

En la clase *file_storage*, se añadió la función *get_file_instance_url()*, la cual es una copia de *get_file_instance()* en la que se consulta si el archivo tiene material adicional consultado y de ser así, guarda en la variable *$add_content* el string "true", de lo contrario, guarda "false". También se modificó el método *get_area_files()* de la misma clase *file_storage* para que llame a *get_file_instance_url()* en lugar de *get_file_instance()*. La función *get_area_files()* en particular es llamada por el archivo *lib.php* en la carpeta del módulo *resource* que modela los recursos; dentro de este php se almacena la instancia de *stored_file* ahora con la variable que indica si tiene o no material adicional relacionado.

En el archivo *renderer* se añadió el método *public function additional_link()* para que consulte su ícono. Comparar el nombre del ícono es la forma más fácil de identificar el tipo de archivo desde el curso. La otra forma sería acceder al recurso al que hace referencia el *cm_info* del módulo actual, luego acceder al *stored_file* dentro de dicho recurso y ahí acceder a la variable *mimetype*, que viene directamente de la base de datos y aun así esta variable contempla diferencias entre archivos .doc, .docx, .xlsx, etc; lo que es irrelevante para la lectura.

El proceso descrito anteriomente se puede observar en la figura 3.12.

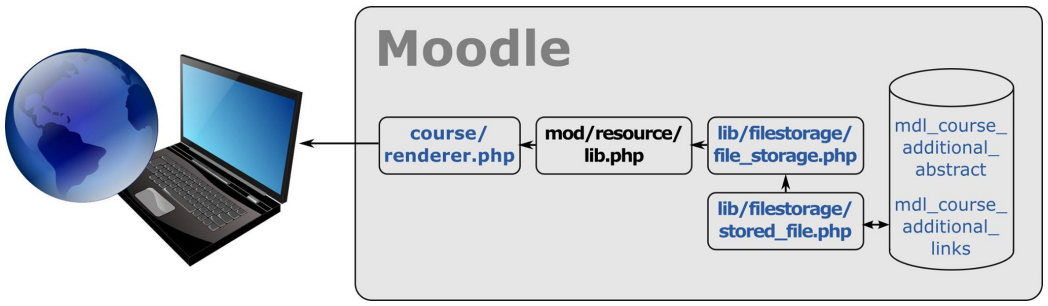


Figura 3.12: Extensión de *Material Adicional* a los archivos de Moodle.

Si el ícono corresponde a un archivo legible (i.e., con extensión pdf, txt, doc, docx, xls, xlsx, ppt, pptx o odt) entonces genera un link al archivo *consultaDBpedia.php* (sección 3.2), el link pasa por variable el id del curso, el tipo de archivo, el nombre del archivo encriptado o *contenthash* y si tiene o no contenido adicional. A su vez el archivo *consultaDBpedia.php* genera un link al archivo *verMas.php* (sección 3.3.4) y el link pasa por variable el id del curso, el id del archivo, el tipo de archivo y la palabra clave; para ver todos los resultados de la consulta a DBpedia.

En el caso del recurso Libro, no es necesario llamar al servicio de lectura, sino únicamente al de NER. Una vez que se consultan y al tiempo que se imprimen, los resultados se guardan en la base de datos, para lo que es necesario crear una instancia de *stdClass* con los mismos campos de la tabla en la que se va a guardar y luego se llama a la función de *$DB insert_record()* que recibe como parámetros el nombre de la tabla en la que se va a guardar y el objeto *stdClass*.

3.3.4 Conexión Moodle - DBpedia - NER

Se desarrolló un servicio en Java encargado de las funciones de reconocimiento de entidades nombradas, más precisamente palabras clave, el cual recibe la dirección del archivo del que deben extraerse palabras clave para extraer el texto. La funcionalidad del servicio, en lo que a extracción del texto y de las palabras clave respecta, ha sido previamente explicado en la sección 3.1.

En el archivo *consultaDBpedia.php* si el archivo tiene ya contenido adicional, este se consulta directamente de la base de datos para la visualización, de lo contrario es necesario llamar al servicio encargado de las funciones de lectura y NER por medio de CURL para que extraiga el texto del archivo y luego reconozca las palabras clave.

El retorno del servicio se consulta en DBpedia palabra por palabra en busca de links o abstracts y el resultado de la consulta se imprime en un acordeón con las palabras claves como títulos de las secciones y la etiqueta [URIS] en caso de encontrarse URI's o solo la palabra clave si sólo se encontraron abstracts.

Al dar click sobre el título, se despliega la sección con las URIs o el abstract. En el caso de la URI's se mostraran los 10 primeros resultados y cómo es posible que no se alcancen a visualizar todos los resultados de algunas palabras claves (i.e., que existan más de 10 URIs) se agregó un link llamado *Ver Todos*, el cual llama al archivo *verMas.php* que se encarga de realizar una consulta a la base de datos de Moodle para mostrar los resultados completos de la palabra seleccionada (i.e., muestra todas las URI's relacionadas que se obtuvieron al consultar en DBpedia la palabra).

Esta sección finaliza con una breve visualización de Moodle después del enriquecimiento semántico del contenido, en cual se observa en la figura 3.13 como acceder al link de *Material Adicional*, en la figura 3.14 se muestra la forma en que se visualizan los resultados después del proceso de extracción y consulta de un documento en PDF sobre Ingeniería de Software (http://www.wolnm.org/apa/articulos/ingenieria_software.pdf), y en la figura 3.15 se ilustra la opción *Ver Todos* de una palabra seleccionada.

Figura 3.13: Ejemplo de un Curso con *Material Adicional*.

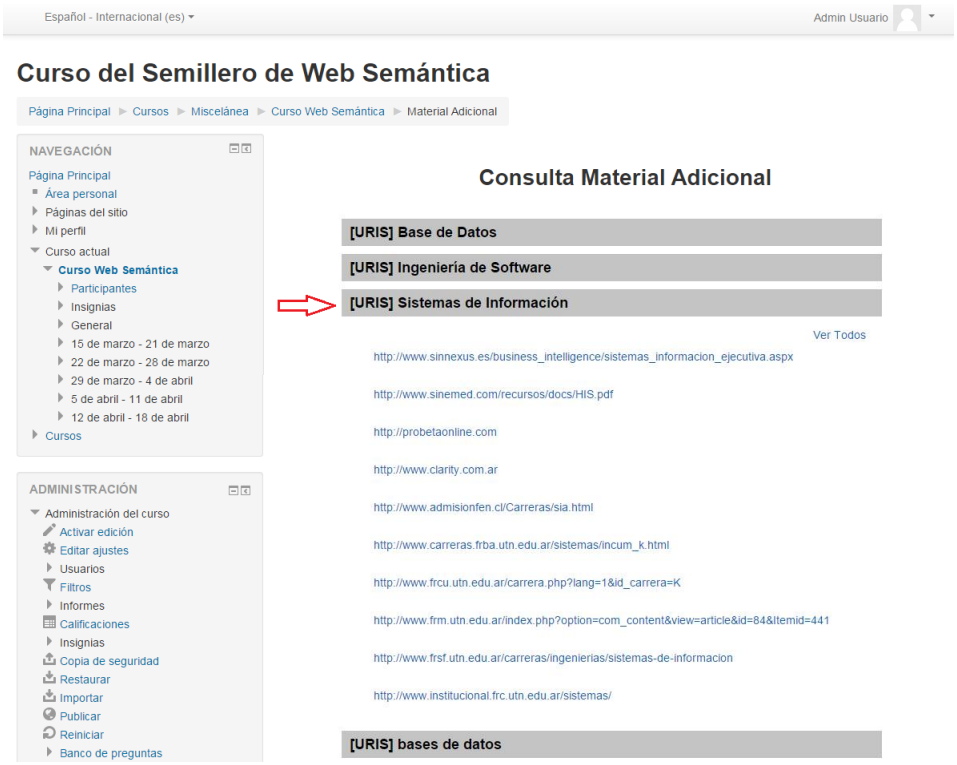


Figura 3.14: Ejemplo de consulta de *Material Adicional*.

Figura 3.15: Ejemplo *Ver Todos* referente a la palabra *Sistemas de Información*.

3.4 Objetivo Adicional

Como se indicó en las secciones anteriores, los procesos de Extracción de Información y de consulta a la Web de Datos, se llevarón a cabo para textos en idioma español. Y además para el entrenamiento del perceptrón se generó un conjunto de entidades representando tópicos de Ciencias de la Computación. Una vez culminados los objetivos principales del presente trabajo, se decidió realizar los procesos de Extracción de Información y de consulta a la Web de Datos para textos en Inglés, y ampliar los tópicos para el entrenamiento del preceptrón.

1. Tópicos para el entrenamiento del preceptrón.

   Para la ampliación del entrenamiento del perceptrón, se incluyó el área de Física, para llevar a cabo este inclusión se generó un conjunto de entidades, se realizó un documento recopilando fragmentos de texto donde estas entidades eran mencionadas al menos una vez y se continuó con el proceso ya explicado en la sección 3.1. También se realizaron las pruebas pertinentes para el entrenamiento y posterior selección de plabras claves, en las que se observó que el proceso de extracción fue exitoso, ya que selecciona palabras relevantes al tema en cualquier tipo de texto (e.i., .pdf, .ppt, .pptx, .xls, .xlsx, .doc, .docx, .odt y .txt).

2. Procesos de Extracción de Información y consulta a la Web de Datos para textos en Inglés

   Una vez se ha leído el documento, se ejecutan las detecciones de plabras clave relacionadas a los temas de Sistemas y Física en Español y en Inglés de manera secuencial, para lo cual se crean cuatro instancias de la clase NER con el fin de realizar tareas de detección, para las palabras clave detectadas de cada tema (i.e., Informática - Español, Informática - Inglés, Física - Español, Física - Inglés), ya sean exactas o no.

   Con el fin de reducir la aparición de palabras no exactas potencialmente no relacionadas, provenientes de la detección de temas no presentes en el texto (i.e., en un texto de Sistemas en español, el detector de Física en inglés no debe detectar nada exacto, ya que se espera que en este texto no deberían estar presentes palabras en inglés de Física, sin embargo es posible que se detecten palabras no exactas por similitud que muy probablemente no serán relevantes), se seleccionará la instancia con más resultados exactos almacenados para retornar todos sus resultados, más los resultados exactos de la instancia del mismo tema en otro idioma (e.g., si se selecciona la instancia encargada de detectar palabras relacionadas a Informática en español, se retornarían todos sus resultados, más aquellos resultados exactos almacenados en la instancia de Informática en inglés).

3.5 Pruebas

Para llevar a cabo las pruebas a la aplicación de Moodle después de su extensión semántica se tienen en cuenta 3 factores; el primero es analizar los resultados que se obtienen después del proceso de extracción y consulta, el segundo es observar los resultados del proceso de extracción de información y el tercero es tener en cuenta la experiencia y percepción del usuario con un grupo reducido de estudiantes.

1. Analizar los resultados que se obtienen después del proceso de extracción y consulta.

   En esta prueba se analizaron las URIs que se muestran como resultado de consultar una palabra clave, con el fin de determinar qué porcentaje de estos resultados se encuentran relacionados con la temática consultada; estas pruebas se realizaron de forma manual, es decir, que de acuerdo a la percepción del usuario se escogieron cuáles de los resultados eran las mejores. Para ello, se seleccionaron al azar 25 palabras claves; cuya mecánica consiste en ir revisando los resultados de cada palabra clave y observando cuántos de estos resultados están relacionados y aportan algo más a la palabra clave consultada; de allí se determina un porcentaje de relación válido entre los resultados y la palabra clave.

   En la tabla 3.2 se observa que la mayoría de los resultados de las palabras claves superan el 50% de enlaces relacionados y que tan solo 3 palabras claves tienen resultados relacionados inferiores al 50%; cabe aclarar que la coincidencia de los resultados con la palabra clave varía según la temática, ya que si la palabra se presta para ambigüedades posiblemente la cantidad de aciertos disminuyan, pero si la palabra es de un tema muy específico se observan resultados mucho más precisos y relacionados con la palabra consultada.

   Al final se sumaron los porcentajes de cada palabra clave para determinar un posible comportamiento en los resultados de cualquier búsqueda y se determinó que en promedio el 72,62% de las URI's como resultado de la consulta, son muy relevantes y están relacionadas con la palabra clave consultada.

Palabra Clave	Número Total de Resultados	Número de Resultados Relacionados	% de Resultados Relacionados
Ingeniería de software	15	7	46,66
Proceso para el desarrollo de software	5	2	40,00
Inteligencia Artificial	31	21	67,74
Red Neuronal Artificial	9	6	66,66
Red Neuronal	10	7	70,00
Web Semántica	11	8	72,72
SPARQL	9	5	55,55
SQL	147	80	54,42
Representational State Transfer	13	6	46,15
Programación Orientada a Objetos	1	1	100,00
Programación con restricciones	11	7	63,63
Espacio de nombres	3	3	100,00
Base de datos orientada a grafos	18	14	77,77
Procesamiento de Lenguajes Naturales	19	14	73,68
Algoritmo Genético	13	10	76,92
Programación genética	13	11	84,61
Algoritmo Evolutivo	3	2	66,66
Autómata finito	1	1	100,00
Resource Description Framework	8	6	75,00
Sistema experto	9	6	66,66
Minería de Datos	8	5	62,50
Servicio web	5	5	100,00
Aprendizaje automático	22	15	68,18
Procesamiento digital de imágenes	5	4	80,00
Serialización	3	3	100,00

Tabla 3.2: Resultados pruebas aplicadas al proceso de extracción y consulta.

2. Observar los resultados del proceso de Extracción de Información.

   Las pruebas de NER se realizaron con 5 archivos de temas y longitudes diferentes relacionados a sistemas, a saber: Redes Neuronales, Ingeniería de Software, Web de Datos, Bases de datos relacionales y Sistemas Expertos. Estas pruebas se llevaron a cabo de forma manual, es decir, que con base en la percepción del usuario se seleccionaron los resultados más relavantes de acuerdo a los textos extraídos. Se evaluó la cantidad de palabras seleccionadas y la cantidad de estas que son relevantes al texto para calcular el porcentaje de acierto de selección. De

igual manera se consideraron las palabras ignoradas, es decir detectadas por el perceptrón pero eliminadas por los filtros, ya sea por contener números, por tener preposiciones al final o por no incluidos en la selección aleatoria de resultados no exactos; se evaluó la cantidad de palabras ignoradas y la cantidad de estás que no hubieran sido relevantes al texto para calcular el porcentaje de acierto de los filtros y finalmente, el promedio de ambos porcentajes sería el porcentaje de acierto del sistema de NER que fue desarrollado (ver tabla 3.3).

Archivo	Seleccionadas	Seleccionadas Correctas	% Acierto Selección	Ignoradas	Ignoradas Correctas	% Acierto Ignoradas	% Acierto General
1	8	6	75,00	18	17	94,44	**84,72**
2	9	7	77,78	14	14	100,00	**88,89**
3	13	12	92,31	18	18	100,00	**96,15**
4	11	8	72,73	19	18	94,74	**83,73**
5	5	4	80,00	3	2	66,67	**73,33**
Total	**46**	**37**	**80,43**	**72**	**69**	**95,83**	**88,13**

Tabla 3.3: Resultados pruebas aplicadas al proceso de Extracción de Información.

De la tabla 3.3, se puede concluir que: en promedio, 4 de 5 resultados del sistema de NER desarrollado son correctos, 19 de 20 palabras eliminadas no eran relevantes y el sistema en general presenta un 88,13% de aciertos en su funcionamiento.

3. Tener en cuenta la experiencia y percepción del usuario con un grupo reducido de estudiantes.

   Para llevar a cabo este tipo de prueba se realizó una comparación entre el Moodle original y el Moodle enriquecido semánticamente, con la finalidad de observar la percepción del estudiante, su impacto académico y analizar qué tanto puede contribuir la aplicación a su aprendizaje.

   Este análisis se realizó con un número de 20 estudiantes de Ingeniería de Sistemas de la Universidad del Valle Sede Tuluá, ya que como se ha mencionado anteriormente el perceptrón para la extracción de información está entrenado con temas Computacionales; este número de estudiantes fueron divididos en 2 grupos de 10 personas cada uno, con el fin de que el grupo 1 se matriculara en el Moodle original y el grupo 2 en el Moodle enriquecido semánticamente; ambos grupos fueron matriculados en un curso y previamente se asignó un usuario y una contraseña a cada estudiante para su acceso a Moodle.

En ambas aplicaciones se encontraba un curso llamado Curso del Semillero de Web Semántica, el cual tenía una duración de 5 semanas y los estudiantes matriculados en este curso podían acceder a consultar el material publicado. En la última semana se realizó una evaluación final del curso; con la que se pretendió analizar y comparar las calificaciones de los estudiantes.

Para realizar las pruebas en el Moodle original se hizo uso de Gnomio[11], el cual presta un servicio de Moodle online gratuito, que permite crear cursos y matricular usuarios sin ningún tipo de restricción; el curso se encuentra en el siguiente link https://websemantica.gnomio.com (figura 3.16) en esta prueba los estudiantes realizan el curso de forma virtual y si el estudiante tenía alguna duda se comunicaba con el docente para ser solucionada.

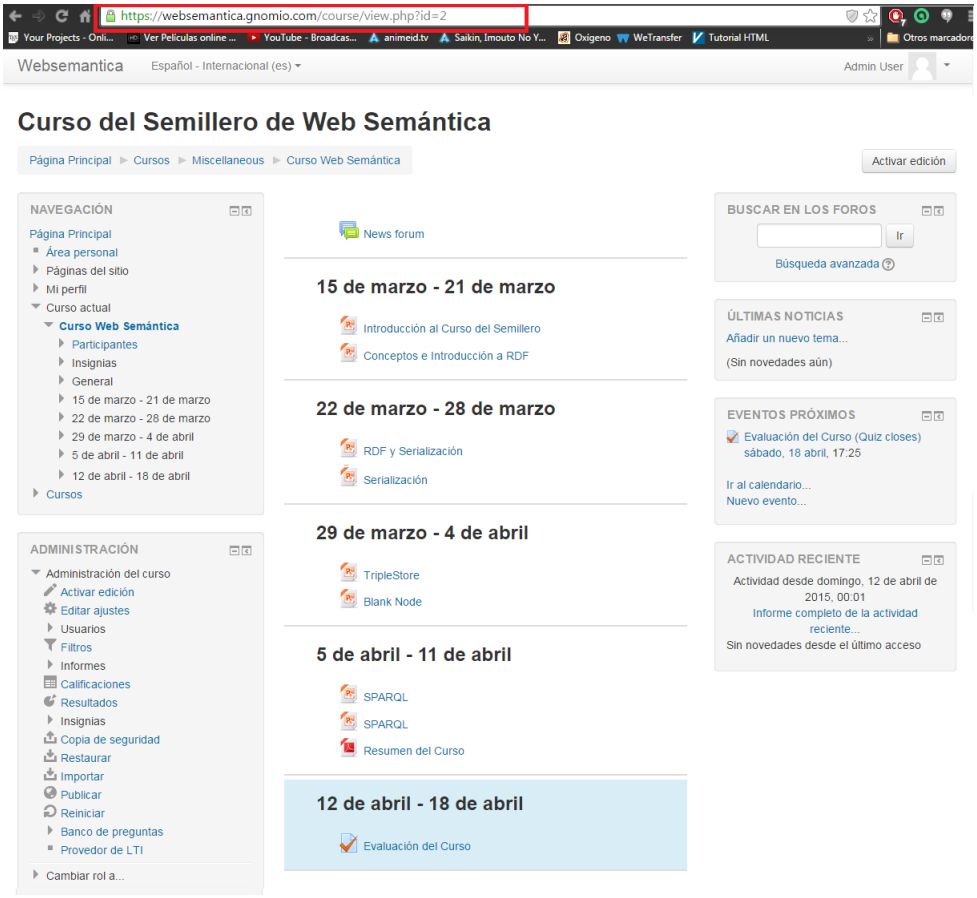


Figura 3.16: Prueba Moodle original - Online.

[11]https://www.gnomio.com/

Las pruebas en el Moodle enriquecido semánticamente se realizaron con una metodología de evaluación semi-presencial, es decir que el curso se realizó de manera asistida donde se estudiaban las clases publicadas en Moodle (figura 3.17) y se solucionaban las dudas existentes; para ello se dispuso de 3 equipos de cómputo para que los estudiantes pudieran interactuar y realizar el curso de manera offline y al finalizar el curso se realizó la evaluación de forma individual alternando los 10 estudiantes en cada equipo de cómputo.

Figura 3.17: Prueba Moodle Enriquecido Semánticamente.

Como el contenido del curso se empleó en ambas versiones de Moodle (original y enriquecido semánticamente) nos permitió hacer un análisis comparativo basándonos en la evaluación que se realizó al final del curso, para determinar que tanto puede influir el enriquecimiento semántico de Moodle en el desempeño académico de los estudiantes. En la tabla 3.4 se muestra la calificación más alta, la más baja y un promedio total de cada curso, la calificación más alta permitida en esta evaluación es de 100 puntos.

Grupo	Calificación más alta	Calificación más baja	Promedio de calificación del curso
Grupo 1 (Moodle original)	100,00	58,00	76,20
Grupo 2 (Moodle enriquecido semánticamente)	100,00	84,00	95,6

Tabla 3.4: Resultados de Moodle original y Moodle enriquecido semánticamente.

Al observar los resultados de la tabla 3.4, se evidencia que las calificaciones del curso en el Moodle enriquecido semánticamente son más altas que las del Moodle original, por tanto se puede concluir que el enriquecimiento semántico de Moodle ha contribuido en un 25,46% para mejorar el desempeño académico de los estudiantes de Ingeniería de Sistemas de la Universidad del Valle Sede Tuluá.

Por otra parte, a los estudiantes que conformarón el grupo 2 se les realizó una encuesta con la que se pretendió analizar el punto de vista del usuario después de interactuar con Moodle enriquecido semánticamente; la tabla 3.5 muestra la encuesta realizada a los usuarios y las respuestas brindadas por los mismos.

Pregunta	Respuestas	Cantidad de respuesta	Porcentaje (%)
1. ¿Le parecen adecuadas al texto, las palabras que se están consultando?	Adecuadas	10	100
	Poco adecuadas	0	0
	Inadecuadas	0	0
2. ¿Qué tan relevantes pueden ser los resultados ofrecidos por el link Material Adicional?	Muy relevantes	6	60
	Relevantes	4	40
	Poco relevantes	0	
3. ¿Qué tan útil puede ser la aplicación para contribuir en su proceso de aprendizaje y comprensión del tema?	Muy útil	10	100
	Poco útil	0	0
	Totalmente inútil	0	0
4. ¿Con que frecuencia consultaría la opción de Material Adicional?	Mucha frecuencia	7	70
	Poca frecuencia	3	30
	Nunca	0	0
5. Su percepción respecto a la modificación realizada a Moodle es:	Muy buena	5	50
	Buena	5	50
	Regular	0	0
	Mala	0	0
	Muy mala	0	0
6. ¿Le parece adecuada la forma en cómo se visualizan los resultados?	Adecuado	10	100
	Poco adecuado	0	0
	Inadecuado	0	0

Tabla 3.5: Resultados encuesta de usuario para Moodle enriquecido semánticamente.

Como se muestra en la tabla 3.5, según la percepción de los usuarios que evaluaron la plataforma extendida, el enriquecimiento semántico del contenido de Moodle puede llegar a influir positivamente en el proceso de aprendizaje de los estudiantes, motivando su interés por el uso de plataforma y el aprovechamiento de los recursos que se brindan en la misma.

Además que contribuye en pro de la comprensión de los temas vistos durante el curso, haciendo que se pueda mejorar la calidad académica de los estudiantes en las instituciones educativas; según la opinión de éstos.

Capítulo 4

Conclusiones y trabajos futuros

En este capítulo se presentan las conclusiones y trabajos futuros del trabajo de grado con respecto a los objetivos cumplidos.

4.1 Conclusiones

- Durante el proceso de Extracción de Información, se evidenció que durante el entrenamiento del perceptrón, en el archivo que contiene el contexto de las palabras empleadas para el entrenamiento, es necesario que cada palabra aparezca mínimo 4 veces, si no es así es más difícil para el perceptrón identificar estas palabras en otro contexto y más aún si las palabras están como títulos o si son palabras ”sueltas” (i.e., que no se encuentran dentro de un párrafo).

- Se evidenció que, a pesar de que NER no está pensado inicalmente para reconocer tópicos académicos presentes en textos de la misma naturaleza; sino que estaba direccionado para reconocer entidades tales como nombres de personas, organizaciones, lugares, expresiones temporales, cantidades, valores monetarios, porcentajes, etc. Se puedo observar que sus cambios fueron exitosos y que se adapto de manera correcta para extraer infomación de textos académicos.

- Las consultas a DBpedia pueden ser muy exactas si se consulta algo muy específico cuya consulta sea con predicados y sujetos específicos (e.g., Nombres y sobrenombres de músicos de jazz latino). Pero en nuestro caso, donde se consultan palabras claves extraídas de documentos, la estructura de la consulta a DBpedia debe ser lo más ”abierta” posible a cualquier tema que se esté consultando, ya que DBpedia cuenta con una gran cantidad de categorías y estas a su vez pueden

estar inmersas en otras, por tanto si la consulta se ciñe a determinada categoría, se excluirían resultados y a su vez una consulta muy ”abierta” también ocasionarían que los resultados sean muy dispersos.

- Con respecto a los resultados de la consulta a DBpedia; se observó que hay cierto grado de incertidumbre con relación a la relevancia de los resultados, ya que en algunos casos la consulta retorna resultados que no están muy relacionados con lo que se está consultando, o estos resultados devuelven recursos (páginas web) que podrían no estar disponibles.

- A pesar que se mejoraron las consultas, cabe resaltar que en horas de alto tráfico en la red, las consultas a DBpedia se hacen más lentas muy probablemente debido a la alta concurrencia de la misma.

- El proceso de adaptación de la arquitectura de un LMS a la de un SCMS resultó natural, ya que la arquitectura de Moodle permitió hacer posible las modificacicaciones pertinentes para el enriquecimiento semántico del contenido, logrando asi, identificar y adaptar las capas pertenecientes a la misma.

- El código de Moodle requiere de un conocimiento amplio para su modificación, ya que la documentación ofrecida por la comunidad de Moodle no siempre es lo suficientemente clara y en algunos casos la programación bajo su estándar no es tan estrcita, esto hace que la adpatación en algunas versiones cambien totalmente la forma de programar la extensión.

4.2 Trabajos futuros

Como trabajos futuros se consideran:

- Implementar un sistema que detecte palabras implícitas en un texto; es decir, que con base en las palabras claves que se extraen actualmente, se identifiquen temas de estudio relacionados con las mismas, y que estos nuevos términos se agrupen como palabras claves. Ejemplo: Palabras claves: Corazón, Venas, Arterias. Temas de estudio: Cardiología, Sistema Cardiovascular, etc. Ya que en algunos casos es más relevante el tema de estudio que la palabra clave. Para el desarrollo de este sistema se podría hacer uso de Redes Neuronales o Algoritmos de Clustering.

- Ampliar el sistema de Extracción de Información con el fin de permitir el análisis y extracción no solo de documentos (i.e, .pdf, .ppt, .pptx, .xls, .xlsx, .doc, .docx, .odt y .txt), sino también de imágenes, archivos comprimidos, entre otros.

- Extender el sistema para realizar consultas a otras fuentes de datos aparte de DBpedia, con el fin de brindar mayor infromación y posiblemente información en tiempo real, como por ejemplo Twitter, Facebook, la BBC, Library of Congress, Freebase, Geonames, entre otros.

- Realizar una interfaz de usuario para permitir la actualización de los datos almacenados en la base de datos de Moodle que provienen de la consulta a DBpedia. Ya que en este proyecto se consulta solo una vez a DBpedia y estos resultados se guardan en la base de datos para que cuando el usuario los consulte de nuevo la respuesta sea más rápida; por ende al pasar el tiempo se pueden perder resultados por no estar actualizando periódicamente.

- Desarrollar un mecanismo que depure los resultados de la consulta a DBpedia (i.e, las URIs relacionadas) y puedan categorizarlos de una mejor manera con respecto a su contenido. Para ello, se propone hacer uso del Algoritmo de PageRank empleado por el buscador Google que consiste en dar un valor numérico a las páginas web, para así determinar la relevancia de una u otra dentro de un buscador Web.

Referencias

[1] T. Berners-Lee, J. Heath, and O. Lassila. The semantic web. *Scientific America.*, (501):29–37, 2001, Mayo.

[2] J. Betetta, M. Castro Díaz, C. Flores, and R. Palavecino. Evaluación de las características y comparación de los sistemas de gestión de contenidos. *In VII Workshop Ingeniería de Software.*, 2010.

[3] C. Bizer, R. Cyganiak, and T. Heath. How to publish linked data on the web. `http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/LinkedDataTutorial/`, 2007.

[4] C. Bizer, T. Heath, and T. Berners-Lee. Linked data: The story so far. in international journal on semantic web and information systems. 5(3):1–22, 2009.

[5] J. Bonue. Plataforma abiertas de e-learning para el soporte de contenidos educativos abiertos. *In RUSC - Revista de Universidad y Sociedad del Conocimiento.*, 4:36–47, 2007.

[6] T. Chaudhary, C. Ambrín-Guimerá, and G. Ruiz. Hacia una nueva generación de campus virtuales: Integración de plataformas en el campus virtual. `http://eprints.ucm.es/11274/`, 2010. Universidad Complutense de Madrid.

[7] F. Christ and B. Nagel. A reference architecture for semantic content management systems. *Conference: Enterprise Modelling and Information Systems Architectures: Proceedings of the 4th International Workshop on Enterprise Modelling and Information Systems Architectures, (EMISA 11)*, P-190:135–148, 2011.

[8] ¿Qué es Moodle? ¿Para qué? Universidad luterna salvadoreña. `http://www.uls.edu.sv/pdf/manuales_moodle/queesmoodle.pdf`, Accesado el 3 de Octubre de 2014.

[9] B.; Florian and M. Kaltenbock. *Linked Open Data: The Essentials - A Quick Start Guide for Decision Makers.* In The Semantic Web Company., Vienna, Austria., 2012.

[10] M.J. Garcá Alba. Análisis del desarrollo de extensiones para moodle: Desarrollo de un módulo para la gestión de laboratorios docentes. http://www2.uah.es/libretics/files/GruposLab.pdf, Diciembre 2010. Universidad de Alcalá.

[11] R. García, J. M. Gimeno, F. Perdrix, R. Gil, and Oliva M. The rhizomer semantic content management system. *Emerging Technologies and Information Systems for the Knowledge Society.*, 5288:385–394., 2008.

[12] N. Heino, S. Tramp, and S. Auer. Managing web content using linked data principles combining semantic structure with dynamic content syndication. *Computer Software and Applications Conference (COMPSAC).*, 1(IEEE 35th Annual):245–250, 18–22., 2011.

[13] D. M. Herzig and B. Ell. Semantic mediawiki in operation: experiences with building a semantic portal. *In Proceedings of the 9th international semantic web conference on The Semantic Web*, Part II:114–128, 2010.

[14] F. Izaurieta and Saavedra C. Redes neuronales artificiales. http://www.uta.cl/charlas/volumen16/Indice/Ch-csaavedra.pdf. Universidad de Concepción.

[15] F. Massa, V. Menéndez, and Garcilazo J. Diseño de una extensión de moodle para búsquedas semánticas de recursos digitales de aprendizaje. *Universidad Autónoma de Yucatán*, 1:3–5, 2013, Septiembre.

[16] Moodle. Blocks-moodle docs. http://docs.moodle.org/dev/Blocks, Accesado el 25 de Abril de 2014.

[17] Moodle. Developer docs. http://docs.moodle.org/dev/Developerdocumentation, Accesado el 25 de Abril de 2014.

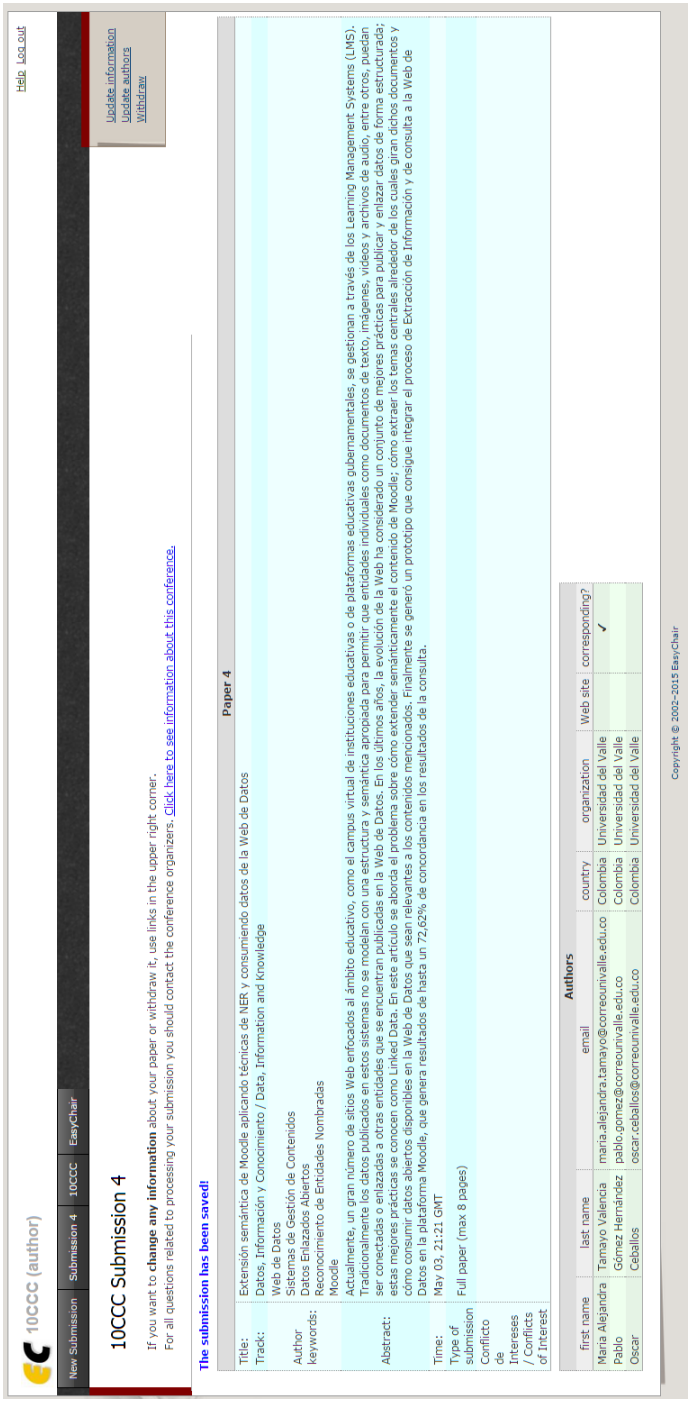
[18] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge unifying wordnet and wikipedia. *In Proceedings of the 16th international conference on World Wide Web.*, pages 697–706, 2007, Mayo.

ANEXOS

ANEXO A: Certificación RREDSI 2014

Certificación participación en RREDSI 2014

ANEXO B: Constancia de envío de Resumen a 10CCC



10CCC (author)

Help Log out

New Submission | Submission 4 | 10CCC | EasyChair

Update information
Update authors
Withdraw

10CCC Submission 4

If you want to **change any information** about your paper or withdraw it, use links in the upper right corner.
For all questions related to processing your submission you should contact the conference organizers. Click here to see information about this conference.

The submission has been saved!

Paper 4

Title:	Extensión semántica de Moodle aplicando técnicas de NER y consumiendo datos de la Web de Datos
Track:	Datos, Información y Conocimiento / Data, Information and Knowledge
Author keywords:	Web de Datos Sistemas de Gestión de Contenidos Datos Enlazados Abiertos Reconocimiento de Entidades Nombradas Moodle
Abstract:	Actualmente, un gran número de sitios Web enfocados al ámbito educativo, como el campus virtual de instituciones educativas o de plataformas educativas gubernamentales, se gestionan a través de los Learning Management Systems (LMS). Tradicionalmente los datos publicados en estos sistemas no se modelan con una estructura y semántica apropiada para permitir que entidades individuales como documentos de texto, imágenes, vídeos y archivos de audio, entre otros, puedan ser conectadas o enlazadas a otras entidades que se encuentran publicadas en la Web de Datos. En los últimos años, la evolución de la Web ha considerado un conjunto de mejores prácticas para publicar y enlazar datos de forma estructurada; estas mejores prácticas se conocen como Linked Data. En este artículo se aborda el problema sobre cómo extender semánticamente el contenido de Moodle; cómo extraer los temas centrales alrededor de los cuales giran dichos documentos y cómo consumir datos abiertos disponibles en la Web de Datos que sean relevantes a los contenidos mencionados. Finalmente se generó un prototipo que consigue integrar el proceso de Extracción de Información y de consulta a la Web de Datos en la plataforma Moodle, que genera resultados de hasta un 72,62% de concordancia en los resultados de la consulta.
Time:	May 03, 21:21 GMT
Type of submission	Full paper (max 8 pages)
Conflicto de Intereses / Conflicts of Interest	

Authors

first name	last name	email	country	organization	Web site	corresponding?
María Alejandra	Tamayo Valencia	maria.alejandra.tamayo@correounivalle.edu.co	Colombia	Universidad del Valle		✓
Pablo	Gómez Hernández	pablo.gomez@correounivalle.edu.co	Colombia	Universidad del Valle		
Oscar	Ceballos	oscar.ceballos@correounivalle.edu.co	Colombia	Universidad del Valle		

Constancia de envío de Resumen al 10CCC

10CCC submission 4 Recibidos x

10CCC <10ccc@easychair.org> 16:21 (hace 6 minutos)

para mí

Dear authors,

We received your paper:

Authors : Maria Alejandra Tamayo Valencia, Pablo Gómez Hernández and Oscar Ceballos
Title : Extensión semántica de Moodle aplicando técnicas de NER y consumiendo datos de la Web de Datos
Number : 4
Track : Datos, Información y Conocimiento / Data, Information and Knowledge

The paper was submitted by Alejandra Tamayo <maria.alejandra.tamayo@correounivalle.edu.co>.

Thank you for submitting to 10CCC.

Best regards,
EasyChair for 10CCC.

10CCC
10ccc@easychair.org
Mostrar detalles

Correo de notificación de recibido del 10CCC