Universidad Autónoma Metropolitana Unidad Azcapotzalco División Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Proyecto de Integración:

Representación semántica y extracción de información sobre publicaciones en expedientes curriculares

Francisco Alejandro Gudiño Pérez

Matricula: 209306332

Asesores:

Maricela Claudia Bravo Contreras Profesora Asociada, Departamento de Sistemas

José Alejandro Reyes Ortiz Profesor Asociado, Departamento de Sistemas

Trimestre 2014 Otoño

Yo, Maricela Claudia Bravo Contreras, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Yo, José Alejandro Reyes Ortiz, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Yo, Francisco Alejandro Gudiño Pérez, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Contenido

Tabla de Figuras

1 Resumen

En la actualidad existe una gran cantidad de información en los expedientes curriculares, pero la información por si sola y en cantidades muy grandes no es de gran utilidad, se debe de tener un proceso para la información deseada que permita la extracción de forma automatizada pero a la vez que resulte útil para los fines que los usuarios lo requieran.

Este proyecto surge como posible solución para la extracción de información de publicaciones sobre expedientes curriculares para su almacenamiento y recuperación de la misma mediante ontologías. Con el sistema es posible extraer artículos y publicaciones de los expedientes utilizando técnicas de procesamiento natural.

El sistema lee los expedientes curricular en formato PDF de ellos se extrae solo la parte de publicaciones o artículos y sobre lo extraído utilizando reglas gramaticales se extrae aspectos importantes como el año, título y los autores y esos se almacenan en una ontología.

2 Introducción

La extracción de información es una disciplina dentro del procesamiento del lenguaje natural que se considera un tipo de recuperación de información, se concentra en localizar y extraer las partes del texto que contengan información relevante con el fin de satisfacer una necesidad concreta; Además proporcionar dicha información de forma adecuada para su procesamiento, por ejemplo, cierta información de algún expediente curricular.

Los expedientes curriculares están elaborados por elementos como: datos personales, experiencia laboral, experiencia profesional, investigaciones, etc. Existe gran cantidad información que se puede tener en un expediente curricular y ésta es muy importante para las empresas. También se tiene información que no llega a ser útil. En una empresa es muy importante que se cumplan ciertos requisitos en los expedientes curriculares, como investigaciones realizadas o artículos publicados. Para esto se tiene la necesidad de clasificarlos, localizar cierta información o identificar los pre-requisitos necesarios, tareas que son tediosas y consumen mucho tiempo si se realizan de manera manual.

Es muy importante poder contar con la información relevante de un expediente curricular de manera rápida y eficiente, es por eso que, en este proyecto se propone extraer la información sobre investigaciones a partir de los expedientes curriculares descritos en lenguaje natural en español utilizando técnicas de minería de textos[1]. La información extraída será almacenada para facilitar la búsqueda de información relevante

[1] Proceso de extracción de información y patrones de comportamientos que permanecen ocultos entre grandes cantidades de textos.

3 Antecedentes

- Proyectos terminales.

Lenguaje de manipulación y minería de datos [1].

El proyecto consiste en la realización de un lenguaje de manipulación de datos, el cual permitirá la clasificación de datos. El objetivo principal es un lenguaje que permita realizar minería de datos, en datos almacenados en un manejador de base de datos.

EL proyecto propuesto la minería de datos se utilizara para la extracción de información sobre investigación en expedientes curriculares y su almacenamiento en una basa de datos.

Sistema de procesamiento de textos de investigación [2].

El sistema propuesto en el proyecto es acerca de la extracción de información de textos de investigación. La implementación permite la extracción automatizada de información relevante de artículos. La información extraditada es de gran utilidad para el proyecto propuesto ya que utiliza minería de datos sobre aspectos de investigaciones

La diferencia más significativa es la extracción ya que esta es sobre expedientes curriculares y no sobre textos de investigación.

Sistema Configurable de Minería Web [3].

La idea principal del proyecto es la de extraer información útil del contenido de los documentos Web utilizando técnicas de minería de datos. El proyecto propuesto utiliza la misma técnica de minería pero en aspectos de investigación y en expedientes curriculares y sobre tod3eo que es extracción de información.

- Tesis

Extracción de información con algoritmos de clasificación [4].

En la tesis se explica la extracción de información, así como algunos algoritmos de extracción, el proyecto a diseña utiliza la extracción con minería de datos y en algún momento podemos utilizar algunos algoritmos descritos.

- Artículo

Modelos clásicos de recuperación de la información [5].

El Artículo es de gran aporte par la propuesta ya que abarca varios aspectos que se van a tomar en cuenta como es la extracción de información. También habla sobre los documentos no estructurados y como es la extracción de estos.

4 Justificación

Existe la necesidad de resolver los graves problemas asociados a la gestión tradicional delos expedientes curriculares en papel, tales como la falta de control, acumulación o pérdida de expedientes, etc. Estos requieren de mecanismos que aprovechen las ventajas de los dispositivos de almacenamiento y recuperación de información y las tecnologías de información que permitan obtener solos aquello que nos interesa de manera automática.

Con el sistema propuesto tendrá grandes beneficios ya que almacenara la información deseada y así será más fácil poder tener acceso a ella y poder manipularla con más facilidad.

Se implementará una aplicación con técnicas de minería de textos, específicamente aprendizaje automático con reglas semánticas, con la finalidad de realizar una integración con un proyecto de Investigación en desarrollo en el Grupo de Investigación de Sistema de Información Inteligentes.

5 Objetivos

5.1 Objetivo General

Modelar semánticamente aspectos de investigación mediante ontologías y extraer información sobre publicaciones a partir de expedientes curriculares en español utilizando técnicas de minería de textos, específicamente aprendizaje automático con reglas semánticas.

5.2 Objetivos Específico

- Diseñar e implementar un modelo de datos semántico basado en ontologías para el almacenamiento y representación de información de productos de investigación.

- Implementar un algoritmo de aprendizaje automático basado en reglas semánticas para la identificación de información relevante sobre los productos de investigación en los expedientes.

- Implementar una aplicación que integre el proceso de extracción de la información relevante.

- Diseñar e Implementar un módulo para la visualización de la información extraída de manera que un usuario toma la decisión final para su representación en el modelo semántico.

6 Marco Teórico

Actualmente casi toda la información se maneja por computadora también esta se almacena y se tiene la necesidad de recuperarla, para las organizaciones esto es de gran utilidad ya que es una mejor forma de gestionar y compartir toda esta información ayudando a reducir costos.

La mayoría de organizaciones piden los expedientes curriculares vía internet y no existe una estructura para elaborarlos esto quiere decir no que cada persona escribe como creen ellos que es mejor hacerlo, se le llama que están escritos en lenguaje humano por lo tanto implica que para poder encontrar información deseada puede ser muy tardado y tedioso así que es necesario de apoyarnos de la computadora para extraer y transformar la información deseada.

Como respuesta a esta problemática surge lo que conocemos como el procesamiento de lenguaje natural.

6.1 Lenguaje Natural

El lenguaje natural es el medio primario por el cual los humanos se comunican entre ellos, haciendo preguntas, expresando creencias, deseos, actitudes, así como reportar eventos, estados y acciones

El Procesamiento del Lenguaje Natural consiste típicamente en una aplicación secuencial de diferentes componentes de análisis como se representa en la Figura 1. [6].

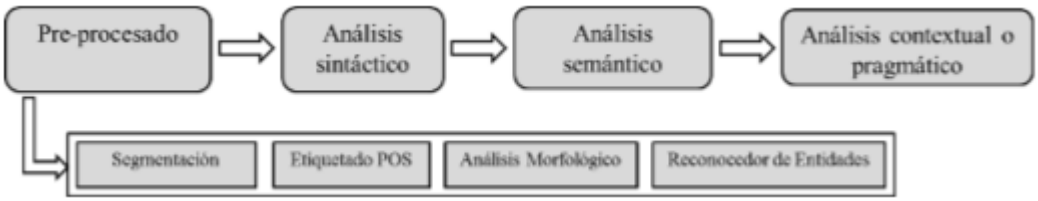

Figura 1. Componentes de Análisis de PLN

1 Figura 1

El lenguaje natural es el lenguaje que hablamos todos los días, nuestra forma de comunicarnos por excelencia y el procesamiento del lenguaje natural, busca precisamente permitir que los ordenadores sean capaces de captar la información transmitida por una persona y trasladarla luego a otra, interactuando con ella con su mismo lenguaje, o al menos, de un modo inteligible.

Sin embargo el procesamiento de lenguaje natural plantea muchos problemas: los múltiples significados de cada palabra, los acentos de cada zona, la jerga de cada lugar, expresiones típicas, lenguaje ambiguo, ironías...etc.

La aplicación del procesamiento de lenguaje natural más obvio y quizá más importante en el momento actual es la búsqueda de información, extracción de información y recuperación de información

6.2 Extracción de Información

La extracción de información se ocupa de estructurar información contenida en textos que son relevantes para el estudio de un dominio (o escenario) particular, llamado dominio de extracción. En otras palabras, el objetivo de un sistema de extracción de información es encontrar y enlazar la información relevante mientras ignora la extraña e irrelevante, cabe destacar que algunos autores consideran a la extracción de información como una etapa posterior de la recuperación de información [7].

Podemos entender como recuperación de información como el proceso de comunicación entre los usuarios y los sistemas, que mediante un término se logra recuperar un conjunto de elementos, que son evaluados por el interesado en la búsqueda, para lograr satisfacer su necesidad de información. [8], para poder acceder a la información necesita estar almacenada.

Una de las mayores ventajas del uso de la computadora es la posibilidad de almacenar información y hay muchas formas de hacer el almacenamiento una de ella es mediante ontologías utilizando el lenguaje *OWL*.[2]

[2] OWL es el acrónimo del inglés Web Ontology Language, un lenguaje de marcado para publicar y compartir datos usando ontologías en la WWW. OWL tiene como objetivo facilitar un modelo de marcado construido sobre RDF y codificado en XML.

6.3 Representación Con Ontologías

Existen muchas definiciones para las ontologías vamos a ver algunas de ellas:

"Una ontología define los términos básicos ya las relaciones que comprenden el vocabulario de una a rea o tópico, así como las reglas para combinar los términos y las relaciones que definen las extensiones al vocabulario." [9].

"Una ontología es una base de datos donde se describen conceptos del mundo o algún dominio en específico, sus propiedades y como se relacionan los conceptos entre sí" [10],

Las ontologías nos van a ayudar a definir conceptos, relaciones, clases, instancias las cuales van son indispensables para poder poblar la ontología.

Por lo antes mencionado ha surgido la necesidad de poder hacer sistemas eficientes y capaces de poder brindar apoyo para las necesidades del usuario. Tales como la identificación de información deseada seguido de su almacenamiento óptimo y su rápida recuperación.

7 Desarrollo del proyecto

En este proyecto se diseñó e implemento un sistema que extrae la información sobre publicaciones como el título año y los autores de expedientes curriculares este consta de 6 módulos bien definidos y relacionados entre sí como se muestra en la Figura2.

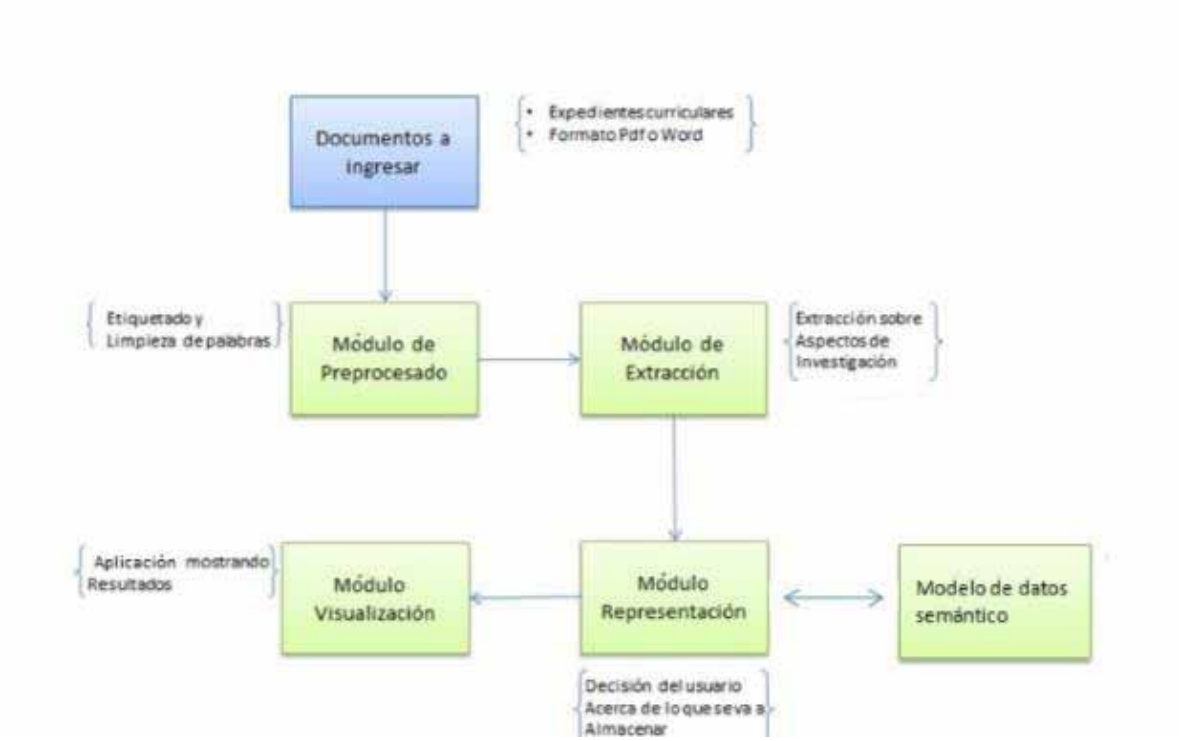

Figura 2. Estructura del Sistema

2 Figura 2

Se observa que cada módulo depende del otro y el conjunto de documentos se encuentran almacenados en algún directorio de la computadora y se van a procesar después de todo el proceso la información extraída se va a almacenar en una ontología.

7.1 Módulo de Preprocesado

En este módulo como se muestra en la Figura 3 se realizó una segmentación de nuestros documentos para después llevar a cabo un análisis morfológico[3] de las palabras clave, esto es clasificar en categoría. Asimismo se llevó a cabo una limpieza para quitar palabras irrelevantes, se generaran cadenas de palabras clave pre-procesadas.

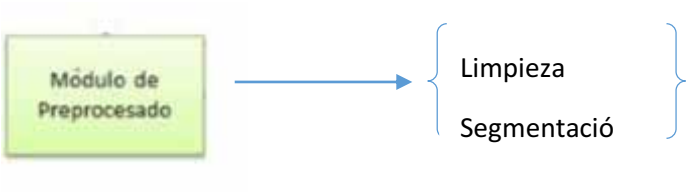

Figura 3. Módulo de Preprocesado

3 Figura 3

7.1.1 Limpieza

Para nuestros documentos en PDF la primera tarea es pasarlos a formato TXT para poder llevar a cabo la limpieza de palabras. En la limpieza se tiene que quitar las palabras vacías, es el nombre que reciben las palabras sin significado como artículos, pronombre, preposiciones, etc. Como se muestra en la Figura 3.1

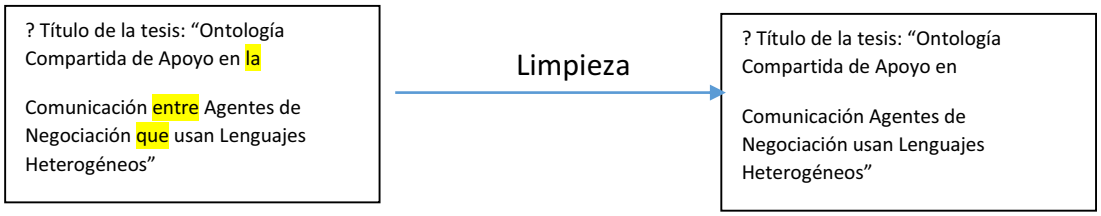

Figura 3.1 Limpieza de Texto

4 Figura 3.1

[3] Consiste en determinar la forma, clase o categoría gramatical de cada palabra de una oración

Para llevar a cabo la limpieza de palabras primero se lee un archivo TXT donde este contiene todas las palabras vacías que deseamos quitar del PDF, ya leídas y almacenadas en un arreglo las palabras estas se sustituyen en el archivo por espacios, para lo cual utilizamos la función en java llamada replace () y replaceAll () como se muestra en el Anexo A.

7.1.2 Segmentación

Para la segmentación del documento se tiene que extraer la parte que sea sobre publicaciones que se han realizado hasta la siguiente sección como se muestra en la Figura 3.2.

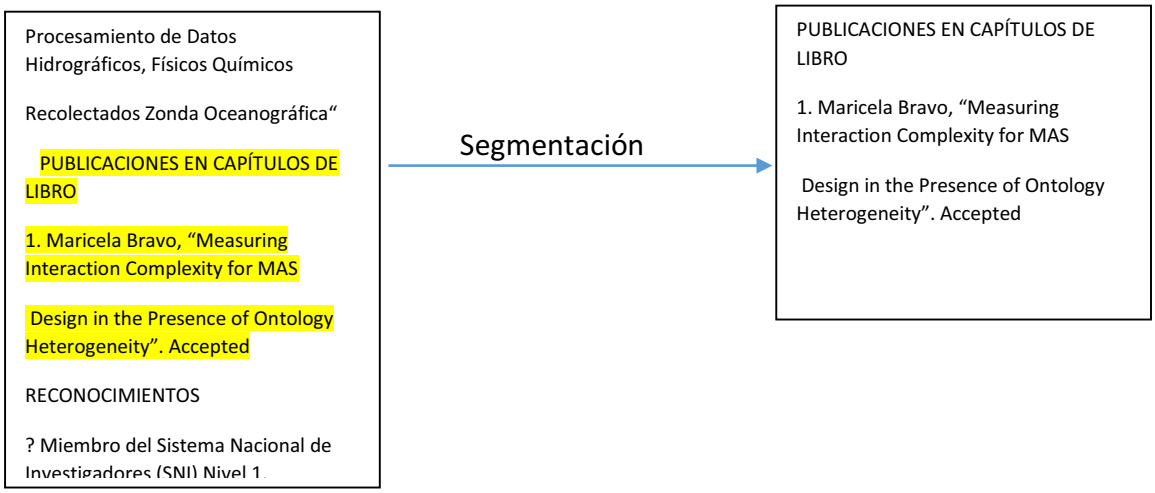

Figura 3.2 Segmentación

5 Figura 3.2

Para poder extraerlo se tuvieron que hacer unas reglas donde se encuentre la palabras publicaciones, publicados, trabajos de investigación o artículos de investigación y a partir de ahí extraer toda la información hasta siguiente sección como se muestra en el Anexo B.

7.1.3 Etiquetado

Después de haber hecho la limpieza y segmentación del documento ahora se hace un etiquetado de palabras para saber de qué tipo son si nombre propios o cardinales, etc. Como se ve en la Figura 3.3

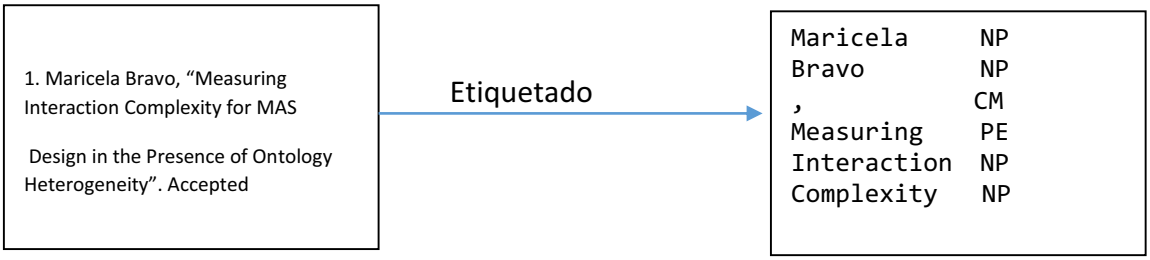

Figura 3.3 Etiquetado de Palabras

6 Figura 3.3

El etiquetado que se realiza nos va a ayudar en el siguiente modulo para poder identificar algunas palabras claves. Para realizar el etiquetado se utilizó una librería llamada TreeTagger y una función la cual se muestra en el Anexo C.

7.2 Módulo de Extracción

Este módulo recibe los datos del módulo anterior y éste permite realizar la extracción sobre aspectos de investigación en los expedientes curriculares utilizando técnicas de aprendizaje automático basado en reglas semánticas. Su salida permitirá al siguiente módulo tener la representación de esta información (Figura 4.).

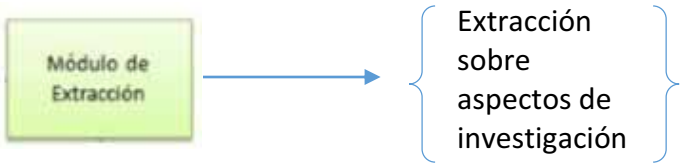

Figura 4. Extracción

7 Figura 4

7.2.1 Extracción

Después de haber obtenido la parte del documento que nos interesa hay que obtener el nombre el año y los autores de las publicaciones como se muestra en la Figura 4.1

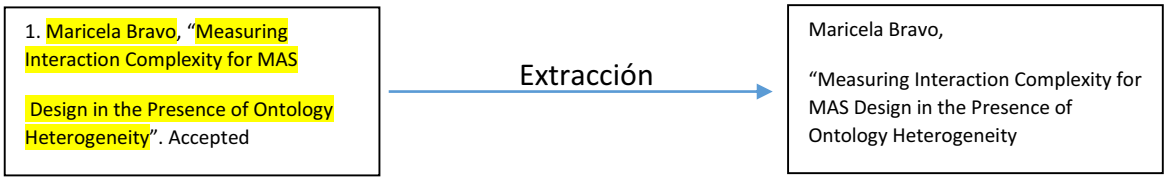

Figura 4.1 Extracción

8 Figura 4.1

Para poder llevar acabo la extracción se tuvo que hacer algunas reglas semánticas como para poder extraer el año debe coincidir que es un cardinal de 4 dígitos, para los autores se tiene que debe de ser un NP seguido de un NP tal como se muestra en el Anexo D.

7.3 Módulo de Representación.

En este módulo el usuario toma la decisión final para su representación en el modelo semántico, aquí ya podremos observar la información que ya ha sido extraída y esto permitirá que solo la información deseada sea almacenada (Figura 5.).

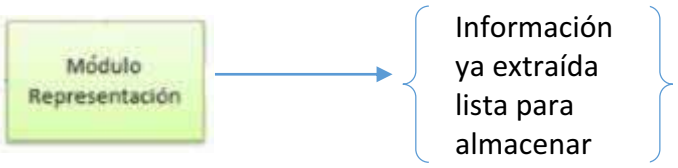

Figura 5. Representación

9 Figura 5

7.3.1 Representación

En este módulo al ya tener la extracción ya podemos observar la información a guardar en modelo de datos semánticos como se muestra en la Figura 5.1

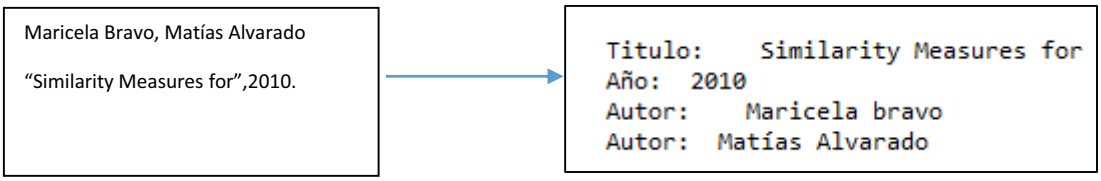

Figura 5.1 Representación

10 Figura 5.1

Con la representación es más fácil identificar lo que se va a almacenar en el modelo de datos semánticos.

7.4 Módulo de Datos Semánticos

Se diseñó e implemento un modelo de datos semánticos basado en ontologías el cual va a almacenar la información relevante extraída. Este tendrá comunicación con el módulo de representación ya que solo se almacenará la información que el usuario haya seleccionado y así podremos evitar información irrelevante o no deseada (Figura 6.).

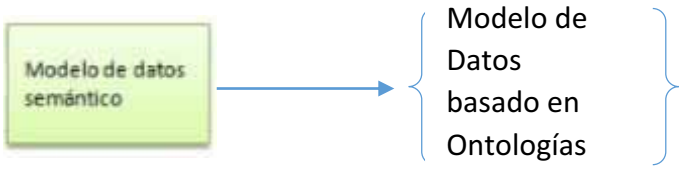

Figura 6 Modelo de Datos Semánticos

11 *Figura 6*

7.4.1 Ontologías

En la ontología se va a almacenar los datos de los demás módulos .Esta está formada por dos clases principales (Figura 6.1) que son la de autor y publicaciones en ellas se almacenan la información extraída

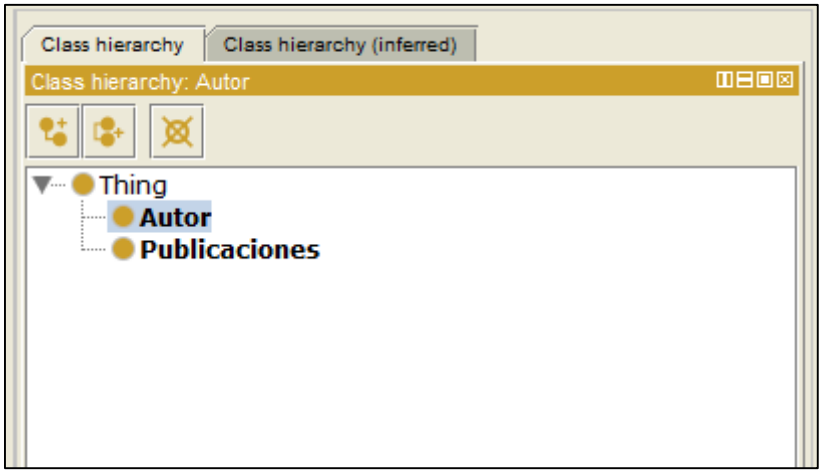

Figura 6.1 Ontología

12 Figura 6.1

Los individuos tienen una relación entre clases llamada “pertenece _a” la cual hace referencia a que una publicación pertenece a uno o varios autores además de que la clase tiene su propiedad “tieneNombre” que hace referencia el nombre del autor y la clase publicación tiene sus propias propiedades como la de “tieneTitulo” y “tieneAño“ Figura 6.2

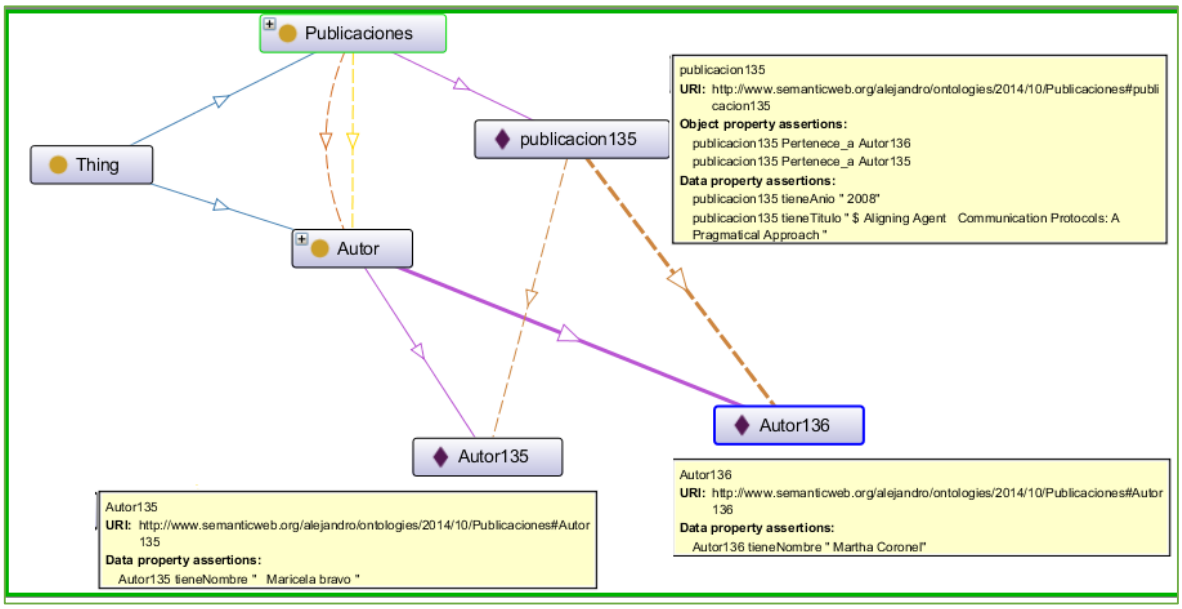

Figura 6.2 Elementos de Ontología

13 Figura 6.2

Para el almacenamiento en la ontología se utilizó la API OWL_API la cual ayuda a poder tener la conexión entre el programa y la ontología como se muestra en el Anexo E.

7.5 Modelo de Visualización

En este módulo se puede visualizar la información respecto a investigaciones de los expedientes curriculares ingresados Figura 7.

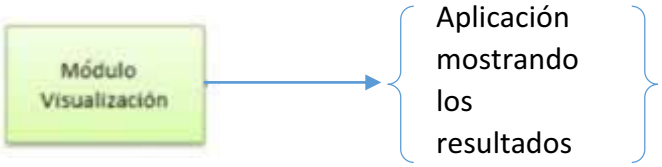

Figura 7. Visualización

14 Figura 7

7.5.1 Visualización

La visualización muestra los resultados obtenidos de la aplicación tanto la extracción como la conexión con la ontología Figura 7.1

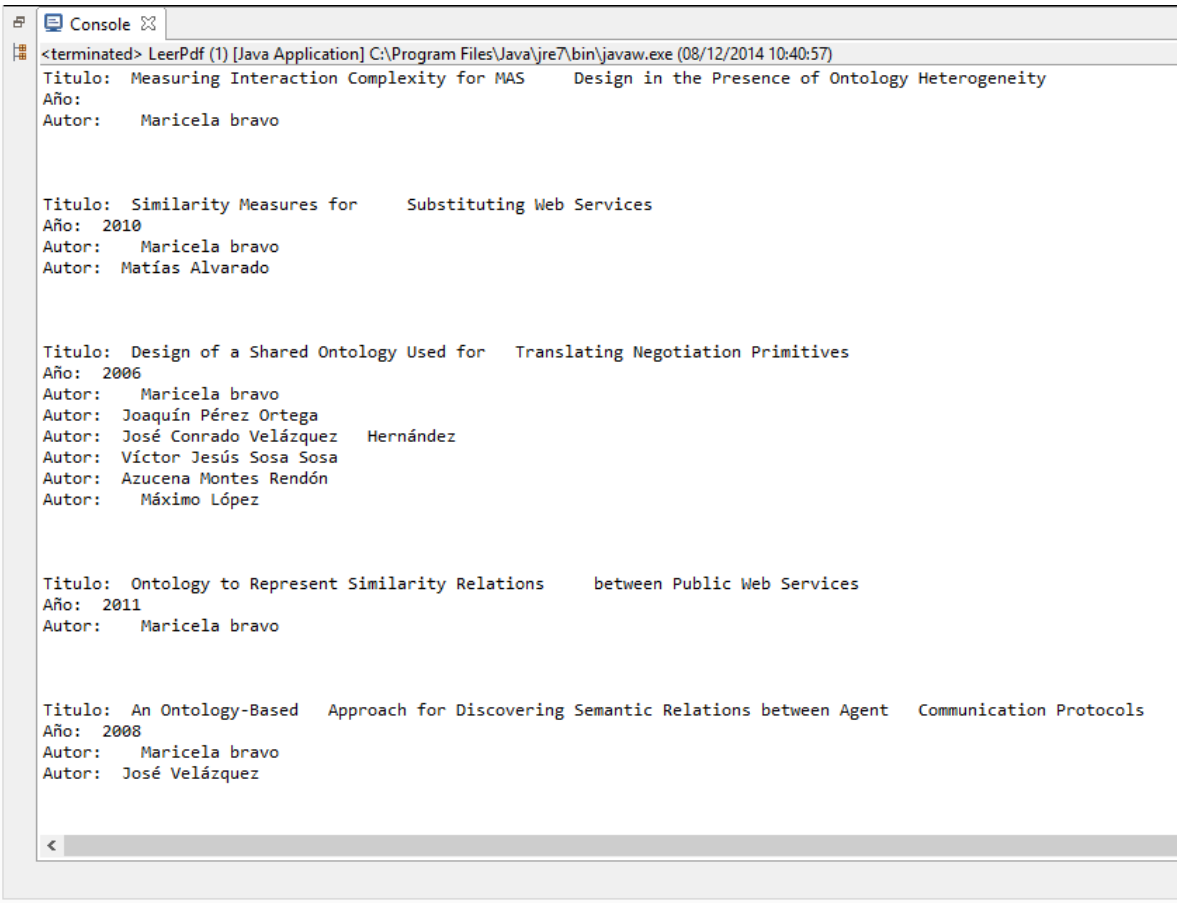

Figura 7.1 Visualización

15 Figura 7.1

8 Resultados

Nuestro sistema consta de algunas carpetas, alguna con todos los PDFS a extraer información también debemos de tener la ontología, palabras vacías a quitar de los documentos, librerías, recursos y una carpeta para los PDF en TXT (Figura 8)

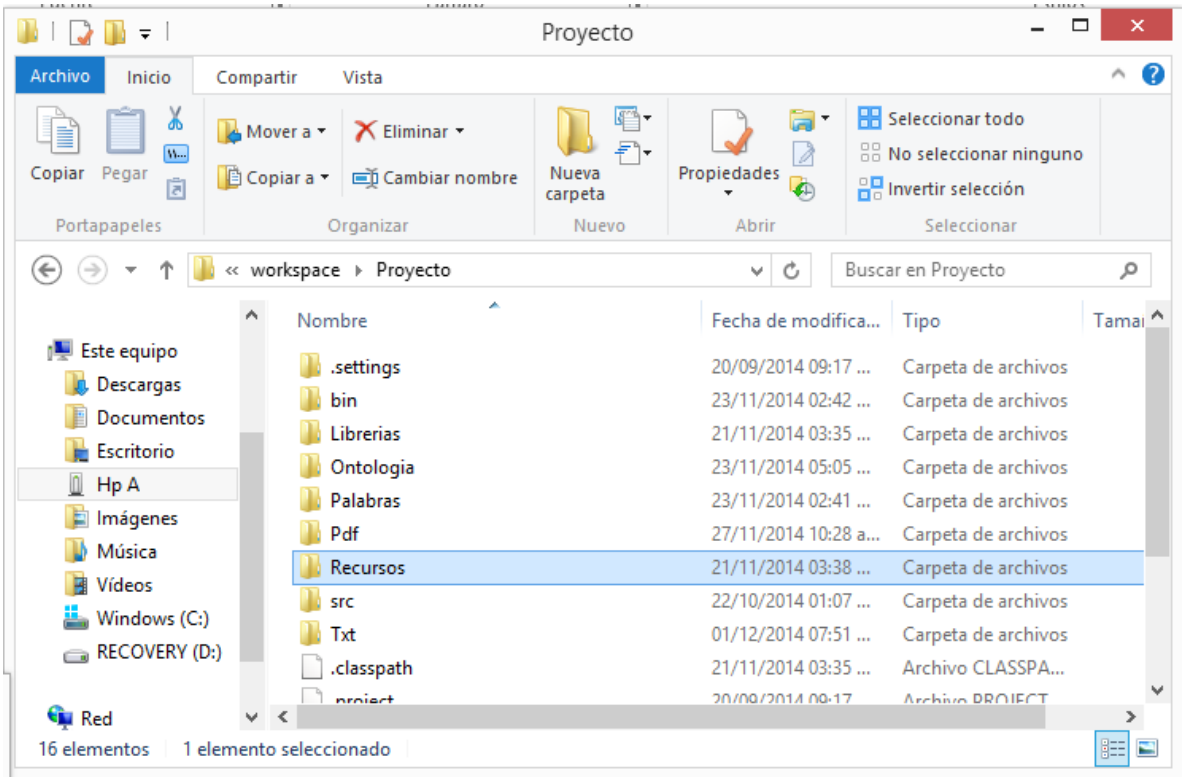

Figura 8. Carpetas

16 Figura 8

Después de haber corrido nuestra aplicación el módulo de visualización nos muestra los resultados Figura 9.La cual muestra el título autores y año sobre las publicaciones que se encontraron en los documentos ingresados a la aplicación.

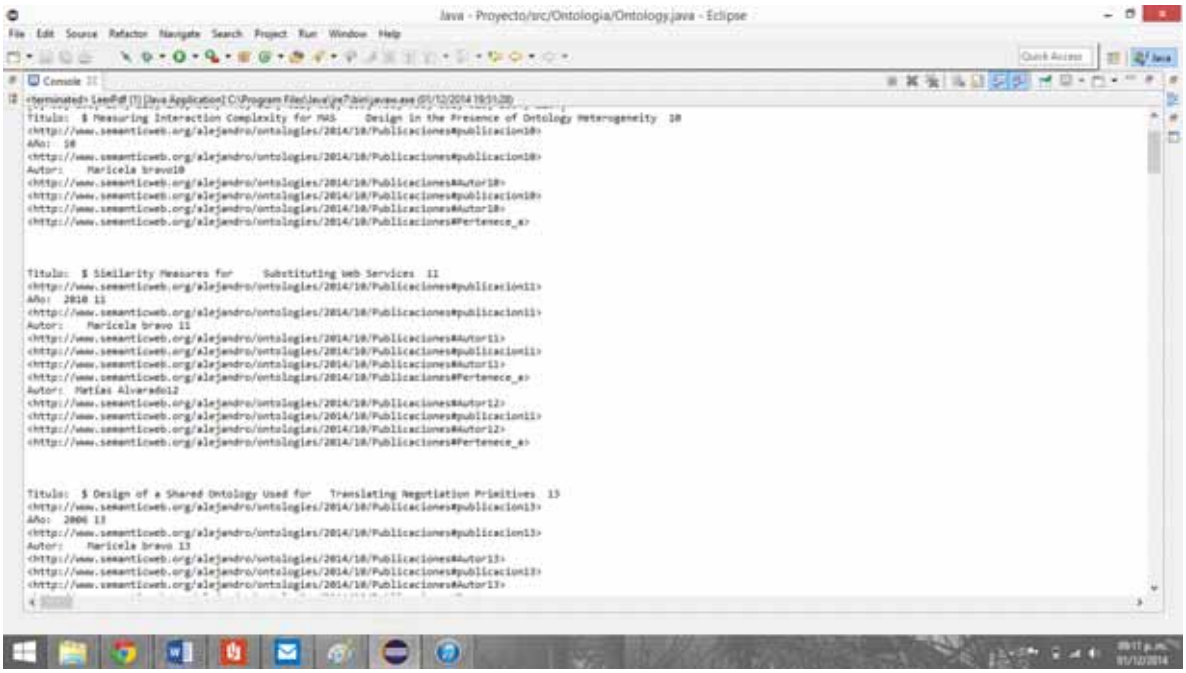

Figura 9 Visualización de la aplicación

17 Figura 9

De igual forma nuestra ontología es modificado de tal manera que almacena la información extraída en la aplicación Figura 10.En ella se muestran las propiedades que tiene las clases y las relaciones que existen entre ellas.

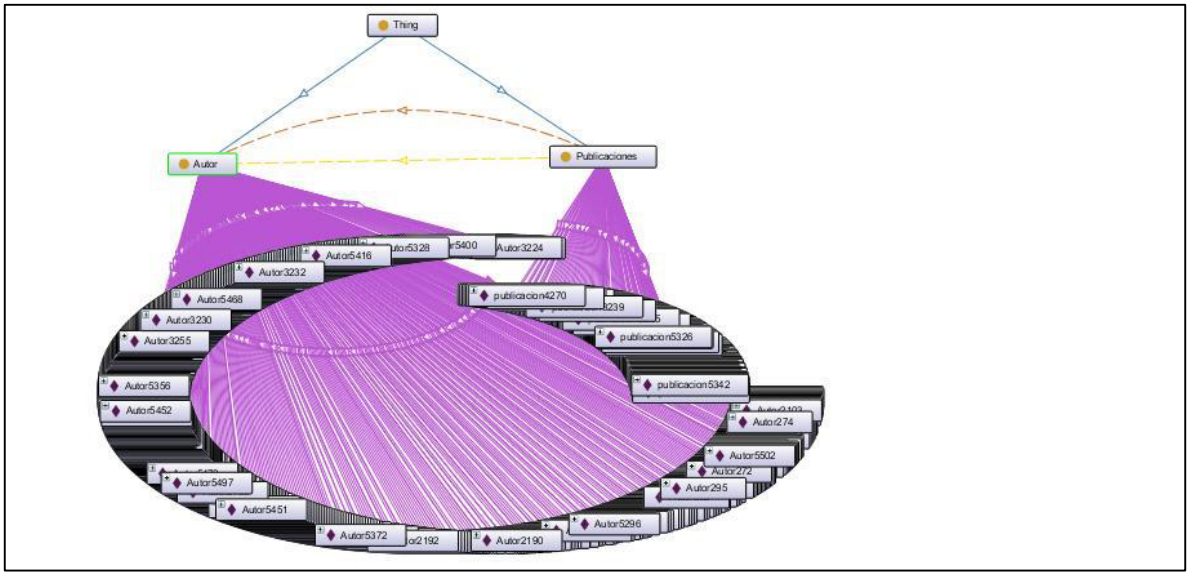

Figura 10 Ontología Completa

18 Figura 10

8.1 Eficiencia

Se probó el sistema con 5 PDFS de profesores de la Universidad Autónoma Metropolitana y los resultados se muestran en la tabla siguiente:

PDF	***Total de publicaciones***	***Extracciones Correctas***	***% de efectividad***
1	***17***	***17***	***100***
2	***39***	***36***	***92.30***
3	***19***	***18***	***94.73***
4	***7***	***7***	***100***
5	***55***	***35***	***63.63***

Tabla 1. Tabla de resultados

19 *Tabla de resultados*

Para el PDF 4 se lograron extraer las 7 publicaciones contenidas en el. Estas son las publicaciones:

```
Título: piña: Supporting a community of co-authors on the web
Año: 2002
Autor: L Morán
Autor: D. Decouchant
Autor: J. Favela
Autor: A. M. Martínez Enríquez
Autor: B. González Beltrán
Autor: S.   Mendoza


Título: tecnología de   información y comunicación apoyar aprendizaje
colaborativo
Año: 2007
Autor:   E. acuñar Garduño
Autor: B. A. González Beltrán
```

Autor: M. Á. Herrera Batista

Título: propuesta metodológica análisis de entorno virtual de aprendizaje colaborativo
Año: 2007
Autor: E. acuñar Garduño
Autor: B. A. González Beltrán

Título: modelo conceptual de sistema de filtrar de información apoyar a comunidad universitario
Año: 2007
Autor: L. gallardo López
Autor: B González Beltrán

Título: A new collaborative filtering model based on robust graph 1 coloring for generation of communities
Año: 2009
Autor: L. gallardo López
Autor: P. Lara Velázquez
Autor: M. Á. Gutiérrez Andrade
Autor: S. G. de Cobos Silva
Autor: B. González Beltrán

Título: caracterización de usuario móvil adaptación de contener web
Año: 2010
Autor: A. granado García
Autor: B. A. González Beltrán

Título: sistema de evaluación dinámico del proceso de enseñanza-aprendizaje
Año: 2011
Autor: Rafaela Blanca Silva-López
Autor: Beatriz Adriana González-Beltrán
Autor: Ángel A. Santos- Palacios

Para el PDF 6 no se lograron extraer el 100% de las publicaciones ya que las reglas que se implementaron no fueron las suficientes para ese expediente curricular. Algunas extracciones mal realizadas son:

Titulo:
Año: 2009
Autor: a. Miguel A. Macias-Garcia

Autor: Víctor J. Sosa-Sosa
Autor: Iván López-Arévalo A generic aproach for data integration using RDF
Autor: OWL
Autor: XML Workshop on Machine Learning
Autor: Data Mining in conjuction with MICAI ISBN: 978-607-95367-1-8 noviembre
Autor: Guanajuato
Autor: México

Título: referencia 558
Año: 2009
Autor: Ocampo-Guzmán
Autor: Lopez-Arevalo
Autor: G. Acosta-Villareal
Autor: V. Sosa- Sosa construcción de ontologías mediante extracción de tema latentes
Autor: Word Net congreso internacional en innovación
Autor: desarrollo tecnológico Octubre
Autor: Cuernavaca
Autor: México

Título: referencia 568
Año: 2009
Autor: F. Pech-May
Autor: Lopez-Arevalo
Autor: Sosa-Sosa Hacia aplicación de guía de Prácticas Clínicas validar congreso internacional en innovación
Autor: desarrollo tecnológico Octubre
Autor: Cuernavaca
Autor: México

Título: referencia 592
Año: 2009
Autor: M. A. Macias-Garcia
Autor: Sosa-Sosa
Autor: Lopez-Arevalo Marco de trabajo extracción
Autor: integración
Autor: consulta de fuente de dato heterogéneo congreso internacional en innovación
Autor: desarrollo tecnológico Octubre
Autor: Cuernavaca
Autor: México

Título: referencia 593
Año: 2009
Autor: M. A. Gomez-Rodriguez
Autor: Sosa-Sosa

Autor: Lopez-Arevalo servicio portable de intercambio automático de archivo dispositivo móvil congreso internacional en innovación
Autor: desarrollo tecnológico Octubre
Autor: Cuernavaca
Autor: México

Título: págs.
Año: 2009
Autor: Ocampo-Guzmán
Autor: Lopez-Arevalo
Autor: E. Tello-Leal
Autor: Sosa-Sosa Hacia construcción automático de ontologías The 7th Brasilina Simposio in información
Autor: Human Language Technology

Título: 54- 58
Año: 2009
Autor: Ocampo-Guzmán
Autor: Lopez-Arevalo
Autor: E. Tello-Leal
Autor: Sosa-Sosa Enfoque probabilístico construcción de ontologías II International Workshop on Web
Autor: Text Inteligencie WTI
Autor: págs.

Título: volumen 5518 2009
Año: 2009
Autor: F. Pech-May
Autor: Lopez-Arevalo
Autor: Sosa-Sosa Validador for clínica practica juilines Lectura Notes in Competer Sáciense

Título: volumen 5518 2009
Año: 2009
Autor: Lopez-Arevalo
Autor: Sosa-Sosa
Autor: E. Tello Leal Abstract models for redesign of technical processes Lecture Notes in Computer Science

9 Conclusiones

Los Expedientes Curriculares 1 y 4 se lograron extraer el 100% de sus publicaciones mientras que en los demás no se logró extraer el total. Esto quiere decir que de un total de 137 publicaciones se lograron extraer correctamente 113 para dar un porcentaje de 82.48% de efectividad del sistemas

El 82.48 % obtenido se debe a que el expediente curricular 6 no se lograron extraer un buen porcentaje de sus publicaciones ya que las reglas implementadas no son las suficientes para poder extraerlas correctamente y no se pueden realizar más reglas ya que esto afectaría los demás expedientes curriculares y solo seria para un caso en específico. Como no existe una regla o un formato para poder escribir los expedientes curriculares esto hace que no se obtenga el 100% de las publicaciones.

Con el sistema implementado es posible extraer artículos y publicaciones de los expedientes utilizando técnicas de procesamiento natural. Esto nos ayuda a poder tener la información almacenada y lista para poder acceder a ella de tal forma que podemos responder preguntas tales como:

¿Quién es el autor o autores de X publicación?

¿De qué año es X publicación?

¿Cómo se llaman las publicaciones hechas por x autor?

¿Cuál es nombre de las publicaciones hechas de X año a Y año?

¿Cuáles son los nombre de los autores con los que x autor ha colaborado?

10 Referencias bibliográficas

[1] Díaz Jiménez Cristina Alicia, "Lenguaje de manipulación y minería de datos", proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2011.

[2] Fernando Alejandro Acosta, "Sistema de procesamiento de textos de investigación", proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2011.

[3] Alina Urquiza Pérez "Sistema Configurable de Minería Web", proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2012.

[4] A. Téllez Valera, "Extracción de información con algoritmos de clasificación", tesis de maestría, Instituto Nacional de Astro física Óptica y Electrónica, México, Puebla, 2005.

[5] James, Luis Gabriel; Vega Riveros, Fernando "Modelos clásicos de recuperación de la información" Revista Integración, vol. 23, núm. 1, 2005, pp. 17-26 Universidad Industrial de Santander Bucaramanga, Colombia.

[6] P. Cimiano. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications, volume 1 of 0-387-30632-3.

[7] J. Cowie and W. Lehnert. Information extraction. Communications of the ACM, 39(1):80-91, 1996.

[8] SALVADOR OLIVÁN, José Antonio y ARQUERO AVILÉS, Rosario. Una aproximación al concepto de recuperación de información en el marco de la ciencia de la documentación. Investigación Bibliotecológica, Vol. 20, Núm. 41, julio/diciembre, 2006, México, ISSN: 0187-358X. pp. 13-43.

[9] Neches et al., 1991.

*[10[*H. Weigand. Multilingual ontology-based lexicon for news filtering. In K. Mahesh, editor, The TREVI Project, pages 138–159, 1997.

11 Anexos

Anexo A:

```
package Limpieza;


import java.io.File;
import java.io.FileWriter;
import java.io.BufferedWriter;
import java.io.IOException;

import org.annolab.tt4j.TreeTaggerException;

import com.itextpdf.text.pdf.PdfReader;
import com.itextpdf.text.pdf.parser.PdfTextExtractor;
public class LeerPdf{
 public static void main(String[] args) throws IOException, TreeTaggerException{

  Limpieza.palabras_vacias();
  selectPDFFiles();
  Etiquetado.Documentos.LeerTxt();


 }




//allow pdf files selection for converting
 public static void selectPDFFiles() throws IOException, TreeTaggerException{



 System.out.println("Procesando Archivos:");
              String sDirectorio = "Pdf";
              String bDirectorio="Txt";
              File f = new File(sDirectorio);

               Borrar(bDirectorio);
      File[] ficheros = f.listFiles();
      for (int x=0;x<ficheros.length;x++){
        System.out.println(ficheros[x].getName());

Text_Limpieza(ficheros[x].toString(),ficheros[x].getName()+".txt",Limpieza.datos
);
      }
```

```
  System.out.println("Terminado");


              }


public static void Borrar (String ruta) {
        File file = new File(ruta);
        String[] contenido = file.list();
        for (int i = 0; i < contenido.length; i++) {
         File fil = new File(ruta + "/" + contenido[i]);
         if (!fil.delete()) {
          fil = new File(ruta + "/" + contenido[i]);
          if (fil.delete()) {
           Borrar(ruta + "/" + contenido[i]);
          } else {
           Borrar(ruta + "/" + contenido[i]);
          }
         }
         if (fil.isDirectory()) {
          fil.delete();
         }
        }
         }


public static void Text_Limpieza(String src,String desc, String[] palabras){
 try{
  //create file writer
        File archivo=new File("Txt\\"+desc);
  //create buffered writer

  BufferedWriter bw= new BufferedWriter(new FileWriter(archivo));
  //create pdf reader
  PdfReader pr=new PdfReader(src);
  //get the number of pages in the document
  int pNum=pr.getNumberOfPages();
  //extract text from each page and write it to the output text file
  for(int page=1;page<=pNum;page++){
   String text=PdfTextExtractor.getTextFromPage(pr, page);
   for(int i = 0; i < palabras.length; i++) {
     text=" "+text;
     text=text.replace(",", " ,");
     text=text.replace("(", " ");
     text=text.replace(")", " ");
     text=text.replace("/", " ");
     text=text.replace(".", " . ");
     text=text.replace('"', '“');
     text=text.replace("“", " ");
     text=text.replace("”", "  ");



     text = text.replaceAll("[\n]","  "); //REEMPLAZA (_PALABRA_)
```

```
    text = text.replaceAll(" "+palabras[i].trim()+" "," "); //REEMPLAZA
(_PALABRA_)




 }



    bw.write(text);
    bw.newLine();


   }


   bw.flush();
   bw.close();



  }
  catch(Exception e){e.printStackTrace();}

 }

}
```

```
package Limpieza;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;

/**
 *
 * @author Alejandro
 */
public class Limpieza {
        static  String[] datos = null;



   public static void palabras_vacias(){
```

```
   File archivo = null;
   FileReader fr = null;
   BufferedReader br = null;

   String linea = null;

   try {
      //Cargamos el archivo de la ruta relativa
      archivo = new File("Palabras\\Palabras.txt");
      //Cargamos el objeto FileReader
      fr = new FileReader(archivo);
      //Creamos un buffer de lectura
      br = new BufferedReader(fr);

       //Leemos hasta que se termine el archivo
      while ((linea = br.readLine()) != null) {


         //Utilizamos el separador para los datos
        datos = linea.split(",");

         //Presentamos los datos


      }

      //Capturamos las posibles excepciones
   } catch (Exception e) {
      e.printStackTrace();
   } finally {
      try {
         if (fr != null) {
            fr.close();
         }
      } catch (Exception e2) {
         e2.printStackTrace();
      }
   }

}
         }
```

Anexo B:

```
package Extraccion;
import java.io.IOException;
import java.util.ArrayList;
import org.annolab.tt4j.TreeTaggerException;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;

import Etiquetado.Etiquetado;


	public class Segmentacion {
		static int x=1;
		 static String s="";

	public static String getS() {
			return s;
		}



		public static void setS(String s) {
			Segmentacion .s= s;
		}



	public static void Expresion(ArrayList<String> Arr,ArrayList<String> ArrL) throws
IOException, TreeTaggerException, OWLOntologyStorageException,
OWLOntologyCreationException {

		System.out.println("\r\n"+"Documento "+x+"\r\n");

		 for( int i = 0 ; i< ArrL.size(); i++){

			if (i+1==Arr.size() ){

			}else {
```

```
                        if(ArrL.get(i).toLowerCase().equals("publicados")
||ArrL.get(i).toLowerCase().equals("publicaciones:")
||ArrL.get(i).toLowerCase().equals("publicación")
||(ArrL.get(i).toLowerCase().equals("trabajo")&& ArrL.get(i-
1).toLowerCase().equals("de")&& ArrL.get(i+2).toLowerCase().equals("investigación"))|| (
ArrL.get(i).toLowerCase().equals("artículo")&&
ArrL.get(i+1).toLowerCase().equals("de")&&
ArrL.get(i+2).toLowerCase().equals("investigación"))){

                            //System.out.println("Aqui"+ArrL.get(i));
                            for( int j = i+1 ; j< Arr.size(); j++){


                                if((ArrL.get(j).toLowerCase().equals("producto")
)||(ArrL.get(j).toLowerCase().equals("proyecto") && Arr.get(j+1).equals("ADJ"))||
(ArrL.get(j).toLowerCase().equals("dirección") &&
ArrL.get(j+2).toLowerCase().equals("tesis:")) ) {
                                    //System.out.println("Aqui2");
                                        // System.out.println(i+" "+j);
                                          for( int y = i ; y<j; y++){
                                              s=s+" "+ArrL.get(y);

                                    }

                                        i=j;
                                            break;
                                    }

                            }

        }
                    }

    }

            System.out.println(s);

            String[] E;
            E=Segmentacion .getS().split(" ");
            Etiquetado.Etiquetar(E);
```

```
                        Extraccion.
Extraer(Etiquetado.getE().get(1),Etiquetado.getE().get(0),x);
                s="";

                x++;
        }

        }
```

Anexo C:

```
package Etiquetado;



import java.io.IOException;
import java.util.ArrayList;

import org.annolab.tt4j.*;

import static java.util.Arrays.asList;

public class Etiquetado {
        public static final String Documentos = null;
        private static ArrayList<ArrayList<String>> palabras = new
ArrayList<ArrayList<String>>();
        private static ArrayList<ArrayList<String>> E = new
ArrayList<ArrayList<String>>();
        private static ArrayList<String> lema_e = new ArrayList<String>();
        private static ArrayList<String> Pos_e = new ArrayList<String>();




        //static int x=0;



        public static ArrayList<ArrayList<String>> getPalabras() {
                return palabras;
        }



        public static void setPalabras(ArrayList<ArrayList<String>> palabras) {
                Etiquetado.palabras = palabras;
        }
```

```
	/**
	 * @param args
	 * @throws IOException
	 * @throws TreeTaggerException
	 */

	public static void Etiquetar(String [] Texto) throws IOException,
TreeTaggerException {


		// Point TT4J to the TreeTagger installation directory. The
executable is expected
		// in the "bin" subdirectory - in this example at
"/opt/treetagger/bin/tree-tagger"
		System.setProperty("treetagger.home", "Recursos/TreeTagger");
		TreeTaggerWrapper<String> tt = new TreeTaggerWrapper<String>();

		//getLema_e().add(new ArrayList<String>());
		//getPos_e().add(new ArrayList<String>());

	getLema_e().clear();
	getPos_e().clear();
		getE().clear();


		try {
			tt.setModel("Recursos/TreeTagger/modelos/spanish.par:iso8859-
1");
			tt.setHandler(new TokenHandler<String>() {
				public void token(String token, String pos, String
lemma) {
				 // System.out.println(token + "\t" + pos + "\t" +
lemma);
					//System.out.println( pos + "\t" + lemma);
				//	get_E().get(x).add();
				  if (lemma.equals("@card@")){
					lemma=token;
				  }

				  if (lemma.equals("@ord@")){
					lemma=token;
				  }



				    getLema_e().add(lemma);
				    getPos_e().add(pos);
				    getE().add( getLema_e());
				    getE().add(getPos_e());
				    //getPalabras().get(0).add(pos + " " + lemma);
```

```
                        }

                });

                //String [] cadenaEntrada={"El", "archipiélago", "de", "Puerto",
"Rico","incluye"};
                tt.process(asList(Texto));

             /// x++;

          }

          finally {
              tt.destroy();

          }

       }



       public static ArrayList<ArrayList<String>> getE() {
             return E;
       }



       public static void setE(ArrayList<ArrayList<String>> e) {
             E = e;
       }



       public static ArrayList<String> getLema_e() {
             return lema_e;
       }



       public static void setLema_e(ArrayList<String> lema_e) {
             Etiquetado.lema_e = lema_e;
       }



       public static ArrayList<String> getPos_e() {
             return Pos_e;
       }



       public static void setPos_e(ArrayList<String> pos_e) {
             Pos_e = pos_e;
       }
             }
```

Anexo D:

```
package Extraccion;

import java.util.ArrayList;

import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;

public class Extraccion {
	static int xa;
	static int xe;
	static int FINAL;
	static String p="";
	static String anio="";
	static String cadena = "";

	static String s="";


	private static ArrayList<String> NUM= new ArrayList<String>();
	public static void Extraer(ArrayList<String> Arr,ArrayList<String> ArrL,int x) throws
OWLOntologyStorageException, OWLOntologyCreationException{

		xa=x;

		 for( int i = 0 ; i< ArrL.size(); i++){

				if (i+1==Arr.size() ){

				}else {
				 if ((ArrL.get(i).equals("%&") &&
ArrL.get(i+1).equals("?"))||(ArrL.get(i).equals("%&") && Arr.get(i+1).equals("CARD") &&
ArrL.get(i+1).length() < 3 && ArrL.get(i+2).equals("."))||(ArrL.get(i).equals("%&") &&
ArrL.get(i+1).equals("%&") && (ArrL.get(i-1).equals(".")||ArrL.get(i-
1).equals("CARD")||ArrL.get(i-1).equals("libro"))) ){

					for( int j = i+1 ; j< Arr.size(); j++){
```

```
                                        if((ArrL.get(j).equals(".")  &&
ArrL.get(j+1).equals("%&"))|| (ArrL.get(j).equals(".")  &&  Arr.get(j-1).equals("CARD")&&
ArrL.get(j-1).length() >3 &&(ArrL.get(j+1).equals("ISBN:")||
ArrL.get(j+1).equals("ISBN")||ArrL.get(j+1).equals("ISSN:")||ArrL.get(j+1).equals("ISSN")) )
){//||(ArrL.get(j).equals(".") &&  Arr.get(j-1).equals("NP")) ) {


                                            cadena = String.valueOf(i);

                                           FINAL=ArrL.size();
                                    NUM.add(cadena);

                                        for( int y = i ; y<j; y++){



                                        }


                                            break;
                                    }

                                }

                    }


                            }
                }

                        // System.out.println("Publicacion: "+p);

                    cadena = String.valueOf(FINAL);
                    NUM.add(cadena);
                    Publicaciones(NUM,ArrL,Arr);
                    NUM.clear();


            }
```

```
public static void Publicaciones (ArrayList<String> indices, ArrayList<String> L,
ArrayList<String> R) throws OWLOntologyStorageException,
OWLOntologyCreationException{
          System.out.println(indices);
         for (int i=0; i<indices.size()-1;i++){



              int inicio=Integer.parseInt(indices.get(i));
              int fin =Integer.parseInt(indices.get(i+1));

              for( int y = inicio ; y<fin; y++){


           s=s+" "+L.get(y);

          }

              datos(s,inicio,fin,L,R);

              inicio=fin;
          //if ()


              s="";

        }


}
public static void datos(String publi, int inicio, int fin, ArrayList<String> l, ArrayList<String>
r) throws OWLOntologyStorageException, OWLOntologyCreationException{
          String spruebas = "";
          int t=0;
          for( int i = inicio ; i< fin; i++){


               if (i+1==fin ){
```

```
                }
                else {
        if (   l.get(i).equals("$")  ) {

                                for( int j = i+1 ; j< fin; j++){

                                if(l.get(j).equals("$") ) {


                                                for( int y = i ; y<j; y++){


                                                        spruebas=spruebas+" "+l.get(y);

                                                }
                                                    spruebas=spruebas.replaceAll("%&", "
");

                                                    spruebas=spruebas.replaceAll("$", " ");


                                                            p=spruebas;

                                                            i=fin;
                                                                spruebas="";
                                                                t=1;

                                                            break;
                                        }
                                }

                        }




 }

 }
```

```
                for( int i = inicio ; i< fin; i++){
                        if (i+1==fin ){

                        }
                        else {
                 if (r.get(i).equals("CARD") && l.get(i).length()==4 || r.get(i).equals("CODE")
&& l.get(i).length()==4 ){
                                        for( int j = i+1 ; j< fin; j++){

                                        if( l.get(j).equals(".")|| l.get(j).equals(",") && r.get(j-
1).equals("CARD") ||  r.get(j-1).equals("CODE") ) {


                                                for( int y = i ; y<j; y++){


                                                    spruebas=spruebas+" "+l.get(y);

                                                    }


                                                String sSubCadena =
spruebas.substring(1,2);
                                                String sSubCadena2 =
spruebas.substring(2,3);



                                            if (((sSubCadena.equals("2") &&
sSubCadena2.equals("0") )||(  sSubCadena.equals("1")&& sSubCadena2.equals("9") ))&&
spruebas.length()==5 ){


                                                     anio=spruebas;
                                                    i=fin;


                                                                spruebas="";

                                                    break;
                                                }
```

```
                                        spruebas="";

                                        }
                                    }

                                }


                            }


                    }

            for( int i = inicio ; i< fin; i++){



                            if (i+1==fin ){

                            }
                            else {
                        if (  l.get(i).equals("%&")  ) {
                                    for( int j = i+1 ; j< fin; j++){

                                    if(( l.get(j).equals(".") && ( r.get(j-1).equals("NP")||
r.get(j-1).equals("ADJ") && l.get(j-2).equals("."))) ||( l.get(j).equals(",") && (
l.get(j+1).equals("$")|| l.get(j+2).equals("$") ) ) ||( l.get(j).equals(".") && (
l.get(j+1).equals("$")|| l.get(j+2).equals("$")) )|| l.get(j).equals("$") ) {


                                                for( int y = i ; y<j; y++){

            if(r.get(y).equals("CARD")|| l.get(y).equals(".") && !r.get(y-1).equals("ALFS")){

                                                                }
                                                                else if
(l.get(y).equals("and")|| l.get(y).equals("y")){

            spruebas=spruebas+""+",";
```

```
                                                    }
                                                    else{

                                                spruebas=spruebas+" "+l.get(y);
                                                    }

                                                 }


                                    spruebas=spruebas.replaceAll("%&", " ");



                                                String[] arrayAutor = spruebas.split(",");



                                                int n=xa;
                                                int m=xe;

                                                spruebas="";
                                                 if (!l.get(j+1).equals("$") ||
!l.get(j+2).equals("$")){

                                                        for( int z = j+1 ; z< fin; z++){

                                                            if(l.get(z).equals(",") ||(
l.get(z).equals(".") && !l.get(z-1).equals("ALFS") )) {


                                                                                for( int y
= j ; y<z; y++){



spruebas=spruebas+" "+l.get(y);

                                                                                    }
                                                                                break;
```

```
                                                    }

                                                                }
                                                                if(t==0){


spruebas=spruebas.replaceAll("%&", " ");

        spruebas=spruebas.replaceAll("$", " ");

        System.out.println("Titulo : "+spruebas+" "+xa+xe);


        Ontologia.Ontology.createClassIndividual("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"Publicaciones", "publicacion"+xa+xe);

        Ontologia.Ontology.createDataPropertyAssertion("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"publicacion"+xa+xe, "tieneTitulo", spruebas);
                                                                }
                                                                else {

System.out.println("Titulo: "+p+" "+xa+xe);

        Ontologia.Ontology.createClassIndividual("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"Publicaciones", "publicacion"+xa+xe);

        Ontologia.Ontology.createDataPropertyAssertion("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"publicacion"+xa+xe, "tieneTitulo", p);

                                                                        t=0;
                                                                }


                                                                 i=fin;
                                                                 j=fin;
```

```
System.out.println("Año: "+anio+" "+xa+xe);

        Ontologia.Ontology.createDataPropertyAssertion("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"publicacion"+xa+xe, "tieneAnio", anio);



                                                                anio="";

                                                            for (int i1 = 0; i1 <
arrayAutor.length; i1++) {

        System.out.println("Autor: "+arrayAutor[i1]+xa+xe);


        Ontologia.Ontology.createClassIndividual("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",  "Autor",
"Autor"+xa+xe);

        Ontologia.Ontology.createDataPropertyAssertion("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"Autor"+xa+xe, "tieneNombre", arrayAutor[i1]);

        Ontologia.Ontology.createObjectPropertyAssertion("Ontologia/Publicaciones.owl",
"http://www.semanticweb.org/alejandro/ontologies/2014/10/Publicaciones#",
"Pertenece_a","publicacion"+n+m, "Autor"+xa+xe);



                                                                    xe++;
                                                                }

        System.out.println();

        System.out.println();

        System.out.println();
                                                                p="";
```

```
                                                    }


                                                     i=fin;
                                                            spruebas="";


                                                    break;


                                                }
                                            }

                                        }


                }

                }


        }
        }
```

Anexo E:

```
package Ontologia;

import java.io.File;
import java.util.Iterator;
import java.util.Set;

import org.semanticweb.owlapi.apibinding.OWLManager;
import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLLiteral;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyIRIMapper;
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.util.AutoIRIMapper;

public class Ontology {


        public static void Mostar(){
        try
    {
        //Primero se carga la ontologia
        String localOntoDir = "Ontologia";
        File dir = new File( localOntoDir );

       OWLOntologyManager man = OWLManager.createOWLOntologyManager();
       OWLOntologyIRIMapper autoIRIMapper = new AutoIRIMapper(dir, false);
       man.addIRIMapper(autoIRIMapper);
```

```
    File ontoFile = new File("Ontologia/Publicaciones.owl");
    IRI ontoIri = IRI.create(ontoFile);
    OWLOntology ontology = man.loadOntologyFromOntologyDocument(ontoIri);

    System.out.println("Ontología cargada en memoria: " + ontology);
    System.out.println("IRI de la ontología: " + ontoIri.toString());



        //Se despliegan las ontologias

      OWLOntology ont = ontology;
      System.out.println( "Ontologia Importada " + ont);
      //Se despliegan las entidades de cada ontología importada
      showInstances(ont);

    System.out.println();
    //showInstances(ont);

  }
  catch (OWLOntologyCreationException e)
  {
    System.out.println("No se pudo cargar la ontología: " + e.getMessage());
  }
}
      public static void showInstances(OWLOntology ontology)
      {
                    Set<OWLClass> classes;
          Set<OWLObjectProperty> objectProperties;
          Set<OWLDataProperty> dataProperties;
          Set<OWLNamedIndividual> individual;

          classes = ontology.getClassesInSignature();
          dataProperties = ontology.getDataPropertiesInSignature();
          objectProperties = ontology.getObjectPropertiesInSignature();
          individual = ontology.getIndividualsInSignature();

          for(Iterator<OWLClass> it=classes.iterator(); it.hasNext();)
    {
      OWLClass clase = (OWLClass)it.next();
```

```
        System.out.println( "Clase " + clase.toStringID());
      }
      System.out.println();

      for(Iterator<OWLObjectProperty> it= objectProperties.iterator(); it.hasNext();)
      {
        OWLObjectProperty op = (OWLObjectProperty)it.next();
        System.out.println("Propiedad de objeto " + op.toStringID());
      }
      System.out.println();

      for(Iterator<OWLDataProperty> it= dataProperties.iterator(); it.hasNext();)
      {
        OWLDataProperty dp = (OWLDataProperty)it.next();
        System.out.println( "Propiedad de dato " + dp.toStringID());
      }
      System.out.println();

      for(Iterator<OWLNamedIndividual> it= individual.iterator(); it.hasNext();)
      {
        OWLNamedIndividual in = (OWLNamedIndividual)it.next();
        System.out.println("Individuo " +  in.toStringID());
      }

       }
       public static void createDataPropertyAssertion(String ontoFile, String ontoIRI,
String objName, String property, String value) throws OWLOntologyStorageException,
OWLOntologyCreationException   {
                        OWLOntologyManager manager =
OWLManager.createOWLOntologyManager();
                        IRI ontologyIRI = IRI.create(ontoIRI);
                        IRI ontoIri = IRI.create(new File(ontoFile));
                        OWLOntology ontology =
manager.loadOntologyFromOntologyDocument(ontoIri);
                        OWLDataFactory factory = manager.getOWLDataFactory();
                        IRI dataPropertyName = generateIRI(property,ontologyIRI);
                        OWLDataProperty owlDatatype =
factory.getOWLDataProperty(dataPropertyName);
                        IRI namedIndividual = generateIRI(objName,ontologyIRI);
                        OWLNamedIndividual individual =
factory.getOWLNamedIndividual(namedIndividual);
```

```
                        System.out.println(individual.toString());
                        OWLLiteral owlLiteral = factory.getOWLStringLiteral(value);
                        OWLDataPropertyAssertionAxiom dataProperty =
factory.getOWLDataPropertyAssertionAxiom(owlDatatype, individual, owlLiteral);
                        manager.addAxiom( ontology, dataProperty);
                        manager.saveOntology(ontology);
                        }
                public static void createObjectPropertyAssertion(String ontoFile, String
ontoIRI, String relation, String obj1Name, String obj2Name) throws
OWLOntologyStorageException, OWLOntologyCreationException   {
                        OWLOntologyManager manager =
OWLManager.createOWLOntologyManager();
                        IRI ontologyIRI = IRI.create(ontoIRI);
                        IRI ontoIri = IRI.create(new File(ontoFile));
                        OWLOntology ontology =
manager.loadOntologyFromOntologyDocument(ontoIri);
                        OWLDataFactory factory = manager.getOWLDataFactory();
                        IRI namedIndividual1 = generateIRI(obj1Name,ontologyIRI);
                        IRI namedIndividual2 = generateIRI(obj2Name,ontologyIRI);
            IRI hasRelationIRI = generateIRI(relation,ontologyIRI);
            OWLNamedIndividual obj1 =
factory.getOWLNamedIndividual(namedIndividual1);
            System.out.println(obj1.toString());
            OWLNamedIndividual obj2 =
factory.getOWLNamedIndividual(namedIndividual2);
            System.out.println(obj2.toString());
            OWLObjectProperty objProperty =
factory.getOWLObjectProperty(hasRelationIRI);
            System.out.println(objProperty.toString());
            OWLObjectPropertyAssertionAxiom propertyAssertion =
factory.getOWLObjectPropertyAssertionAxiom(objProperty, obj1, obj2);
            manager.addAxiom(ontology, propertyAssertion);
            manager.saveOntology(ontology);
                  }

                public static void createClassIndividual(String ontoFile, String ontoIRI, String
clase, String individual) throws OWLOntologyStorageException,
OWLOntologyCreationException {
                  try {
                     OWLOntologyManager manager =
OWLManager.createOWLOntologyManager();
```

```
                IRI ontologyIRI = IRI.create(ontoIRI);
                IRI ontoIri = IRI.create(new File(ontoFile));
                OWLOntology ontology =
manager.loadOntologyFromOntologyDocument(ontoIri);
                OWLDataFactory factory = manager.getOWLDataFactory();
                IRI namedClase = generateIRI(clase, ontologyIRI);
                IRI namedIndividual = generateIRI(individual, ontologyIRI);
                OWLClass clasePadre = factory.getOWLClass(namedClase);
                OWLNamedIndividual ind =
factory.getOWLNamedIndividual(namedIndividual);
                OWLClassAssertionAxiom classAssertion =
factory.getOWLClassAssertionAxiom(clasePadre, ind);
                manager.addAxiom(ontology, classAssertion);
                manager.saveOntology(ontology);
            } catch (OWLOntologyCreationException e) {
                System.out.println("No se pudo cargar la ontología: " + e.getMessage());
            } catch (OWLOntologyStorageException e) {
                e.printStackTrace();
            }
        }

        private static IRI generateIRI(String clase, IRI ontologyIRI) {
          IRI ontologyI =IRI.create(ontologyIRI + clase);
            return ontologyI;
        }
}
```