ARQUITECTURA DE SERVICIOS PARA REPORTES AUTOMATICOS DE TEXTO A PARTIR DE LA WEB

J. Guadalupe Ramos Díaz[a], Edith Amalia Barragán López[a], José Juan Cabeza Ortega[a], Isela Navarro Alatorre[a]

[a]Instituto Tecnológico de La Piedad, La Piedad, Mich., joguar@hotmail.com, ebarragan80@hotmail.com, jjcabexa@yahoo.com.mx, inavarro@pricemining.com.

RESUMEN

Por años, la búsqueda de información relevante a partir de la Web ha seguido un patrón típico. Dicho patrón consiste, regularmente, en el lanzamiento de una consulta, escrita en un navegador Web y que se dirige hacia un servidor de búsquedas: "el buscador", el cual suele devolver una lista de vínculos de Internet llamados URLs (*Uniform Resource Locator*) que el usuario debe leer y discriminar para encontrar la información requerida.

Esta manera de buscar información es usual en computadoras personales con amplios monitores, pero es bastante incómoda en dispositivos pequeños como tabletas y teléfonos inteligentes. Además, este proceso, eminentemente manual, es insuficiente para aprovechar la enorme cantidad de información presente en la Web.

Consideramos que la incorporación de métodos con una base formal sólida, para la discriminación de información y la creación de reportes de texto a partir de páginas web, permitirían encontrar contenido asociado a una solicitud de consulta de manera automática y además ser presentados de forma reducida y adecuada para dispositivos de pantalla pequeña. Dichos métodos deberían estar colocados como servicios en la Web para que puedan ser accedidos por todas las personas desde sus dispositivos.

En este trabajo, se presenta una arquitectura de software que materializa los mencionados métodos, mismos que se disponen en forma de servicios Web. Además, se documentan pruebas efectuadas a los servicios. De esta manera se concluye el diseño y desarrollo para creación de reportes automáticos, paso necesario para la posterior construcción de agentes al servicio de usuarios de dispositivos móviles.

1. INTRODUCCIÓN

El principal almacén de información que se emplea en todo el mundo, hoy en día, es la Web. La pertinencia y disponibilidad de la información le hacen el medio más atractivo. Sin embargo, descubrir la información de nuestro interés, en ese almacén, es aún una actividad que implica una gran interacción entre la persona y el medio. Por ejemplo, cuando se requiere información específica en algún proceso de investigación, el usuario se ve obligado a leer y analizar cada página web que devuelve un buscador. Considerando la cantidad de fuentes devueltas por el buscador la tarea se vuelve tediosa y compleja.

Resulta atractivo considerar programas de software como herramientas para hacer la discriminación automática de información. Para crear tales herramientas, es conveniente la definición de métodos formales que permitan su construcción correcta y confiable.

En [1] se introdujo un método formal para encontrar los fragmentos de texto de una página web que tienen mayor relación con una consulta de usuario. De esta manera a partir de una consulta de web expresada en lenguaje natural es posible obtener aquellos fragmentos de texto más "parecidos" a la consulta y con ello, se puede confeccionar un resumen que servirá para proveer una respuesta al usuario. Para determinar fragmentos "parecidos" es necesario manipular el texto contenido en las páginas web desde la perspectiva de procesamiento de lenguaje natural.

Así pues, en el presente trabajo se da evidencia de una herramienta de software para la creación automática de reportes a partir de páginas web colocada en una arquitectura de servicios web y que permitirá su consulta desde dispositivos móviles. Los aspectos que se atendieron durante el desarrollo del proyecto fueron:

- Definición de un método formal para la determinación de fragmentos relevantes [1].
- Construcción de una librería de procesamiento de lenguaje natural adaptada a páginas web.
- Configuración y uso de interfaces de servidores de búsqueda global (*Google API Custom Search*).
- Determinación de arquitectura de servidor conveniente para la disposición de servicios: *REST web services*.

Enseguida, en la sección 2, se hace referencia al método formal subyacente, definido en [1]. En la sección 3, se hace referencia a aspectos de diseño de la herramienta y finalmente, en la sección 4 se concluye.

2. TEORÍA

La mayoría de páginas web están escritas en lenguaje natural, por ello, se considera que las técnicas de procesamiento de lenguaje natural son apropiadas para llevar a cabo la obtención de información. Se podrían considerar métodos por ejemplo de clasificación automática de texto, los cuales serían aplicables a un documento completo, pero, dado que el objetivo es discriminar (recortar) información, es necesario un método que pueda operar con fragmentos pequeños, por ejemplo, empleando similitud de texto.

Una representación formal estandarizada para documentos de texto escritos en lenguaje natural, se introdujo en Salton et al. [2], donde se define el modelo de espacios vectoriales para indexado automático. Esta representación se emplea cuando las entidades almacenadas, es decir los documentos, se comparan unos con otros.

Si tenemos un documento de texto *d*, un diccionario de términos es un conjunto cuyos elementos son las diferentes palabras contenidas en el documento *d*. Se asocia un vector al documento de texto *d*, y sus componentes son los factores de peso asociados a cada elemento del diccionario. En el método definido, se asume que los factores de peso para cada elemento emplean el esquema *tf*, el cual establece que el valor de una componente en particular se determina por el número de veces que la palabra correspondiente aparece en el documento *d*. El conjunto de documentos en una colección se podría ver como un conjunto de vectores en un espacio vectorial, en el cual hay un eje para cada término. Esta representación pierde el orden relativo de los

términos en cada documento [3]. A este modelo de documento se le conoce en la literatura como el modelo *bolsa de palabras*. En ella, no es importante el orden de los términos, solo el número de veces que aparecen en el documento.

Ejemplo 1: Consideremos el texto=”autos, autos eléctricos, autos de combustión”, el diccionario correspondiente será {autos, eléctricos, combustión} y el vector asociado al diccionario V(diccionario) es <3,1,1>.

La técnica considera algunas palabras comunes como no determinantes a la hora de diferenciar el significado de un documento de texto y por ello no se toman en cuenta (e.g., la, las, los, de, etc.). A estas palabras se le conoce como *stop words*.

De esta manera, el proceso para manipulación de documentos escritos en lenguaje natural sigue los siguientes pasos:

1. Filtrado: Consiste en eliminar todos los símbolos tipográficos como la coma, punto y coma, dos puntos, guión, etc.
2. Eliminar *stop words*: Se refiere a eliminar los términos que no son determinantes para el significado del documento de texto.
3. Crear *la bolsa de palabras*: Básicamente es construir el *diccionario de términos* del documento.
4. Representación vectorial: En base a los términos del diccionario, se elabora un vector para cada documento.
5. En el caso de documentos en la Web es necesario un paso donde se quitan los niveles de formato [4].

El objetivo de la técnica es calcular la relevancia de los fragmentos de texto de una página en función de su similitud con respecto a una consulta web emitida por un usuario. De este modo, una consulta web de usuario es una expresión en lenguaje natural, por ejemplo: ”web texto lenguaje natural” que normalmente se lanza hacia un buscador web.

La *similitud* es una medida estándar entre fragmentos de texto escritos en lenguaje natural. La forma general de cuantificarla es mediante la similitud de coseno de sus representaciones vectoriales. Por ejemplo si *t1* y *t2* son dos bloques de texto, entonces su similitud viene dada por el producto punto de los vectores V(*t1*) y V(*t2*), dividido entre el producto de las longitudes euclidianas de V(*t1*) y V(*t2*). Formalmente:

$$sim(t1, t2) = \frac{\overrightarrow{V}(t1) \cdot \overrightarrow{V}(t2)}{|\overrightarrow{V}(t1)||\overrightarrow{V}(t2)|}$$

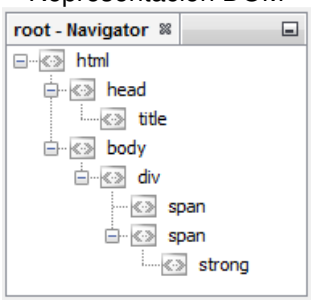
El método tomará primero una consulta de usuario y entonces computará varios tests de similitud contra los bloques que componen una página web. Para obtener bloques de texto la técnica se apoya en la estructura implícita del árbol DOM [5] de la página web. Se aplica la prueba de similitud a cada uno de los bloques del árbol DOM, determinando de ese modo su relevancia.

Por ejemplo:

(X)HTML

```
<html>
  <head>
    <title> Búsqueda y recuperación de información de la Web </title>
  </head>
  <body>
    Recuperación inteligente de información
    de la Web.
    <div>
      Existen dos aproximaciones:
      <span>        a) Métodos de la Web Semántica.      </span>
      <span>        b) Procesamiento de texto en
                  <strong> Lenguaje Natural. </strong>
      </span>
    </div>
  </body>
</html>
```

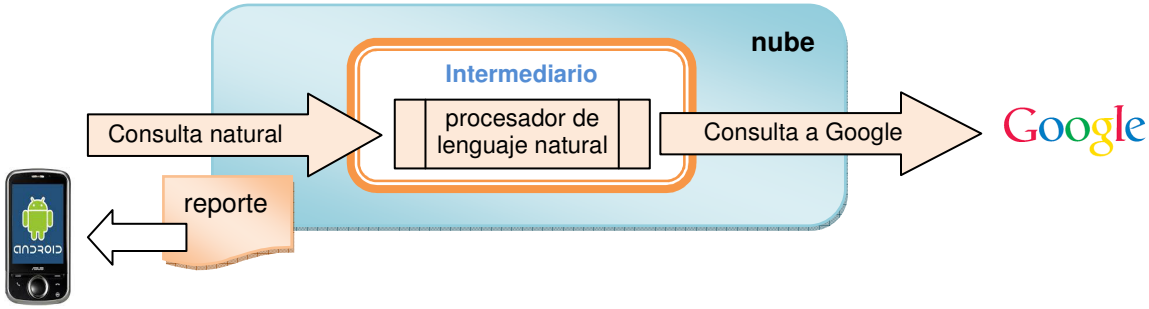
Representación DOM



Arriba, a la izquierda se observa el código de la página. Mientras que a la derecha se muestra gráficamente su árbol DOM. La técnica toma cada uno de los fragmentos de página (un nodo del árbol DOM) y computa la similitud con respecto a la consulta que un usuario haya emitido. El nodo que presente mayor similitud en el cálculo se toma para integrar el reporte que es entregado al usuario.

3. PARTE EXPERIMENTAL

El objeto del presente trabajo es presentar la herramienta como tal, considerando los aspectos de diseño relevantes y que se materializan en la arquitectura y el propio software. El esquema que sigue ilustra la arquitectura:



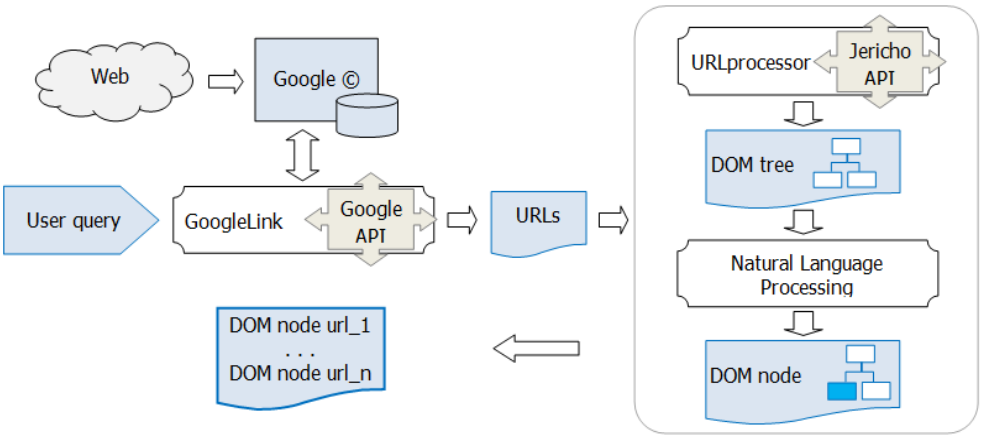
Los elementos que la componen son:

Cliente: El cliente está constituido por un esquema que permite lanzar consultas al intermediario mediante invocaciones a servicios web. Dichas invocaciones pueden llevarse a cabo desde cualquier aplicación cliente, por ejemplo desde aplicaciones de escritorio, pero idealmente desde dispositivos móviles.

Intermediario: El intermediario es un módulo de software localizado en un servidor web que materializa el procesamiento de lenguaje natural, las llamadas a servidores de búsqueda globales y, a partir de la información recibida la manipulación de páginas web y la confección del reporte.

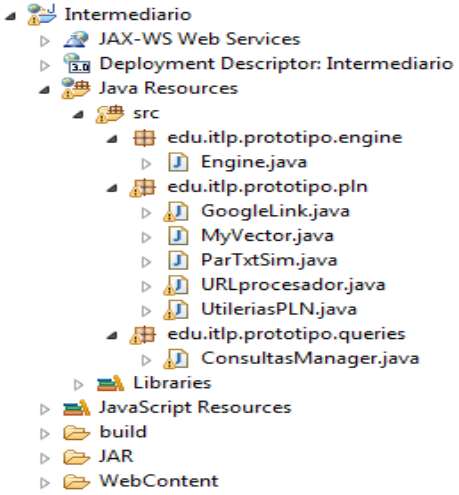
En términos de diseño diremos que se trata de una aplicación multicapa. Cuyas reglas del negocio residirán en la nube, estarán materializadas en un conjunto de *web services* disponibles para invocación desde los clientes. Los clientes serán básicamente apps para dispositivos móviles que interactuarán directamente con el usuario.

Enseguida se esquematiza el proceso interno del Intermediario.



Inicialmente, se recibe la consulta del usuario (*User query*) que fue enviada desde un cliente. Enseguida, la consulta del usuario se lanza a la API de Google, quien nos devuelve un conjunto de enlaces (URLs). Los enlaces son cargados localmente empleando Jericho API (un HTML Parser), de manera tal que de cada página podemos obtener su árbol DOM. A cada nodo DOM se le aplica una prueba de similitud reportada en [1] que finalmente nos arroja el nodo más importante de cada página. A partir de los nodos relevantes de cada página, se confecciona el reporte, el cual es devuelto al usuario emisor de la consulta para visualizarse.

Para la construcción del intermediario se creó un proyecto del tipo “Dynamic Web Project” en el IDE Eclipse compuesto por tres paquetes: edu.itlp.prototipo.engine, edu.itlp.prototipo.pln y finalmente edu.itlp.protoipo.queries.

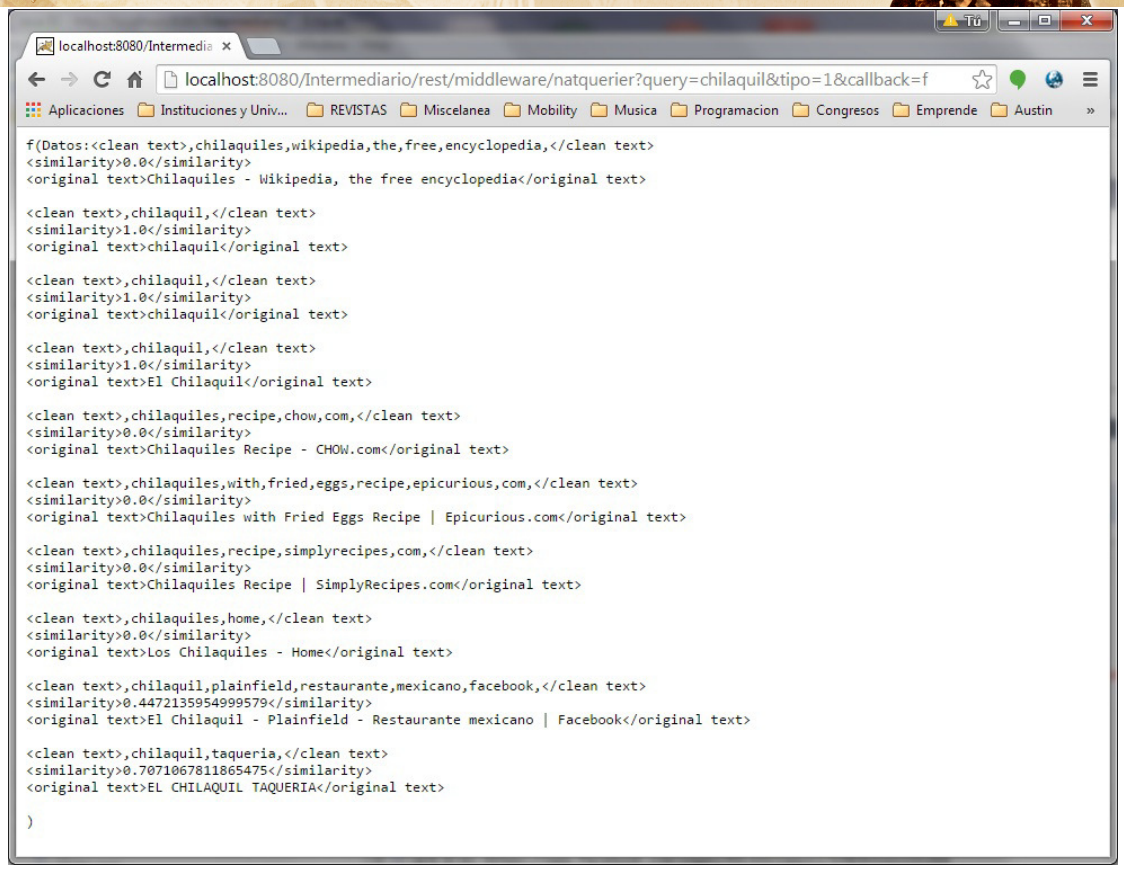


A continuación se describe el contenido de cada paquete:

- edu.itlp.prototipo.engine,
  En este paquete se encuentra un motor de consultas semánticas, particularmente hace uso de JENA para recuperar archivos RDF y extraer conocimiento, este desarrollo forma parte de una extensión semántica en proceso.
- edu.itlp.prototipo.pln
  En este paquete se incluye un conjunto de clases que llevan a cabo el procesamiento de lenguaje natural, por ejemplo eliminación de stop words, limpieza de texto (de símbolos), etc. Estas clases han sido elaboradas como parte del desarrollo del proyecto, no se utilizan librerías de terceros.
  Contiene también, las clases para hacer consultas a servidores de búsqueda global, recibir URLs e invocar la descarga y manipulación de páginas web.
- edu.itlp.prototipo.queries.
  En este paquete está la clase mínima con la definición de servicios web.

Como resultado de la compilación del conjunto de paquetes se obtiene un archivo WAR (ejecutable web) que se coloca en el directorio raíz del Servidor Web Apache Tomcat 7.0.4, para su invocación por los clientes.

La siguiente es una impresión de pantalla que ilustra la funcionalidad de consulta en lenguaje natural



En la impresión se observa la invocación al servicio web de la siguiente manera:

```
http://localhost:8080/Intermediario/rest/middleware/natquerier?query=user
+query&tipo=1&callback=f
```

Donde el parámetro *query* lleva la cadena de búsqueda lanzada por el usuario.

4. CONCLUSIONES

En el presente trabajo hemos mostrado resultados de un proyecto que tiene como fin confeccionar reportes automáticos de texto para presentarse en dispositivos con interfaces simples y pequeñas. Hasta el punto actual hemos desarrollado una infraestructura de servidor para colocarse en la nube y que pone a disposición servicios web para ser accedidos de manera remota. Los experimentos se han hecho de manera local y su ascenso a la nube requiere sólo la selección de un proveedor.

La invocación al servicio web para la demanda del reporte de texto automático encierra un conjunto de operaciones encapsuladas en la aplicación de servidor. Algunas de las operaciones implican: a) preparación, configuración e invocación a servidores de búsqueda global de manera programática. b) Carga de páginas web y trato de las mismas mediante técnicas de procesamiento de lenguaje

natural y c) Confección de reporte de texto automático mediante el cálculo de relevancia de fragmentos de texto.

Consideramos que la técnica se encuentra en una etapa de madurez y por tanto el paso siguiente es la definición de interfaces móviles para explotar la infraestructura aquí presentada, así como su disposición pública en la nube para el empleo de usuarios comunes.

El ejercicio del proyecto en su conjunto demuestra la trascendencia del uso de métodos formales aplicados a problemas cotidianos que pueden eventualmente cambiar la forma en la que las personas, en este caso, consultan información.

BIBLIOGRAFÍA

1. J.G. Ramos Díaz, J.C. Solorio Leyva, L.A. Oceguera Calderón, I. Navarro Alatorre, “Cálculo de relevancia de fragmentos de texto de páginas web con base en la consulta”, *9o Congreso Estatal de Ciencia, Tecnología e Innovación, CECTI*, ISSN: 2007-8617, 2014.
2. G. Salton, A. Wong, y C.S. Yang. “A Vector Space Model for Automatic Indexing.” Commun. ACM, 18(11):613–620, 1975.
3. Ch.D. Manning, P. Raghavan, y H. Schütze. “An Introduction to Information Retrieval”. Cambridge University Press, 2008.
4. X. Qi y B.D. Davison. “Web Page Classification: Features and Algorithms”. ACM Comput. Surv., 41(2), 2009.
5. W3C Architecture Domain. Document Object Model (DOM). ”Available athttp://www.w3.org/DOM/”, 2015.