Capítulo 3. Diseño del Sistema

3.1 Algoritmo para la generación automática del extracto

El algoritmo para generar el resumen automático de un documento HTML, tiene los siguientes pasos:

1. Aplicar el parser al documento HTML para obtener únicamente el texto plano.
2. Separar los signos de puntuación para que sean independientes de los términos.
3. Encontrar las palabras clave del documento.
4. Encontrar los términos multipalabra del documento.
5. Si el número de técnica es igual a 2, encontrar los términos del título y también los términos con tipografía especial.
6. Formar el párrafo de conocimiento usando la combinación de métrica deseada.

Opción 1: Palabras clave, términos multipalabra, título del documento y términos con tipografía especial.

Opción 2: Palabras clave y términos multipalabra.

7. Comparar el párrafo de conocimiento contra cada oración, del documento, y calcular su peso.
8. Ordenar de forma descendente las oraciones de acuerdo a su peso.
9. Generar el extracto del documento.
10. Fin.

El párrafo de conocimiento está compuesto por un conjunto de términos con alta información semántica dentro de un documento [Bueno *et al*, 2006].

Las tipografías que se tomarán en cuenta son H1, H2, H3, negritas, cursiva y subrayado.

A continuación se muestra de manera gráfica el algoritmo para generar el extracto de un documento HTML.

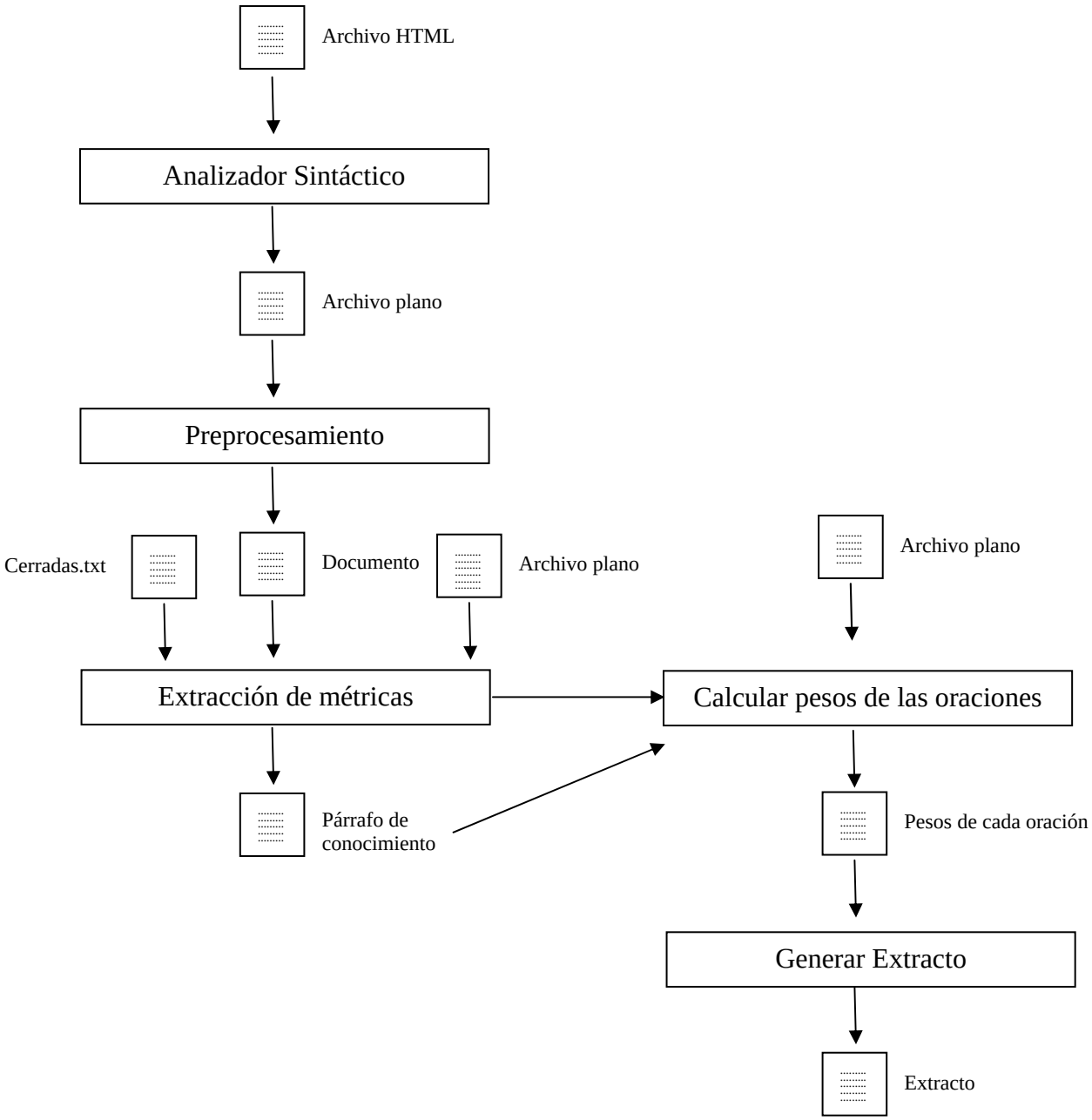
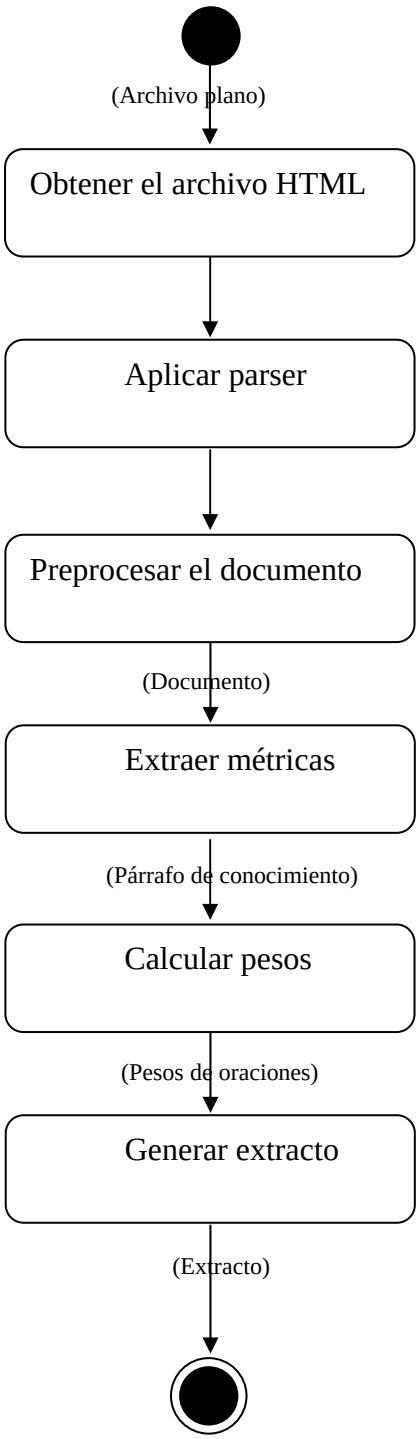


Figura 1 Algoritmo para generar de forma automática el extracto de un documento HTML

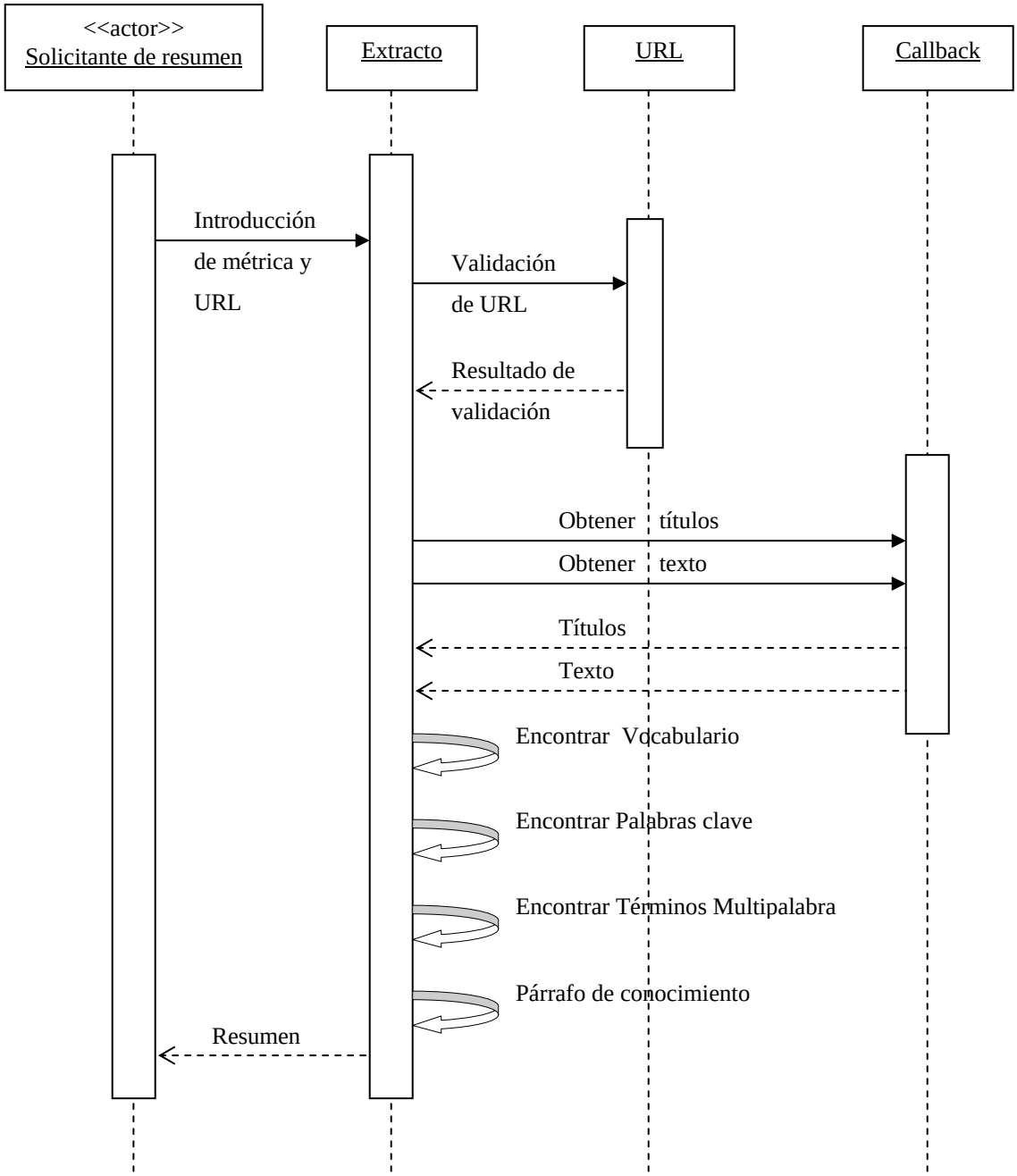
A continuación se muestra el diagrama de actividades del sistema.



La clase Extracto es la clase principal del sistema y su diagrama se muestra a continuación.

Extracto
Vocabulario : Hashtable Kw : Vector Archivo : Vector ArchivoOriginal : Vector Cerradas : Vector VocParrafos : Hashtable Multipalabra : Hashtable(); ArchivoEti : Vector TerminosMultiPalabra : Vector
cargaArchivo(document:String) : int vocabulario() : String palabrasClave() : void vocabularioTM() : String vocabularioPorParrafos() : void funMI(frec1:int, frec2:int, frec3:int) : double etiquetaFile() : void tecnicaParrafoVirtual() : void creaResumen(doc:String) : void creaArchivoPesos(peso:double[], numOracion:int[], oracionesOrig:vector) : void main(args:String[]) : void

Diagrama de Secuencia



3.2 Estructuras de Datos

De acuerdo a las necesidades del problema, se usarán las siguientes estructuras de datos.

Tablas Hash

Para guardar el vocabulario del archivo y los términos multipalabra.

Vectores

Para guardar las palabras clave, el archivo, una copia del archivo original, las palabras cerradas o vacías, el archivo etiquetado, los términos multipalabra, los signos, el vocabulario y también los signos usados en el lenguaje HTML.

3.3 Módulos

Los principales módulos del sistema son los siguientes:

- Parser
- Preprocesamiento del documento
- Palabras Clave
- Términos multipalabra
- Párrafo de conocimiento
- Generar el extracto

A continuación se explicará cada uno de los módulos creados y se mostrarán ejemplos, los cuales toman como base el texto, de la página Web, que se encuentra en el anexo A.

Parser

Este módulo toma como entrada el código fuente de una página Web y cambia el formato del código fuente al formato de texto plano. Inicialmente se pensó en hacer un analizador sintáctico, el cual tomara en cuenta las principales etiquetas del lenguaje HTML. Esta forma de resolver el problema no era adecuada porque la solución no sería completa; por esta razón se decidió usar el paquete javax.swing.text.html, el cual permite analizar documentos HTML. Este paquete toma en cuenta todas las etiquetas y maneja los errores que encuentra en el documento HTML.

Preprocesamiento del documento

Este módulo realiza las siguientes acciones, guarda el documento en el vector archivo, después elimina, dentro del vector, todas las palabras cerradas las cuales se muestran en el Anexo E, y por último, encuentra el vocabulario del documento y lo guarda en el archivo llamado temporalvoc.txt. El vocabulario del documento está formado por dos columnas, la primera columna está compuesta por los términos y la segunda columna tiene la frecuencia de ocurrencia de cada término dentro del documento, en la figura 2 podemos observar un ejemplo del vocabulario del documento del anexo A.

Término	Frecuencia
congreso	8
puebla	3
diputados	3
eventos	2
pericles	2
biblioteca	2
olivares	2
. . .	. . .

Figura 2 Vocabulario del documento del Anexo A

Palabras Clave

El método para encontrar las palabras clave, calcula el valor del punto de transición, utilizando la siguiente fórmula:

$$PT \quad \frac{\sqrt{1 \quad 8I_1} \quad 1}{2}$$

en donde I_1 representa el número de palabras con frecuencia 1 [Jiménez *et al*, 2006].

Después usa un umbral del 25%, se tomó en cuenta este valor, debido a que los experimentos realizados muestran que al tomar una banda de frecuencias desde un 15% hasta un 25% alrededor del punto de transición, se obtienen los mejores resultados [Bueno *et al*, 2006]. Por último agrega al vector de las palabras clave, los términos cuya frecuencia está dentro del umbral. En caso de no encontrar ninguna palabra clave, se usa el método por inspección; en el cual, el punto de transición toma como valor a la frecuencia más alta que no se repite; finalmente, una vez que se encontró el valor del punto de transición, se repite el proceso mencionado anteriormente.

Términos multipalabra

El método para encontrar los términos multipalabra hace uso del concepto de bigramas y de la medida de información mutua entre las palabras. En este trabajo se hizo una modificación muy significativa a los algoritmos que encuentran bigramas, el objetivo era diseñar un algoritmo para encontrar términos multipalabra, sin importar el número de palabras que los formaban.

Párrafo de Conocimiento

El párrafo de conocimiento está formado por la combinación de diferentes métricas. Las dos combinaciones que se tienen son:

- Palabras clave, términos multipalabra, título del documento y términos con tipografía especial.
- Palabras clave y términos multipalabra.

En la figura 3 se muestra un ejemplo del párrafo de conocimiento del texto del anexo A.

El siguiente paso será comparar cada una de las oraciones del documento contra el párrafo de conocimiento, para que de esta manera, encontremos los pesos de cada una de las oraciones.

Se utilizará la función de similitud de Jacccard para determinar que tanto se parece la oración y el párrafo de conocimiento [Pinto & Jiménez, 2006]. La función de similitud de Jaccard es la siguiente:

$$sim(PC, O) = \frac{\#(PC \cap O)}{\#(PC \cup O)}$$

En donde *PC* representa al Párrafo de Conocimiento y *O* representa a la oración.

Párrafo de Conocimiento
congreso
2006
11
28
com
1811
puebla
pue
necesario
diputados
panistas
fundamenten
reforma
electoral
pericles
olivares

Figura 3 Párrafo de conocimiento del documento del Anexo A

Generar el extracto

Las tres oraciones más relevantes formarán el extracto de un documento; en caso de contar con oraciones repetidas, se aplica un filtro para usar la siguiente oración que no se repita. Estas oraciones se ordenan de acuerdo a su posición dentro del documento y se guardan en el archivo resumen.txt.

Los documentos que están formados solamente por una o dos oraciones, tienen como extracto esas oraciones.

La figura 4 muestra el extracto del documento del anexo A.

Congreso del estado de puebla. Necesario que diputados panistas fundamenten la reforma electoral: pericles olivares. comunicate@congresopuebla.

Figura 4 Extracto del documento del anexo A

Otros métodos usados en el sistema son los siguientes:

- cargaArchivo
- cargaCerradas
- quitaCerradas
- Vocabulario
- cargaSignos
- VocabularioTM
- funcionMI
- etiquetaFile

- quitaMultipalabras
- creaArchivoMultipalabras
- creaArchivo
- cargaSignosHTML
- quitaSignosHTML
- creaResumen

A continuación se explicará la información relevante de cada uno de los siguientes métodos.

cargaArchivo()

Este método guarda el documento en el vector archivo. En este vector se tendrá el documento en minúsculas y los signos de puntuación estarán separados de los términos del documento. En este método se cuenta el número de oraciones que hay en el documento.

cargaCerradas()

Lo que hace este método es guardar el archivo de palabras cerradas o vacías, del idioma español, en el vector cerradas; el vector tiene las palabras en minúsculas. El nombre del archivo que contiene las palabras cerradas lo obtenemos del tercer parámetro, al momento de ejecutar el programa.

quitaCerradas()

Este método se encarga de quitar las palabras cerradas que contiene el archivo.

Vocabulario()

Este método encuentra el vocabulario del documento; es decir, encuentra la frecuencia de ocurrencia de cada uno de los términos del documento.

cargaSignos()

El método cargaSignos guarda los signos de puntuación en el vector signos.

VocabularioTM()

El método VocabularioTM encuentra el vocabulario del archivo y calcula el valor de N (tamaño del documento), el cual es obtenido por medio de la diferencia del número de términos menos el número de puntos.

funcionMI()

Este método encuentra el valor de la información mutua de dos términos.

etiquetaFile()

Este método etiqueta cada uno de los términos multipalabra; específicamente, une los términos del bigrama utilizando el símbolo de guión bajo; por último, guarda el archivo en el vector archivoEti.

quitaMultipalabras()

El método quitaMultipalabras aplica un filtro para eliminar a los términos multipalabra que no cubren las características; específicamente, elimina a los términos multipalabra que empiezan o terminan con una palabra cerrada.

creaArchivoMultipalabras()

Este método crea el archivo que contiene a todos los términos multipalabra, el nombre del archivo es Multipalabras.txt.

creaArchivo()

Crea el archivo temporal r.txt el cual tiene tres columnas, la primera columna corresponde al término multipalabra que se va a revisar, la segunda columna contiene la frecuencia de ocurrencia del término multipalabra y la última columna tiene el valor de la información mutua del término multipalabra; un ejemplo de este archivo se encuentra en la figura 5.

Posibles Términos Multipalabra	**Frecuencia**	**Información Mutua**
olivares flores	1	6.977279923
que se	2	3.807354922
actividad legislativa	1	7.971543554
en línea	2	5.672425342
del palacio	1	4.807354922
y participa	1	5.392317423
histórica del	1	4.807354922
.	.	.
.	.	.
.	.	.

Figura 5 archivo generado por el método creaArchivo, obtenido del documento del anexo A

cargaSignosHTML()

Guarda en el vector signosHTML cada uno de los signos usados en el lenguaje HTML. Algunos de los símbolos usados en el lenguaje HTML son “, &, [,], etc. Es importante tomar en cuenta estos símbolos para poder usarlos con el sentido correcto de lo que representan.

quitaSignosHTML()

Como su nombre lo dice, este método elimina los signos HTML de un término.

creaResumen()

Este método crea el archivo Resument.txt, el cual contiene el resumen del documento.

3.4 Archivos

Los archivos que utiliza el sistema son los siguientes:

Cerradas.txt, este archivo contiene las palabras cerradas del idioma español.

Resumen.txt, contiene las oraciones que representan el extracto del documento.

Multipalabras.txt, contiene todos los términos multipalabra del documento.