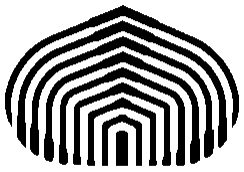


UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

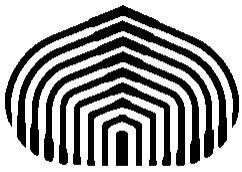
EXTRACCIÓN DE INFORMACIÓN FOCALIZADA BASADA EN RESPUESTAS INCOMPLETAS (FIE)

Por:
FRANCISCO RODRÍGUEZ DRUMOND

Realizado con la asesoría de:
PROF. SORAYA ABAD

PROYECTO DE GRADO
Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar al título de
Ingeniero de Computación

Sartenejas, Septiembre de 2012



UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

ACTA FINAL PROYECTO DE GRADO

EXTRACCIÓN DE INFORMACIÓN FOCALIZADA BASADA EN RESPUESTAS INCOMPLETAS (FIE)

Presentado por:
FRANCISCO RODRÍGUEZ DRUMOND

Este Proyecto de Grado ha sido aprobado por el siguiente jurado examinador:

_____
PROF. 1

_____
PROF. 2

_____
PROF. 3

Sartenejas, X/X/12

Resumen

RESUMEN GOES HERE - To be done.

Acrónimos

FIE	Focalized Information Extraction	Extracción Focalizada de Información
DIA	Document Interrogation Architecture	Arquitectura de Interrogación de Documentos
MBFI	Módulo de Búsqueda Focalizada de Información	Módulo de Búsqueda Focalizada de Información
MDFI	Módulo de Determinación de la Fuente de Incompletitud	Módulo de Determinación de la Fuente de Incompletitud
EFI	Extractor Focalizado de Información	Extractor Focalizado de Información

DUDA: Que hago con los acrónimos que son sólo en español? Deberia poner los acronimos siempre en internet?

Índice general

Índice general vi

Índice de tablas viii

Índice de figuras ix

Introducción 1

1 Planteamiento del Problema 2

2 Marco Teórico 3

2.1 Revisión de trabajos existentes en el área de extracción de Información 3

3 Análisis de incompletitudes en consulta y propuesta para determinar las fuentes de incompletitud 6

4 Diseño 7

4.1 El Módulo de Determinación de la Fuente de Incompletitud (MDFI) 7

4.2 El Módulo de Búsqueda Focalizada de Información (MBFI) 7

5 Detalles de implementación 9

5.1 Consideraciones generales 9

5.1.1 Lenguaje de programación 9

5.1.2 Librerías utilizadas 9

5.2 Implementación del Módulo de Búsqueda Focalizada de Información (MBFI) 10

5.2.1 Diseño de las expresiones regulares para la configuración del Preprocesador de textos y del Extractor Focalizado 11

6 Pruebas 13

6.1 Selección de Documentos de Prueba 13

6.2 Pruebas de Extractor Focalizado 13

7 Resultados **15**

Conclusiones y recomendaciones **18**

Bibliografía **19**

Índice de tablas

7.1 Resultados de la evaluación del Preprocesador de Textos 15
7.2 Resultados detallados la evaluación del Preprocesador de Textos 15
7.3 Resultados detallados la evaluación del Preprocesador de Textos: Designaciones 15
7.4 Resultados detallados la evaluación del Preprocesador de Textos: Escalafón 16
7.5 Resultados detallados la evaluación del Preprocesador de Textos: Jurados de Ascenso . . 16
7.6 Resultados de la evaluación del Extractor Focalizado - Dominio: Designaciones. UnitHit Measure mínimo:.75 17

Índice de figuras

4.1 Diagrama del Módulo de Búsqueda Focalizada de Información 8

5.1 Funcionamiento interno del Módulo de Búsqueda Focalizada 12

Introducción

Este trabajo consiste en el estudio de una variante del problema de enrutamiento de vehículos (VRP por las siglas de Vehicle Routing Problem) el cual se define como un problema de optimización combinatoria del área de logística y distribución de bienes.

Capítulo 1

Planteamiento del Problema

Plantamiento del problema goes here.

Es necesario aclarar que la extracción focalizada de información de este proyecto de grado está acotada en descubrir valores desconocidos de campos. El problema de descubrir incompletitudes puede ser más complicado y abarcar, por ejemplo, el problema de descubrir interrelaciones entre conceptos existentes en la ontología.

Capítulo 2

Marco Teórico

Marco Teórico goes here.

2.1. Revisión de trabajos existentes en el área de extracción de Información

En "A survey of Uncertain Data Algorithms and Applications" (2005) [1], Aggarwal y Fellow presentan un estudio de las tecnologías que se han desarrollado para trabajar con datos inciertos. Los autores definen los datos inciertos como datos que pueden tener estar incompletos o que pueden contener errores. Por ejemplo, se puede tener una base de datos con datos de mediciones meteorológicas cuya precisión depende de los instrumentos utilizados y que por ende sus datos pueden contener errores. Puede darse también el caso en el que no se tenga la información completa, como en la base de datos un censo en la cual no se tiene la información de todos los ciudadanos de un país. En el caso particular de este proyecto de grado se trabaja con datos incompletos.

En base a la definición de datos inciertos, los autores examinan las técnicas más recientes en tres campos fundamentales: modelado de datos inciertos, manejo de datos inciertos y minería de datos inciertos. Este proyecto de grado está enmarcado en las dos primeras categorías: modelado y manejo de datos inciertos. Esto se debe a que se quiere estudiar la forma como modelar el problema así como un conjunto de mecanismos que permitan completar la información que falta mediante extracción focalizada.

Los autores definen una base de datos probabilística como un espacio de probabilidad finito en el cual los resultados son las posibles bases de datos consistentes con un esquema dado. En pocas palabras, se tiene una representación de los "posibles mundos". Existen varias soluciones para ello, como modelar la probabilidad de que una tupla esté en la base de datos o, si se quiere más detalle, la probabilidad de que un atributo de una tupla de base de datos esté presente. Sobre esta base se pueden construir técnicas para

el manejo de datos y para realizar minería sobre ellos.

Por otro lado, Aggarwal y Fellow (2006) [1] proponen utilizar una función de probabilidad sobre la correctitud de los datos presentes en la base de datos. Si bien el objetivo de este proyecto es el desarrollo de un mecanismo para contestar una consulta cuya respuesta no está en su totalidad en la ontología, la aproximación que hacen estos autores puede ser útil para definir una medida de la calidad de los datos presentes en la Base de Datos.

Es importante destacar sin embargo, una referencia que se hace al trabajo "Evaluating Probabilistic Queries over Imprecise Data" (2003), de Cheng, Kalashnikov y Prabhakar [2]. En el mismo se hace un análisis sobre las consultas con datos imprecisos o propensos a errores. Una vez más, esto es diferente de lo que se quiere en este trabajo de investigación: trabajar consultas incompletas. Sin embargo, Cheng et al proponen una clasificación de consultas probabilísticas que puede ser tomada en cuenta para clasificar las consultas incompletas.

Básicamente los autores toman en consideración dos criterios: el tipo de elemento devuelto por la consulta y el uso de funciones agregados. En pocas palabras, cuando se toma en consideración el tipo de elemento devuelto por la consulta se tiene que puede devolver un valor puntual o un conjunto de tuplas. En el contexto de las consultas probabilísticas esto puede tener implicaciones: para las consultas que buscan un valor los autores definen cotas superiores e inferiores que definen intervalos en los que los valores de la función deben estar, con una probabilidad acumulada de 1.

La segunda clasificación toma en cuenta si existen agregados o no. La presencia de agregados puede afectar como se verá la evaluación de consultas incompletas.

Además de estos dos trabajos se han examinado otros 3. Sin embargo, su aporte y utilidad para el presente de trabajo de investigación es menor. Chen, Chen y Xie proponen en "Cleaning Uncertain Data with Quality Guarentees" (2008) [3], una métrica para cuantificar la ambiguedad de una respuesta de consulta bajo semánticas de mundo posibles. Sobre esta base, se podría construir un mecanismo para "limpiar" la base de datos.

El trabajo "On databases with Incomplete Information" de Lipski (1981) [4] es muy citado por otros investigadores en el área de consultas incerteras y sin duda alguna constituye un hito muy importante en ésta area. Sin embargo, en dicho trabajo se busca tratar el tema con un formalismo matemático que va más allá de los alcances de este proyecto.

Por otro lado, Kang y Kim proponen en "Query type classification for web document retrieval"

(2003) [5] un mecanismo de clasificación de consultas para extracción de información del WEB. Sin embargo, dicha clasificación corresponde con el tipo de operación que se desea: úbicar algo por tópico, ubicar un homepage o ubicar un servicio. Dicha clasificación no está enmarcada dentro del presente trabajo de investigación y por ende tiene poca utilidad.

Por último, Rocquenco, Segoufin y Viano presentan en "Representing and querying XML with incomplete information" (2001) [6] un modelo para hacer consultas incompletas sobre XML. Existe cierto paralelismo con lo que se quiere hacer en este trabajo de investigación: realizar una segunda extracción de información cuando no se pueda contestar una consulta con lo que está en la ontología. Sin embargo, dicho trabajo no fue de utilidad para la clasificación de las consultas.

Capítulo 3

Análisis de incompletitudes en consulta y propuesta para determinar las fuentes de incompletitud

Capítulo 4

Diseño

El sistema para realizar la Extracción Focalizada de Información está conformado por dos módulos: el Módulo de Determinación de la Fuente de Incompletitud y el Módulo de Búsqueda Focalizada de Información.

4.1. El Módulo de Determinación de la Fuente de Incompletitud (MDFI)

4.2. El Módulo de Búsqueda Focalizada de Información (MBFI)

El Módulo de Búsqueda Focalizada de Información es el encargado de encontrar los valores de los campos cuyos valores son desconocidos o incompletos para una consulta dada. Dichos valores son buscados en un conjunto de documentos presentes en el sistema de archivos donde se ejecute el sistema utilizando información de contexto proporcionada por el Módulo de Determinación de la Fuente de Incompletitud (MDFI).

Para llevar a cabo esta tarea, el MBFI tiene dos componentes: un preprocesador de texto y un extractor focalizado de información.

El Preprocesador de Texto tiene como objetivo preparar el conjunto de documentos sobre el cual se hará la Búsqueda Focalizada de Información y dejarlo listo para la realización de la extracción. Las tareas para ello incluyen leer el documento en su formato original y extraer el fragmento o el conjunto de fragmentos del documento que están relacionados con el dominio sobre el cual se hará la Búsqueda Focalizada de Información.

El preprocesamiento de texto es una tarea que se realiza *offline*. Es decir, cada conjunto de documentos

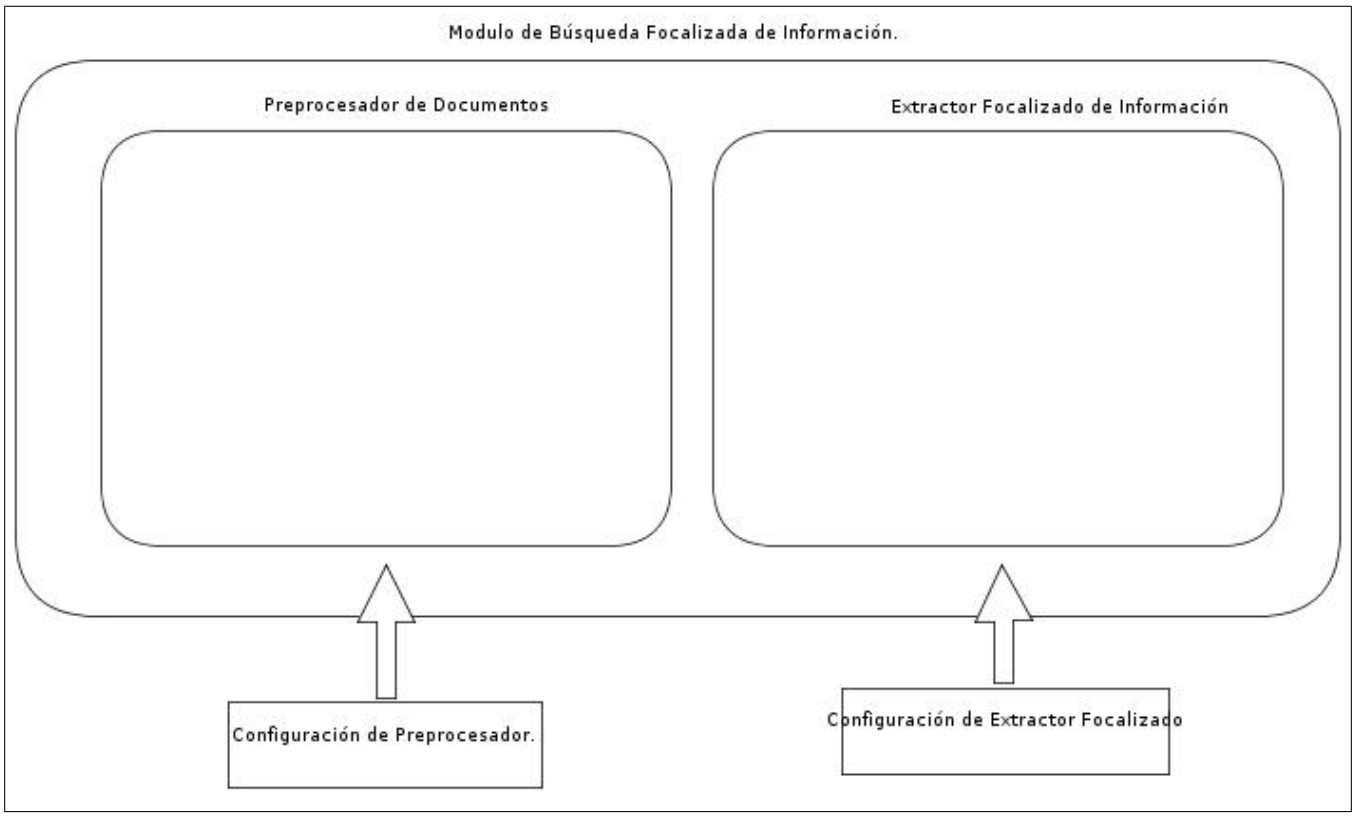


Figura 4.1: Diagrama del Módulo de Búsqueda Focalizada de Información

puede ser preprocesado previamente de forma para que las consultas sean respondidas lo más rápidamente posible.

Por su parte, el Extractor Focalizado de Información (EFI) tiene como objetivo extraer el valor del campo faltante utilizando como materia prima el conjunto de documentos preprocesados y un contexto de extracción. El contexto de extracción consiste en toda la información que el MDFI puede recopilar sobre la consulta que se está realizando que puede ser de útil para hacer la extracción focalizada de información. Puede incluir por ejemplo valores de campos que están relacionados con el campo faltante dentro del dominio sobre el cual se hace la búsqueda.

El MDFI trabaja directamente con el Extractor Focalizado de Información para encontrar los valores faltantes. Esto es, una vez que se determinan las fuentes de incompletitud, el EFI recibe solicitudes de búsqueda para cada uno de los campos con valores faltantes. Esas solicitudes permiten descubrir y completar la ontología y dar respuesta a las consultas que tiene el usuario. El preprocesador, como se ha dicho, se ejecuta una única vez por conjunto de documentos.

Capítulo 5

Detalles de implementación

5.1. Consideraciones generales

5.1.1. Lenguaje de programación

El lenguaje de programación utilizado para la implementación del sistema de Extracción Focalizada de Información fue Java en su versión 1.6.0_20.

5.1.2. Librerías utilizadas

¡Para parsear las consultas, se trabajó con las librerías elaboradas por Andueza y Arteta (año) en su proyecto de Grado (referenciar). Hablar aqui de ellas.¿

Se utilizaron ademas las siguientes librerías Open Source según la necesidad listada a continuación:

- Para desplegar mensajes de traza y errores se utilizó la librería Log4J. Dicha librería permite en lugar de imprimir mensajes por el Standard Output, imprimir mensajes dependiendo de un nivel de desarrollo espeficiado en un archivo de configuración. A la hora de desplegar un mensaje se elige un nivel de desarrolo: traza, error, error fatal, warning, entre otros. Luego con modificar el archivo de configuración de la librería se puede fácilmente apagar y prender los niveles para que se impriman o dejen de imprimir algunos mensajes. Se utilizó la version 1.2.16.
- Para procesar archivos PDF se utilizó la librería Jericho y Tika, en sus versiones 1.6 y 1.1.
- Para procesar archivos HTML se utilizó la librería Jericho, en su versión 3.2.
- Para procesar archivos .doc se utilizó la librería POI, en su versión 3.8.

- Para leer archivos de configuración de XML se utilizó la librería que ofrece Java para parsear y consultar XML, JAXP, en su versión 1.4.

5.2. Implementación del Módulo de Búsqueda Focalizada de Información (MBFI)

Como se ha explicado en 4.1, el Módulo de Búsqueda Focalizada de Información (MBFI) es el módulo que encuentra los valores de los campos cuyos valores son desconocidos o incompletos para una consulta dada. Estos valores son buscados en un conjunto de documentos definidos por el usuario y se encuentran tomando en cuenta un contexto de extracción que contiene información útil para la búsqueda. El MBFI tiene dos componentes: un preprocesador de texto y un extractor focalizado de información.

En grandes rasgos, el funcionamiento y la implementación de ambos componentes son bastantes similares. Ambos componentes operan como extractores de texto utilizando expresiones regulares definidas previamente por un usuario experto. Dichas expresiones regulares funcionan como delimitadores del fragmento de texto que interesa según la tarea que se esté realizando: en el caso del preprocesador de texto, el objetivo es extraer el fragmento de texto que está relacionados con el dominio sobre el cuál se realiza la extracción focalizada; el extractor focalizado busca encontrar el valor del campo faltante para responder una consulta.

Ambos componentes están hechos para trabajar con un archivo de configuración en XML que especifica los parámetros necesarios para realizar el preprocesamiento y la extracción. En particular los parámetros que se guardan en estos archivos de configuración incluyen las expresiones regulares, las rutas de directorio de los archivos de entrada, la ruta de los archivos de salida para el caso del preprocesador, entre otras cosas. De esta manera, el usuario del sistema debe preocuparse tan sólo por entender cómo realizar un archivo de configuración para cada uno de los dominios necesarios.

Más adelante se expondrán algunas consideraciones sobre las expresiones regulares de interés para el entendimiento de la implementación del MBFI.

Ahondando más en el componente de pre-procesamiento, su objetivo como se ha dicho es extraer de cada documento el conjunto de fragmentos que está relacionado con el dominio sobre el cual se hace la búsqueda focalizada. En tal sentido tiene como entrada un conjunto de documentos en su formato original (PDF, Portable Readable Document; HTML , Hyper Text Markup Language; DOC y DOCX, archivos de Microsoft Word; entre otros) y un archivo de configuración. Su salida es un archivo .txt en el cual se tiene el conjunto de fragmentos que están relacionados con el dominio en cuestión según lo especificado por las expresiones regulares. Para procesar los documentos se utilizaron las librerías descritas anteriormente.

En cuanto al componente de extracción focalizada, su objetivo es dado un conjunto de fragmentos de documentos que puede contener valores de campos necesarios para responder una consulta, encontrar el valor de esos campos. Para ello opera en 2 fases: primero rompe cada documento en unidades relacionadas con el dominio en cuestión. Tomando en cuenta los 3 dominios que se consideraron para este trabajo, las unidades vienen siendo designaciones, ingresos y ascensos dentro del escalafón y designación de un jurado de ascenso, por trabajo. Este refinamiento sobre los documentos es hecho utilizando también expresiones regulares y su objetivo es aislar fragmentos de texto donde haya una cierta cohesión que permita ubicar la respuesta utilizando el contexto de extracción.

Una vez separados los documentos en unidades, se utiliza el contexto de extracción para ordenar las unidades en un orden de preferencia. Dicho orden se realiza tomando un valor denominado *UnitHitMeasure* que busca medir por medio de una media ponderada cuánto se aproxima o relaciona a la unidad correcta según lo especificado por el contexto de extracción. Por ejemplo, si se está trabajando con designaciones de cargos, se buscan ordenar las unidades tomando en cuenta el contexto de extracción y los valores de campos conocidos sobre una designación las designaciones según la presencia de los valores conocidos en cada unidad.

El UnitHitMeasure es calculado de la siguiente manera. Cada campo tiene un peso que es asignado por un usuario experto en el archivo de configuración. Cada contexto de extracción tendrá uno o más valores de campos conocidos. Para cada campo presente el contexto de extracción cuyo resultado es conocido y que está presente en la unidad se suma el peso asociado a ese campo. Luego se divide esa sumatoria entre la máxima suma posible (el valor de la suma si todos los campos cuyo valor se conocen estuviesen en la unidad con los valores conocidos). De esta manera se obtiene un número del 0 al 1 que indica el grado de relación que tiene la unidad en cuestión con el contexto de extracción focalizado.

Vale acotar que a la hora de determinar si un valor está presente en la unidad se pueden realizar modificaciones del String con el objetivo de incorporar el uso de sinónimos, modificaciones de formatos de fecha, entre otros, que pueden permitir aumentar la posibilidad de que un campo tenga esté contenido dentro de una unidad de una forma equivalente en cuando a significado. Por ejemplo, se implementó una funcionalidad para dada una fecha en un formato cualquiera, generar las formas de fechas más comunes según lo utilizado en los documentos.

Una vez ordenadas las unidades en orden de precedencia se ubica el valor del campo que se quiere encontrar por medio de expresiones regulares.

En la figura 5.2 puede apreciarse el funcionamiento interno del MBFI.

5.2.1. Diseño de las expresiones regulares para la configuración del Preprocesador de textos y del Extractor Focalizado

Explicar esto.

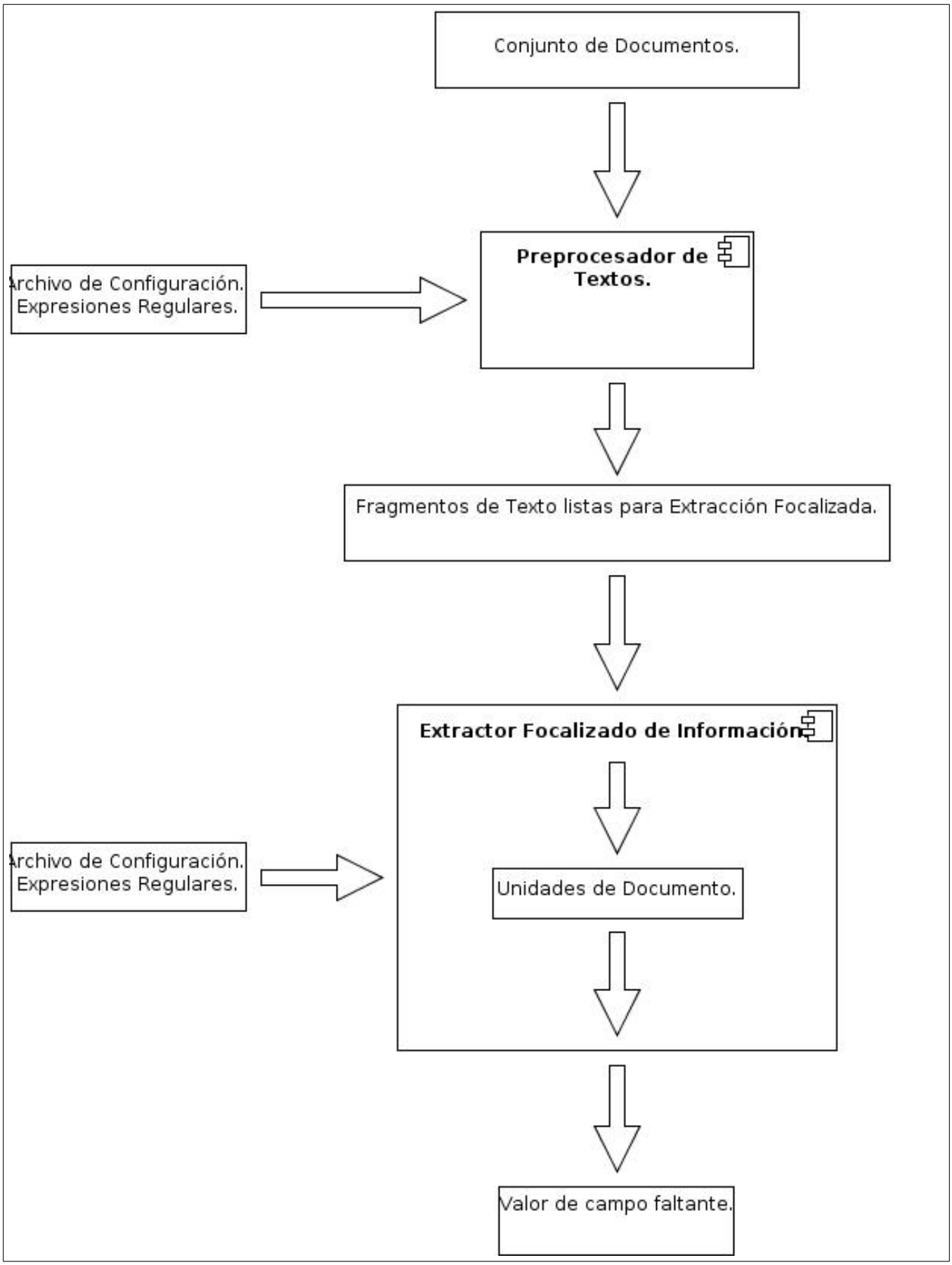


Figura 5.1: Funcionamiento interno del Módulo de Búsqueda Focalizada

Capítulo 6

Pruebas

Descripción del conjunto de pruebas ¡Hablar de los dominios: se trabajó sobre designaciones, etc¿

Duda: esto es una especie de marco metodológico?

6.1. Selección de Documentos de Prueba

La selección de los documentos se realizó sobre el conjunto de las actas de los Consejos Directivos y Académicos de la Universidad Simón Bolívar. Con el objetivo de probar apropiadamente el Extractor, se decidió trabajar con las actas desde el año 1998 hasta el año 2012. Esto permite que las pruebas se hagan sobre un conjunto de documentos semiestructurado, pero que puede mostrar variabilidad en su redacción y estructura en los años.

Del conjunto de actas en el intervalo de tiempo especificado, se hizo una preselección de los documentos para utilizar las actas de los Consejos Extraordinarios. Adicionalmente, según cada dominio de aplicación se extrajeron las actas que sí contienen información sobre los dominios. Es decir, se utilizaron los documentos que contienen información sobre los dominios elegidos.

Para probar el preprocesador, se tomó un conjunto de archivos seleccionados aleatoriamente con uniformidad sobre los años. La prueba consistió en ver si el fragmento de texto extraído por el preprocesador coincidía exactamente, estaba contenido, contenia o era completamente disjunto con el segmento de texto que debería ser extraído. Esta prueba se hizo manualmente.

6.2. Pruebas de Extractor Focalizado

Para probar el extractor focalizado, se tomó un conjunto de prueba seleccionado aleatoriamente con uniformidad sobre los años. De cada archivo seleccionado, se tomaban hasta 3 designaciones (falta explicalro para los otros dominios), que se utilizaban para realizar pruebas. Una prueba consite en hacer una búsqueda focalizada sobre cada uno de los campos de una designación: se asume que se tienen algunos valores de esa designación (un contexto de extracción) y se buscan los campos faltantes.

Estas pruebas se hicieron variando 2 parámetros. En primera instancia, se varió sobre el minimum hit measure (varió entre 0.25;0.5;0.75;1) que mide la proporción de los campos con valores conocidos que aparecen en una unidad de extracción (designación). Luego para cada hit measure se varió sobre la probabilidad de que cada uno de los campos fuese desconocido. Esto es, para cada hit measure al hacer una prueba se determinaba aleatoriamente cuantos campos tenían valores conocidos. Esto para poder simular los casos de la vida real en los que no se tienen todos los campos de una designación menos el conocido. Las probabilidades de no tener un valor puntual variaron entre 0.1;0.25;0.75.

Capítulo 7

Resultados

Tabla 7.1: Resultados de la evaluación del Preprocesador de Textos

Dominio	Correctos	Incorrectos
Designaciones	.953	0.046
Designaciones	.953	0.046

Tabla 7.2: Resultados detallados la evaluación del Preprocesador de Textos

Dominio	**Aprobados**		**No aprobados**	
	Correctos	Con texto de más	Incompletos	Incorrectos
Designaciones	87.69 %	7.69 %	3.07 %	1.54 %

Tabla 7.3: Resultados detallados la evaluación del Preprocesador de Textos: Designaciones

Dominio	**Aprobados**		**No aprobados**	
	Correctos	Con texto de más	Incompletos	Incorrectos
Designaciones	87.69 %	7.69 %	3.07 %	1.54 %
	Aprobados: **95.39**		No aprobados: **4.61 %**	

El conjunto de prueba es de tama no 73, un tercio de la población.

Por que el extractor falla con designaciones? (Observaciones hechas al hacer las pruebas que pueden ser útiles en el análisis de datos)

1. Las unidades de informacion no son precisas. A veces hay multiples designaciones en una misma linea.

2. Hay multiples designaciones que coinciden en un mismo día para una misma persona.

3. Hay ratificaciones, correcciones, posteriores a la designacoines que modifican el resultado que uno pensaria correcto.

4. Hay errores de tipeo por parte de la secretaria.

5. Hay casos "patologicos"que es imposible generalizar con expresiones regulares que no sean hechas a la medida.

6. Hay casos en los que los campos tienden a tener muchos valores null y al no encontrar la respuesta en el

Tabla 7.4: Resultados detallados la evaluación del Preprocesador de Textos: Escalafón

Dominio	**Aprobados**		**No aprobados**	
	Correctos	Con texto de más	Incompletos	Incorrectos
Designaciones	80.51 %	12.98 %	6.4 %	0 %
	Aprobados: **93.6**		No aprobados: **6.4 %**	

El conjunto de prueba es de tama no 77, un tercio de la población.

Tabla 7.5: Resultados detallados la evaluación del Preprocesador de Textos: Jurados de Ascenso

Dominio	**Aprobados**		**No aprobados**	
	Correctos	Con texto de más	Incompletos	Incorrectos
Designaciones	77.55 %	16.32 %	0 %	6.12 %
	Aprobados: **93.88**		No aprobados: **6.12 %**	

El conjunto de prueba es de tama no 49, un tercio de la población.

mejor hit, se procede al segundo. Arreglar esto en el extractor? Hay muchos casos en los que un segundo match ayuda.

Tabla 7.6: Resultados de la evaluación del Extractor Focalizado - Dominio: Designaciones. UnitHit Measure mínimo:.75

Campo	Prob. Campo Faltante	**P. Aprobados**		**P. No aprobados**	
		Correctos	Con texto de más	Incompletos	Incorrectos
EsAsignado.calificacion	0	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.10	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.25	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.50	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
EsAsignado.fechaAsignacion	0	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.10	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.25	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.50	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
EsAsignado.fechaFinal	0	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.10	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.25	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.50	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
EsAsignado.motivo	0	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.10	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.25	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.50	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
Profesor.Nombre	0	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.10	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.25	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	
	0.50	87.69 %	7.69 %	3.07 %	1.54 %
		Aprobados: **95.38 %**		No aprobados: **4.61 %**	

Prob. Campo Faltante es la probabilidad de que no se tenga el valor uno de los campos que se utilizan para hacer extracción focalizada.

Conclusiones y recomendaciones

Destacar:

1. íngeniería del documento'. 2. Dependencia de un extractor más inteligente.

Bibliografía

[1] Aggarwal and Fellow, "A survey of uncertain data algorithms and applications", *IEEE*, vol. 21, pp. 609–623, 2009.

[2] D. KALASHNIKOV R. CHENG and S.PRABHAKAR, "Evaluating probabilistic queries over imprecise data", *Proc ACM SIGMOD*, 2003.

[3] J. CHEN R. CHENG and Xie X., "Cleaning uncertain data with quality guarantees", *Proceedings of the VLDB Endowmen*, vol. 1,1, pp. 722–735, 2008.

[4] Lipski Witold, "On databases with incomplete information", *Journal of ACM*, vol. 8,1, 1981.

[5] In-ho Kang and Kim GilChang, "Query type classification for web document retrieval", *Proc Sigir '03*, 2003.

[6] Segoufin Rocquenco and Viano, "Representing and querying xml with incomplete information", *PODS'01*, 2001.