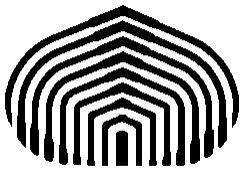


UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

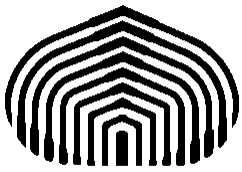
EXTRACCIÓN DE INFORMACIÓN FOCALIZADA BASADA EN RESPUESTAS INCOMPLETAS (FIE)

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 inglés?

Índice general

Índice de tablas

Índice de figuras

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

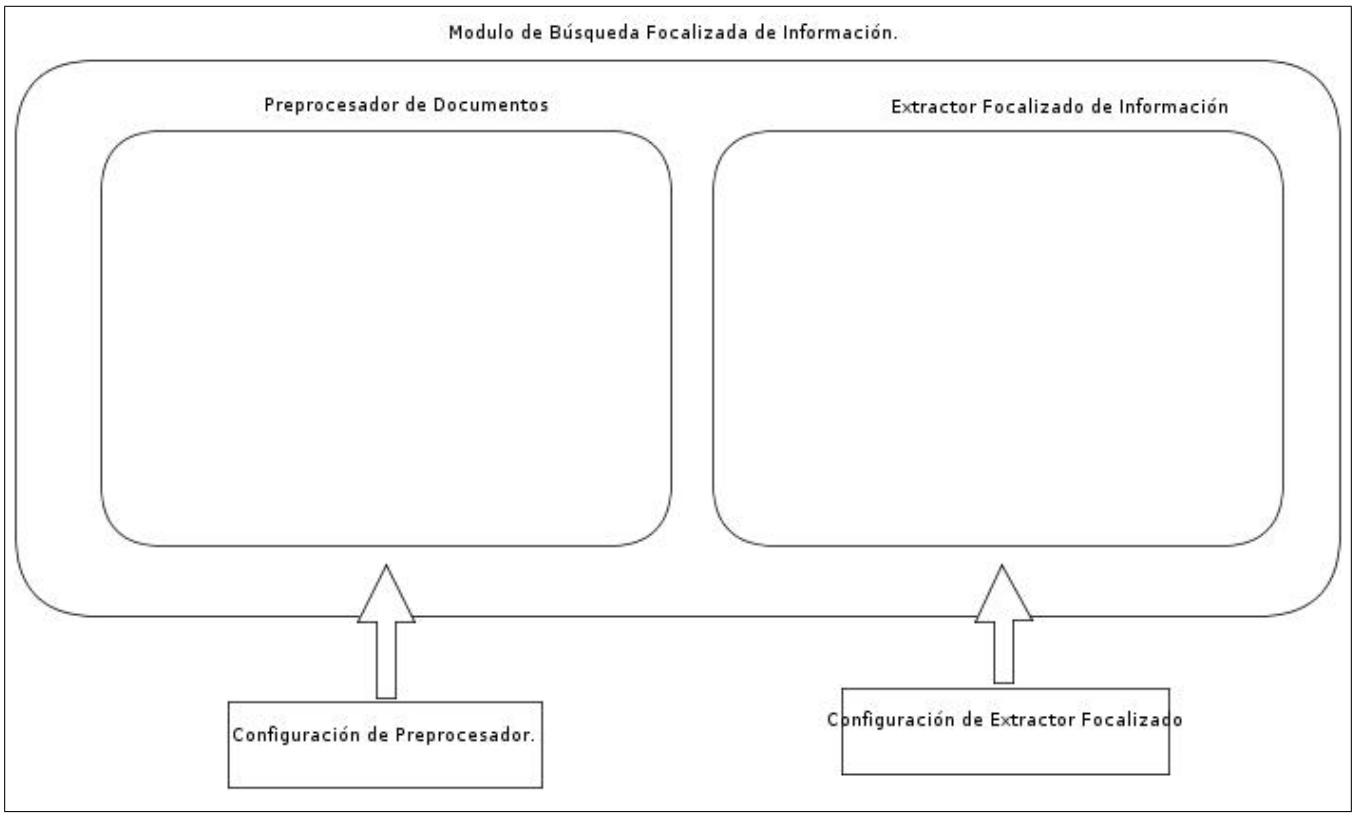


Figura 4.1: Diagrama del Módulo de Búsqueda Focalizada de Información

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:

- Para desplegar mensajes de traza y errores se utilizó la librería Log4J. Dicha librería permite en lugar de imprimir mensajes por el Standard Output, imprimirlos 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 y cambiar rápidamente entre etapas de desarrollo, debugging, pruebas y puesta en producción. Se utilizó la version 1.2.16.
- Para procesar archivos PDF se utilizó la librería PDFBox 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 Determinación de Fuente de Incompletitud (MDFI)

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

Como se ha explicado en 4.2, 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.

5.3.1. Preprocesador de Textos

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) 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.

5.3.2. Extractor Focalizado

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. Para ilustrar este punto, 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 semántica (un fragmento de texto que se refiera a los mismo) 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 a través de una media ponderada cuánto se aproxima o relaciona cada unidad encontrada en los documentos 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 de la ontología 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.

DUDA: Profe, quiere que coloque un ejemplo de cómo se calcula esto para que quede más claro? O se entiende para un lector cualquiera?

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, traducción de idiomas, desglose de siglas, 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. Esto debe ser sin embargo implementado según el contexto y requiere una modificación en los archivos XML, en su lector y la implementación del código para realizar las modificaciones. 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. Cabe destacar que el usuario puede especificar un *Minimum HitMeasure* (valor mínimo de HitMeasure) para que una respuesta sea considerada válida: mientras más cercana a 1 será más probable que la respuesta sea correcta aunque se disminuye la posibilidad de encontrar el valor.

En la figura 5.3.2 puede apreciarse el funcionamiento interno del MBFI.

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

La elaboración de las expresiones regulares es una tarea que debe realizar un usuario experto que conozca el dominio sobre el cual se quiere realizar la extracción focalizada. En general los dominios admisibles son aquellos con textos estructurados o semi-estructurados, donde se puedan deducir patrones en la redacción de los contenidos. Aún teniendo cierta regularidad en la forma del texto, existen retos como errores de redacción, ortografía, transcripción, formato en los documentos fuente, presencia de tablas entre otros, que dificultan el uso de expresiones regulares. Es posible también por ejemplo que hayan presentes un conjunto de carácteres que parezcan espacios en blanco pero que no hagan match con una expresión regular porque no son espacios.

En todo caso, lo relevante en esta descripción de la implementación del sistema es que para atender esta dificultad los archivos de XML permiten contener varias expresiones regulares ordenadas por orden de precedencia. Los textos se exploran con las expresiones regulares según el orden de precedencia: si no se encuentra un fragmento de texto con una expresión regular se explora posteriormente con otra de menor

precedencia y así sucesivamente.

El objetivo de esta precedencia es que el usuario pueda colocar con alta precedencia expresiones regulares que sean bastante selectivas y restrictivas, con el objetivo de asegurar correctitud. Luego a medida que se desciende en la escala de precedencia se puede relajar la restrictividad de la expresión regular, con el objetivo de aumentar la probabilidad de encontrar un fragmento de texto. Relajar la restrictividad implica que la respuesta no sea exactamente la buscada y que por el contrario pueda contener caracteres de más (esto es, que el fragmento de texto obtenido contenga al fragmento de texto correcto). Esto puede ser deseable en muchos casos, dada la complejidad de utilizar expresiones regulares.

En el Apéndice A se muestran fragmentos de archivos de configuracion.

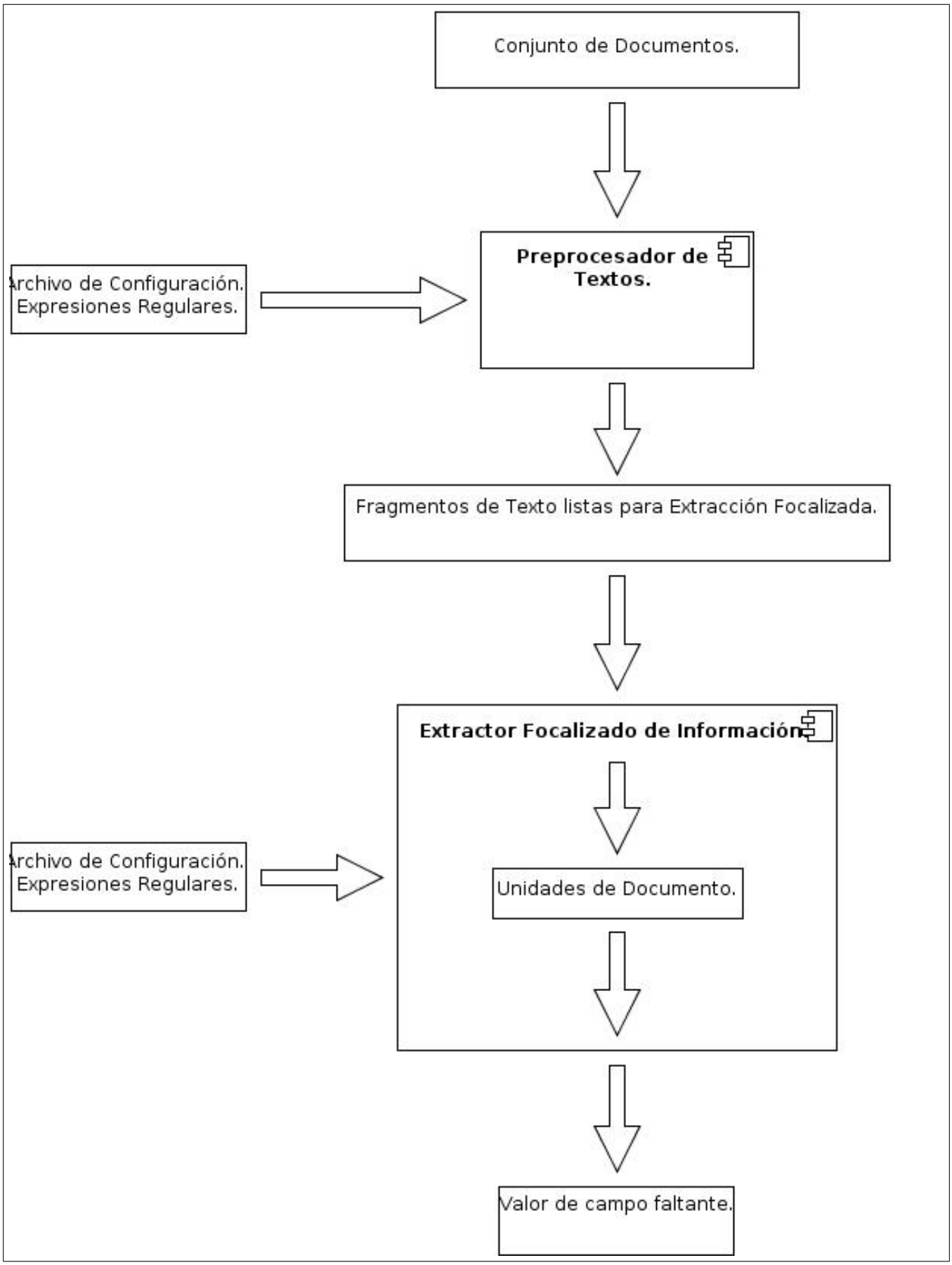


Figura 5.1: Funcionamiento interno del Módulo de Búsqueda Focalizada

Capítulo 6

Pruebas

6.1. Consideraciones Generales

Con el objetivo de realizar las pruebas sobre el sistema se trabajó como ya se ha dicho sobre tres dominios relacionados con el funcionamiento interno de la Universidad Simón Bolívar: las designaciones de cargos, los ingresos y ascensos dentro del escalafón de Profesores y la designación de Jurados para Trabajos de Ascenso. Para trabajar con esos 3 dominios se trabajó con las Actas de Consejos Directivos y Actas de Consejos Académicos. La elección de los 3 dominios se tomó buscando tener una variedad de estilos de texto semiestructurado para obtener resultados diversos sobre el funcionamiento.

La selección de los documentos se realizó sobre el conjunto de las actas de los Consejos Directivos y Académicos de la Universidad Simón Bolívar desde el año 1998 hasta el año 2012. Esto permite que las pruebas se hagan sobre un conjunto de documentos que pueden tener variabilidad en su estilo, 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 Ordinarios. 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, descartando aquellas actas que no contenían texto de interés.

Una vez definidos los dominios se definieron varios conjuntos de prueba. En primera instancia se probaron los Módulos de Determinación de Fuente de Incompletitud y Módulo de Búsqueda Focalizada de Información por separado. Luego se hicieron pruebas sobre ambos compontentes integrados. A continuación se describen con mayor profundidad las pruebas realizadas sobre el sistema.

6.2. Pruebas unitarias sobre el Módulo de Determinación de Origen de la Incompletitud.

6.3. Pruebas unitarias sobre el Módulo de Búsqueda Focalizada de Información

Para probar el Módulo de Búsqueda Focalizada de Información se hicieron diseñaron 2 pruebas por dominio. Por un lado de hicieron pruebas para el Preprocesador de Texto y en segunda instancia Pruebas para el Extractor Focalizado.

6.3.1. Pruebas del Preprocesador de Texto

Para probar el preprocesador, se tomó un conjunto de archivos seleccionados aleatoriamente con uniformidad sobre los años. La muestra elegida fue de un tercio de la población total de actas existentes, para cada dominio. Para las designaciones se trabajó con 73 documentos, para las incorporaciones y ascensos dentro del escalafón se trabajó con 77 documentos y para la designación de Trabajos de Jurado de Ascenso se trabajó con 49 documentos.

Las pruebas en esencia consistieron en ver si el fragmento de texto extraído por el preprocesador coincidía exactamente, estaba contenido, contenía o era completamente disjunto con el segmento de texto que debería ser extraído. Esto es: pueden verse ambos fragmentos de textos como secuencias de caracteres y se puede examinar si ambas secuencias coinciden por completo, tienen relaciones de contención o son completamente diferentes.

Esta prueba se hizo manualmente: se examinó el resultado de la extracción realizada por el preprocesador y se buscó dentro de cada documento el fragmento que debería extraerse.

Se definió como *aprobado* que o bien el fragmento extraído sea completamente igual a la respuesta esperada (caso correcto) o si el fragmento extraído contiene a la respuseta esperada y algunos caracteres de más. Dicho caso se considera aprobado porque en esencia el que el documento preprocesado no contenga algunas líneas más de código no afecta considerablemente la extracción focalizada. Como *no aprobado* se entienden los casos en los que la respuesta está incompleta (la respuesta esperada contiene al fragmento de texto obtenido) o que los fragmentos son completamente disjuntos.

6.3.2. Pruebas de Extractor Focalizado

Para probar el extractor focalizado, se tomó al igual que en las pruebas del Preprocesador de texto un conjunto de prueba seleccionado aleatoriamente con uniformidad sobre los años. La muestra es de un tercio de la población total. Para hacer esto de cada archivo seleccionado, se tomaban hasta 3 unidades de documento según lo explicado en 5.3.2. Dichas unidades fueron vaciadas en un archivo de pruebas que posteriormente era leído por un módulo de pruebas que se encarga de probar el extractor focalizado automáticamente.

Una prueba individual consiste en hacer una búsqueda focalizada sobre un campo presente en una unidad de documento. Una unidad de documento da por lo tanto para hacer tantas pruebas como campos con valor tenga esa unidad. Si el dominio tiene por ejemplo 3 campos y se tiene una unidad de documento con 2 campos, se realizan 2 pruebas para esa unidad: una para cada campo. Las pruebas se hacen iterando entonces sobre cada uno de los campos con valores en la unidad y buscando extraer el valor del mismo utilizando un contexto de extracción dado por los valores de los campos que sí se tienen.

Para simular condiciones en las que no se tienen todos los valores de los campos relacionados con una unidad - esto es, que a pesar de que la unidad contenga valores para varios campos a la hora de hacer la búsqueda focalizada el *contexto de extracción dado por la consulta* no contenga todos esos valores -, los contextos de extracción se construyeron con un subconjunto de los valores de los campos presentes en la unidad de documento. Este subconjunto se determina aleatoriamente: se genera un número al azar y si ese número es mayor que una *probabilidad de tener cada campo* se incluye en el contexto de extracción. De esta manera se evitar suponer para estas pruebas que los contextos de extracción son completos.

Las pruebas se realizaron iterando el valor de la *probabilidad de tener cada campo*. Se tomaron como *probabilidades de tener cada campo* 1; 0.66 y 0.33. Adicionalmente, se iteró también sobre el *Minimum Hit Measure* (según lo definido en 5.3.2) con el objetivo de probar el efecto que tiene la elección de este parámetro en el funcionamiento del extractor focalizado. Los valores sobre los que se realizaron las pruebas fueron: 1; 0.66 y .33.

De esta manera, y resumiendo, las pruebas se realizaron tomando tres unidades de documento de cada documento de cada dominio. Para cada unidad se itera sobre la *probabilidad de tener cada campo* y el *Minimum Hit Measure*. Los resultados fueron posteriormente totalizados por dominio y tabulados.

Capítulo 7

Resultados

A continuación se muestran los resultados de las pruebas realizadas.

7.1. Resultados de las pruebas unitarias sobre el Módulo de Determinación de Origen de la Incompletitud.

7.2. Resultados de las pruebas unitarias sobre el Módulo de Búsqueda Focalizada de Información

Tabla 7.1: Resultados detallados la evaluación del Preprocesador de Textos

Dominio	**Aprobados**			**No aprobados**		
	Correctos	Con texto de más	**Total aprobados**	Incompletos	Incorrectos	**Total no aprobados**
Designaciones	87.69 %	7.69 %	**95.39 %**	3.07 %	1.54 %	**4.61 %**
Escalafón	80.51 %	12.98 %	**93.6 %**	6.4 %	0 %	**6.4 %**
Jurados de Ascenso	77.55 %	16.32 %	**93.88 %**	0 %	6.12 %	**6.12 %**

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

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

Tabla 7.2: 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. 3. Que el software se hizo suponiendo la existencia de unidades de documentos.

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.

Apéndice A

Ejemplos de archivos de configuración XML de Preprocesador de Texto y Extractor Focalizado

A continuación se presentan dos archivo de configuración del Preprocesador de Texto y del Extractor Focalizado del Módulo de Búsqueda Focalizada.

Código A.1: Archivo de configuración del Preprocesador de Texto con Dominio Designaciones

```
<?xml version="1.0" encoding="UTF-8"?>

<Filesources>

    <Filesource>

        <Name>Designaciones</Name>

        <InputFilePath>/home/frandres/Eclipse/workspace/ExtractionModule/tests/
            Designaciones/rawDocuments/</InputFilePath>

        <OutputFilePath>/home/frandres/Eclipse/workspace/ExtractionModule/tests/
            Designaciones/preProcessedDocuments/</OutputFilePath>

        <RegExps>

            <RegExp>
                <Priority>1</Priority>
                <String>De acuerdo al numeral 7 del articulo 16(.*?)[\^0-9 ][23(?:IV)(?:
                    III)(?:II)][\.\t].*</String>
            </RegExp>

            <RegExp>
                <Priority>2</Priority>
                <String>.*(?:[Ii][Nn][Ff][Oo][Rr][Mm][Ee] [Dd][Ee][Ll] [Rr][Ee][Cc][Tt][
                    Oo][Rr]).*?([Dd][Ee][Ss][Ii][Gg][Nn][Aa][Cc][Ii][Oo\u00f3][Nn](?:[Ee
                    ][Ss]){0,1}.*?)[^0-9 ][23(?:IV)(?:III)(?:II)][\.\t].*</String>
            </RegExp>

        </RegExps>

        <readAllFiles>1</readAllFiles>

        <outputExtension>.txt</outputExtension>

    </Filesource>
</Filesources>
```

Código A.2: Fragmento del Archivo de configuración del Extractor Focalizado de Texto con Dominio Designaciones

```
<Configuration>
    <UnitRegExp>^(.+?)$</UnitRegExp>
    <MinimumHitRatio>.5</MinimumHitRatio>
    <DocumentsFilePath>/home/frandres/Eclipse/workspace/ExtractionModule/tests/
        Designaciones/preProcessedDocuments/</DocumentsFilePath>
    <FieldDescriptors>

        <FieldDescriptor>
            <FieldName>EsAsignado.calificacion</FieldName>
            <SpecificRegExps>
                <SpecificRegExp priority="1" >([Jj]ef.*?)(?:(?:a\s+partir\s+del{0,1}
                    \s)|(?:desde)|(?:hasta)|(?:por motivo)|(?:en sustitu)|(?:del\s\d)
                    |(?:al\s\d)|(?:\s+entre\s+el\s+\d)|(?:por\s+el\s+periodo\s+)).*</
                    SpecificRegExp>
                <SpecificRegExp priority="2" >([Cc]oordinador.*?)(?:(?:a\s+partir\s
                    +del{0,1}\s)|(?:desde)|(?:hasta)|(?:por motivo)|(?:en sustitu)
                    |(?:del\s\d)|(?:al\s\d)|(?:\s+entre\s+el\s+\d)|(?:por\s+el\s+
                    periodo\s+)).*</SpecificRegExp>

                <SpecificRegExp priority="13" >([Jj]ef.*)</SpecificRegExp>
                <SpecificRegExp priority="14" >([Cc]oordinador.*)</SpecificRegExp>

            </SpecificRegExps>
            <Weight>1</Weight>
            <isDate>0</isDate>
        </FieldDescriptor>
    </FieldDescriptors>

</Configuration>
```