**Capítulo IV - Propuesta**

En esta sección se explicará en detalle la técnica propuesta para la identificación de los atributos de calidad.

<AMPLIAR>

**IV.1. Análisis de la Entrada**

Este bloque de tareas tiene como finalidad llevar a cabo un análisis léxico y sintáctico de los textos definidos como entrada para la técnica. Esta entrada es la información generada como salida por la herramienta Aspect Extractor Tool [X]. Se considera que la salida de dicha herramienta, luego del procesamiento con la técnica de “análisis semántico” [X], es la lista de casos de uso, los early aspects detectados y las relaciones entre ellos. Los casos de uso respetan el estándar establecido por Rational comentado en la sección XX. De igual forma, los early aspect respetarán el formato definido en el Aspect Extract Tool para la técnica de reconocimiento semántico.

La técnica propuesta, a partir de esta información, crea las entidades Caso de Uso y Early Aspects con los cuál trabajará el algoritmo, basándose además en las relaciones entre estas entidades para agruparlos inicialmente en el Quality Attribute Theme.

Un Quality Attribute Theme (QAT), básicamente está formado por un subconjunto de la especificación de casos de uso y un early aspect, que en forma conjunta hacen referencia a un atributo de calidad. En esta etapa del proceso, se completará la información de los casos de uso y early aspect, mientras se dejara pendiente instanciar la información del atributo de calidad. Esta información será justamente la que retorne el algoritmo.

Para este análisis en necesario realizar en primer lugar un pre procesamiento de los datos y luego un filtrado sobre los mismos. En estas actividades, se logrará determinar un subconjunto del conjunto de palabras que forman la especificación de los casos de uso y la definición del early aspect, las cuales serán relevantes para la identificación del atributo de calidad involucrado.

***IV.1.1 Pre Procesamiento de los datos***

El objetivo de esta tarea es procesar la entrada para dejar la información en un formato uniforme antes de que el algoritmo sea invocado. Este pre procesamiento es realizado a través de la separación de la información de las entidades de entrada en *tokens.* Sobre estos tokens se aplicará luego una secuencia de filtros para transformar las palabras de los casos de uso y early aspects a un formato esperado por el algoritmo. Se considera un *token* a una unidad básica de texto que puede ser enriquecida con diferentes atributos, como por ejemplo, peso, ocurrencias, etc. El resultado de este pre procesamiento es una lista de tokens para las palabras obtenidas de los casos de uso y otra lista de tokens para las palabras del early aspect.

Lista Tokens Early Aspects

Lista Tokens Casos de Uso

Procesamiento de los datos de Entrada

Early Aspects

Casos de Uso

Por ejemplo, considerando la siguiente fracción de un caso de uso:

|  |  |
| --- | --- |
| Nombre | Registrar Usuario |
| Descripción | Se registran los datos del el usuario en el sistema. |

Se genera la siguiente lista de tokens:

**IV.1.2 Filtrado de los datos**

Como se mencionó en la sección anterior, el filtrado de los datos es otra de las tareas que se realiza sobre la entrada. En este caso, se ejecutan una serie de filtros sobre los tokens de las listas de casos de uso y early aspects. Estos filtros, además, serán utilizados luego para el formateo de las palabras de la ontología.

Un filtro es una unidad de procesamiento que realiza una modificación (enriquece, refina o transforma) sobre los datos de entrada y los copia a la salida para que otro filtro trabaje sobre los mismos datos. Mediante los filtros podemos realizar transformaciones independientes sobre el flujo de datos.

El patrón de arquitectura de tubos y filtros provee una estructura para procesar flujos de datos. Cada paso de procesamiento se encapsula en un filtro. Los datos se pasan usando los tubos entre filtros adyacentes. Recombinando los filtros se pueden construir distintas familias de sistemas relacionados

Ventajas:

* futuros cambios intercambiando algunos filtros;
* pequeñas transformaciones son más fácilmente reutilizadas;
* los datos pueden tener diferentes formatos.

En este caso, la entrada para los filtros es una lista de tokens, por lo que el filtro ejecutará acciones sobre los tokens de la lista (modificar la palabra, agregar atributos), para luego devolver la lista de tokens modificada.

**IV.1.2.1 Filtro Lower Case**

Filtro encargado de pasar las palabras a minúsculas.

Para el ejemplo anterior, solo se modificarían los tokens que su palabra tenga alguna mayúscula.

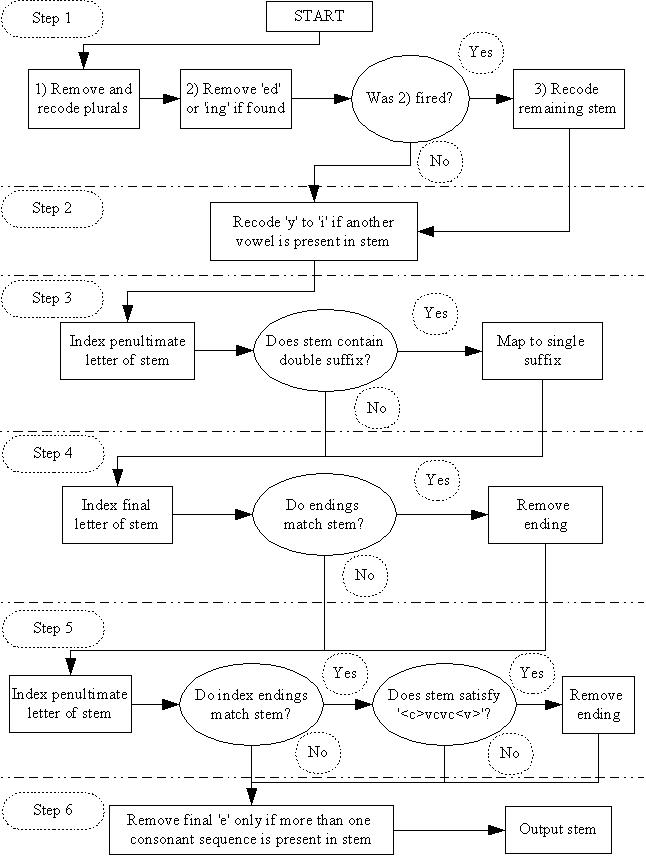
**IV.1.2.2 Filtro Stop Words**

Se eliminan las denominadas *stop-words*, que son palabras qué, desde el punto de vista no lingüístico, no contienen información relevante. Las stop-words difieren de acuerdo al lenguaje, en el caso del idioma Español podemos mencionar artículos, preposiciones, etc. (por ejemplo: a, de, del, el, en, ella, los, se).

**IV.2.1.3 Filtro Stemming**

Dado que diferentes variantes de una misma palabra pueden ser problemáticas al realizar análisis de texto porque tienen diferente deletreo pero el mismo significado, por ejemplo: aprender, aprenden, aprendió, aprender, se decidió utilizar *stemming*, que es el proceso de transformar una palabra en su *raíz* (stem). Para tal ejemplo, todas esas palabras estarían identificadas por su raíz, es decir “aprend”. Con el Stemming se aumenta el recall sobre el número de palabras que se pueden encontrar en una consulta a la ontología. Por ejemplo, una consulta sobre el término “bibliotecas” encontrará instancias en las que solo aparezca “bibliotecario” ya que el stem de las dos palabras es el mismo (”bibliotec”).

Para realizar el stemming se utiliza uno de los algoritmos más comunes y ampliamente usados, el algoritmo de Porter.



*Diagrama XX. Algoritmo de Porter.*

Continuando con el ejemplo anterior, se llevarían las palabras de los tokens a su raíz:

**IV.1.2.3 Filtro Ocurrencias**

Este algoritmo elimina tokens duplicados en la lista y enriquece cada token con el número de ocurrencias. Para los tokens obtenidos de los casos de uso se considera un token duplicado en caso de que el token sea la misma palabra y además aparezca en la misma sección.

**IV.1.2.4 Filtro Pesos**

Este filtro es aplicado para enriquecer los tokens de la lista de tokens de casos de uso asignando un peso a cada token. Este peso depende de la sección donde aparece la palabra. Por ejemplo se asignará un peso alto a una palabra que pertenece al nombre o descripción del caso de uso, mientras que se asignará un peso más bajo a una palabra perteneciente a un flujo alternativo.

Siguiendo el ejemplo, y suponiendo que para la sección “nombre” se asigna un peso de 5 y para la sección “descripción” un peso de 4, la lista de tokens final quedaría de la siguiente forma:

Los filtros ejecutados sobre la lista de tokens de Casos de Uso son:

* Lower Case
* Stop Words
* Stemming
* Ocurrencias
* Pesos

*Diagrama XX. Filtros utilizados para los Casos de Usos*

Para el caso de los Early Aspect no es necesario aplicar el filtro de peso debido a que se considera que todas las palabras dentro del early aspect tienen el mismo peso para el algoritmo. Esta diferencia radica en que el caso de uso está dividido en secciones y claramente hay secciones que deben ser ponderadas sobre otras.

De igual forma a la hora de formatear la ontología no se correrán todos los filtros sobre las palabras de la misma. Solo será necesario correr los tres primeros filtros para dejar las palabras pertenecientes a la ontología en el mismo formato que las listas de tokens.