Capítulo 1

Microplanning.

En éste capítulo describiremos las tres tareas involucradas en la etapa de microplanning: lexicalización, agregación y generación de expresiones de referencia. Luego definiremos en detalle la entrada y salida de esta etapa. Finalmente profundizaremos sobre la *lexicalización*, que será la única de las tareas antes mencionadas realizada por nuestro mircroplanner.

1.1. Tareas del Microplanner

La tarea del microplanner será tomar el document plan generado en la etapa anterior y refinarlo a modo de producir una especificación mas detallada del texto. Cabe aclarar que el resultado de esta etapa no será todavía un texto, sino que quedarán por tomar decisiones acerca de la sintaxis, morfología y cuestiones de presentación, de las cuales se encargará el *realizador de superficie*.

Como mencionamos en el capítulo **??**, las tareas que debería realizar el microplanner son:

Lexicalización. Esta tarea se encargará de elegir que palabras particulares, constructores sintácticos y anotaciones de "mark-up" a usar para comunicar la información contenida en el document plan. Desarrollaremos más en detalle el trabajo realizado por esta etapa en la sección 1.3

Agregación. Esta tarea deberá combinar elementos informativos con el fin de conseguir un texto más fluido y legible. La agregación decide que elementos se pueden agrupar para generar oraciones mas complejas sin modificar el significado de las mismas. Por Ejemplo, dos frases de una descripción para una clase de prueba de un scheduler se podrían expresar como:

- *"El proceso a borrar se encuentra en la tabla de procesos. El estado del proceso a borrar es waiting."*
- *"El proceso a borrar se encuentra en la tabla de procesos y el estado del mismo es waiting."*

Decidimos para este trabajo expresar muestras descripciones siguiendo el estilo de la primer frase del ejemplo anterior, es por esto, que como mencionamos anteriormente, nuestro microplanner no realizará tareas de agregación. Para este trabajo en particular creemos que es útil para el lector que cada frase de nuestra descripción haga referencia a una única restricción del esquema de la clase de prueba. De esta forma podríamos identificar con mayor facilidad cual es la descripción para una expresión particular de la clase de prueba.

Generación de expresiones de referencia. Esta tarea se encarga de determinar que frases deben ser usadas para identificar las diferentes menciones al mismo elemento en un texto a fin de aportar fluidez al mismo. Por ejemplo, en los casos que se hace referencia a una entidad que ya ha aparecido en el texto (referencia posterior) se puede remplazar la misma por otra frase que la referencie (generalmente un sintagma nominal). La elección de que expresión usar para referirse a la entidad dependerá del contexto y deberá hacerse sin generar ambigüedad para el lector. Por ejemplo, siguiendo con el ejemplo anterior del scheduler, podríamos reemplazar la segunda ocurrencia de "el proceso a borrar" en la primer frase por el pronombre "mismo", quedando entonces:

"El proceso a borrar se encuentra en la tabla de procesos. El estado del mismo es waiting."

Como mencionamos anteriormente, nuestro microplanner no realizará tareas de generación de expresiones de referencia ya que se encuentran fuera del alcance de este trabajo. Además, como podemos ver en el *corpus* nuestras descripciones de clases de prueba están formadas por una serie de oraciones individuales, donde cada una de estas describe una restricción de la clase de prueba dada. Estas oraciones resultan relativamente concisas y es extraño que hagan referencia en más de una oportunidad a un elemento, por lo tanto creemos que en este caso no resulta indispensable contar con un generador de expresiones de referencia.

1.2. Entrada y salida del microplanner

Como dijimos en el capítulo anterior, la salida del document planner será una estructura donde se encuentran agrupados los elementos informativos que deseamos comunicar. Estos elementos informativos (o *mensajes*) definidos en el capítulo anterior especifican de una manera abstracta qué debemos comunicar en el texto final, pero no especifica, por ejemplo, que palabras debemos usar para hacerlo. Será el microplanner quien deberá tomar esta estructura, el document plan, y producir una especificación mas refinada del texto que deseamos generar.

Esta especificación del texto, construida a partir del document plan, tendrá también una estructura de árbol donde los hojas especificaran las frases u oraciones a generar (*phrase specification*) y los nodos internos (*text specification*) establecerán cómo estas frases tendrán que ser agrupadas en elementos del documento como párrafos, secciones, lista de items, etc. Luego, será tarea de la etapa de realización convertir los nodos internos en anotaciones especificas para el sistema de presentación (realización de estructura) y transformar las *phrase specification* en oraciones o frases sintáctica, morfológica y ortográficamente correctas (realización lingüística).

En nuestro caso contaremos con sólo dos elementos para modelar las estructura interna del documento: *TSDocumento* y *TSListaItems*. El primero especificará la raíz del documento final y contendrá un conjunto de TSListaItems que modelan la lista de descripciones a generar para cada clase de prueba.

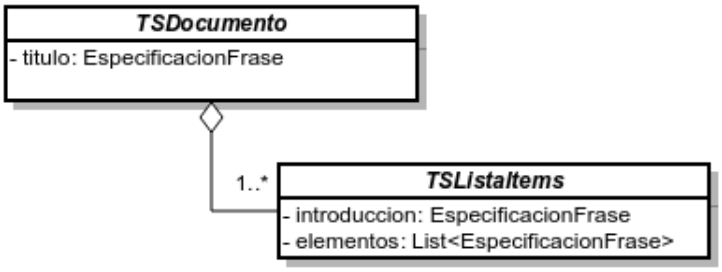

Figura 1.1: Text Specification.

Luego en la etapa de *realización de estructura* se deberán transformar estas estructuras en anotaciones para el sistema de presentación.

En la literatura sobre NLG Podemos encontrar muchas alternativas en lo que respecta a la especificación de frases. Todas estas varían en el nivel de abstracción que tienen. Las representaciones mas abstractas le darán mas flexibilidad a las etapas de document planning y microplanning, pero al mismo tiempo nos obligarán a tener un realizador de superficie mas sofisticado. Por otro lado, las especificaciones menos abstractas, requieren que el document planner y el microplanner realicen un mayor trabajo, pero también tendrán mas control sobre el texto a producir.

Uno de los objetivos que tuvimos a la hora de idear una estructura para especificación de frases fue que ésta sea independiente de nuestro problema, que hable en términos del lenguaje que queremos generar (español en este caso) y no en término de específicos de nuestro problema (podría referirse a elementos de Z en un caso muy abstracto). De esta forma podremos implementar un realizador de superficie que sea independiente de este problema puntual o bien hacer uso de alguno existente sin mayores complicaciones.

Para este trabajo optamos por usar árboles de sintaxis abstracta para especificar las frases a generar. Estos nos permitirán construir un realizador lingüístico independiente del dominio de aplicación de este problema en particular.

En la figura 1.2 podemos ver los elementos que utilizamos para modelar el subconjunto del lenguaje español que nos permitirá generar las descripciones introducidas en el capítulo **??**.

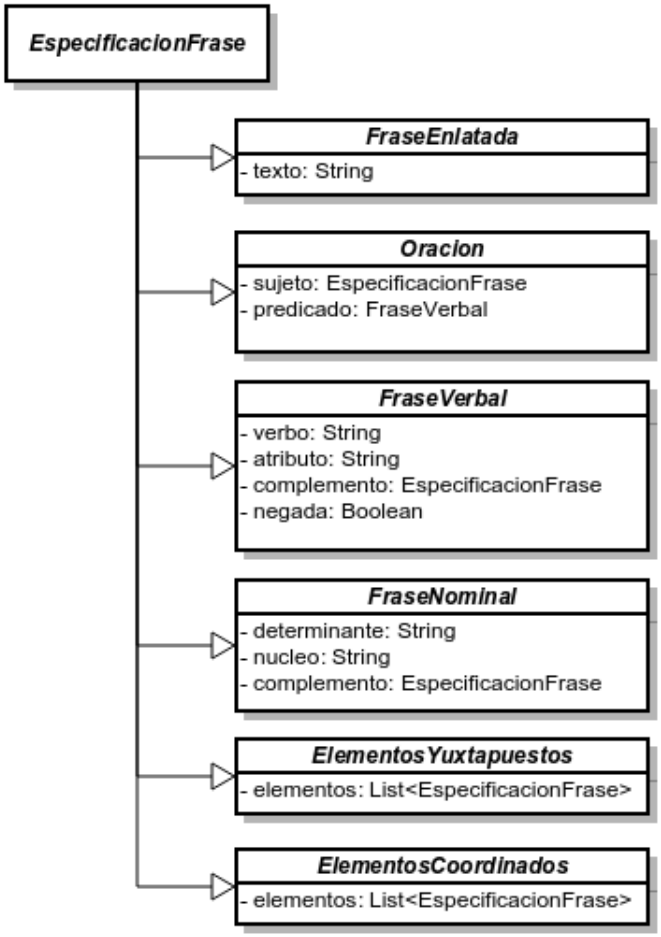

Figura 1.2: Phrase Specification.

La decisión de utilizar esta estructura esta muy relacionada con las tareas que necesitamos realizar en la etapa de realización de superficie. Por ejemplo, necesitamos que nuestro realizador sea capaz de identificar el verbo en una oración y conjugarlo de manera que haya una concordancia de numero entre sujeto y verbo. A continuación, en el capítulo 1.3 detallaremos como construir estas especificaciones a partir de los mensajes incluidos en el document plan

y luego profundizaremos más sobre estas especificaciones cuando tratemos la etapa de realización lingüística en el capítulo **??**.

1.3. Lexicalización

Como mencionamos anteriormente, el proceso de lexicalización será el encargado de elegir que palabras particulares, constructores sintácticos y anotaciones de "mark-up" usar para comunicar la información contenida en el document plan. En esta etapa deberemos producir una especificación de frase para cada mensaje contenido en el document plan. En nuestro caso en particular debemos hacerlo en base a las reglas definidas anteriormente en la sección **??**.

El módulo encargado de esta tarea en nuestro sistema deberá ser capaz de generar una especificación de frase a partir de la expresión Z contenida en un mensaje del document plan. Primero deberá verificar si la expresión en cuestión se encuentra designada, en este caso, deberá construir una especificación de frase en base a su designación. De lo contrario deberá intentar construirla recursivamente de acuerdo a las reglas antes mencionadas. En la figura 1.3 podemos ver un bosquejo del algoritmo que deberá realizar el lexicalizador. Este algoritmo es sólo para ilustrar el comportamiento deseado durante esta etapa, la implementación de este, entre otras cosas, deberá construir una EspecificacionFrase más compleja que la concatenación que podemos observar en el algoritmo.

```
lexicalizacion exp = if estaDesignada exp
                     then designacion exp
                     else lexicalizacion' exp

lexicalizacion' {x} \eq y  = lexicalizacion x ++
                             "es el único elemento de" ++
                             lexicalizacion y
lexicalizacion'  x  \eq {} = "no hay elementos en" ++
                             lexicalizacion y
lexicalizacion'  x  \eq y  = lexicalizacion x ++
                             "es(son) igual(es) a" ++
                             lexicalizacion y
...
```

La función *designacion* deberá ser capaz de construir una especificación de frase a partir de una expresión designada[1]. Para esto deberá recuperar el texto

[1]Escribimos esta función de esta forma para simplificar, pero la implementación de la misma es un poco mas compleja en realidad. Además de la expresión deberíamos saber el nombre del esquema al que pertenece la expresión ya que podría tratarse de una variable de

de esta designacion y procesarlo a fin de crear una especificación de frase. Es razonable suponer que el texto de una designación sea un sintagma nominal, es decir tenga la siguiente estructura:

$$\text{Sintagma Nominal} = [\text{Determinante}] + \text{Núcleo} + [\text{Complemento}]$$

Por lo tanto nuestro sistema deberá trabajar el texto de las designaciones (haciendo uso de algún analizador morfológico) para construir una FraseNominal a partir de cada designación.

Por otro lado, en el caso que la expresión a lexicalizar no se encuentre designada, se deberá analizar recursivamente la expresión para generar el texto adecuado según las reglas antes mencionadas. Tomemos por ejemplo el primer mensaje del el document plan de la figura **??**. Siendo la expresión a verbalizar:

$$s? \in \text{dom } st$$

y contamos con las siguientes designaciones:

$s?$	$\approx$ el símbolo a buscar
$dom\ x$	$\approx$ símbolos cargados en la tabla de símbolos

En este caso, al no encontrarse designada $s? \in \text{dom } st$ nuestro lexicalizador deberá construir una *Oracion* para la frase:

$$\text{lexicalizacion } s? \text{ ++ "pertenece a-+ lexicalizacion dom } st$$

Para esto deberá resolver recursivamente la lexicalizacion de $s?$ y de dom st para formar el *sujeto* y *predicado* necesarios para la oración. Luego deberá formar una *FraseVerbal* a fin de completar tanto el *verbo* como *complemento* a fin de obtener el texto "pertenece a" en el texto final. En la figura 1.3 podemos observar la especificación de frase resultado de la lexicalización.

---

esquema por ejemplo. Otro caso particular es el de las designaciones parametrizadas en el que deberá resolverse la designación del argumento primero para luego construir la designacion pretendida.

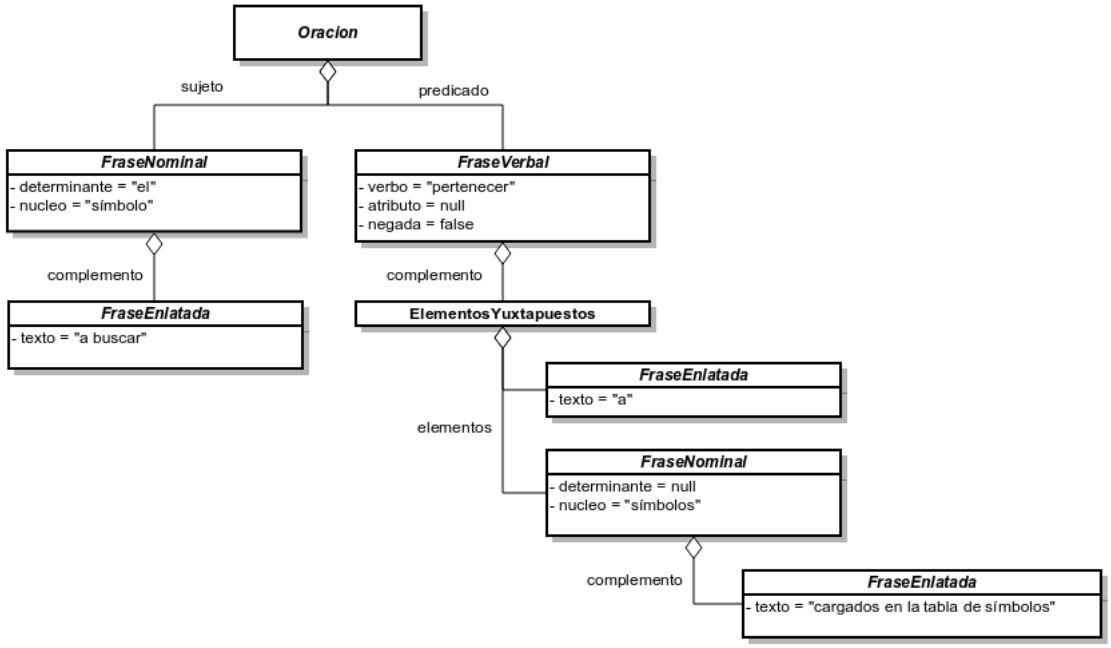

Figura 1.3: Phrase Specification para.

Cabe aclarar que el verbo se encuentra en infinitivo ya que será el realizador lingüístico el encargado de conjugar el mismo de acuerdo a algunas reglas gramaticales que veremos en el capítulo **??**. Otra cuestión a mencionar es la inclusión del elemento *ElementoYuxtapuestos* ideado para salvaguardar la falta de

que para nuestro trabajo, el realizador no necesitará información a cerca de la función sintáctica del complemento del sintagma nominal y del sintagma verbal