Capítulo 5

Modelización

5.1 Introducción

En éste trabajo de tesis buscamos establecer un modelo de acceso con un enfoque para documentos complejos utilizando las relaciones de estructura que guardan los componentes de éstos documentos, es decir su organización sintáctica. En el marco de nuestro proyecto de Tesis, no tenemos la intención de proporcionar ningún otro lenguaje de consultas, pero si nos propusimos orientar nuestro trabajo, hacia un enfoque en las funcionalidades que un lenguaje de consultas debe integrar y particularmente, en nuestro caso en los documentos estructurados. Consideremos el siguiente ejemplo, se trata de una consulta combinando los siguientes aspectos: a) especificaciones estructurales, b) especificaciones de fecha, y c) especificaciones sobre el contenido.

Las especificaciones estructurales, contienen restricciones sobre los tipos de elementos buscados, la composición de éstos elementos, y aún mas sus posiciones respectivas en la estructura sintáctica.

Las especificaciones de fecha, corresponden a restricciones sobre los atributos clásicos de las bases de datos, por ejemplo una fecha, un número de capítulo, etc.

Las especificaciones sobre el contenido, abordan específicamente la problemática de búsqueda de información permitiendo interrogar sobre el contenido del documento, es decir su contenido semántico, su temática, así como las características propias de cada media.

5.2 El modelo

En ésta sección presentaremos el modelo propuesto por Fourel [Fou98] con el objetivo de manipular los documentos estructurados, basado en el punto de vista de la sintaxis. A partir de ésta aclaración consideraremos tres tipos de modalidades de acceso: 1) el acceso a través de la estructura, 2) el acceso a través de los atributos, 3) el acceso a través del contenido semántico.

5.2.1 El acceso a través de la estructura

Esta modalidad implica el punto de vista de la estructura sintáctica, que toma en cuenta las tres relaciones de estructura:

a) **La relación de composición :** define la agregación de las diferentes partes documento. (c.f. figura 5.1),

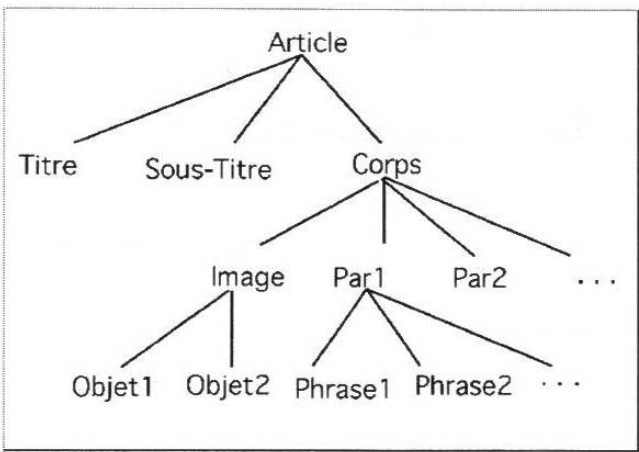
b) **La relación de secuencia :** determina la succesividad entre las diferentes partes del documento. (c.f. figura 5.2),

c) **La relación de referencia :** expresa las ligas entre las partes del documento. (c.f. figura 5.3).

Estas relaciones son las que se encuentran tradicionalmente dentro de los lenguajes de representación de documentos estructurados.

a) Relación de composición

Para ilustrar ésta relación consideraremos el ejemplo de la figura 5.1, de un documento con un cierto tipo de estructura. Artículo está compuesto por un {título, un sub-título y un cuerpo}, donde el cuerpo está compuesto por una {imagen, un Par1 y un Par2}, la imagen está compuesta por un {objeto1 y un objeto2}, y finalmente Par1 está compuesto por las {Frase1 y Frase2}.

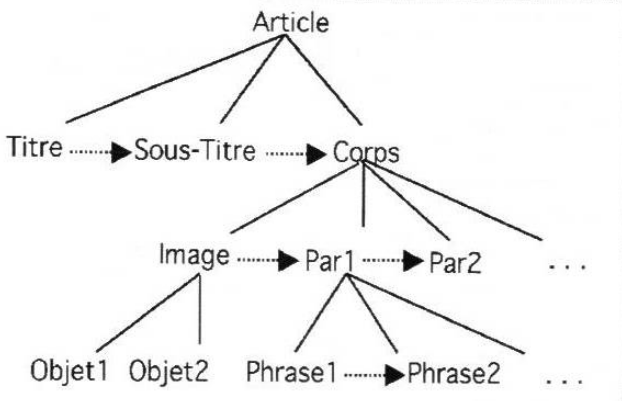


Au niveau de type de structure :
Type de structure: (TYPE, $\leq_{tcomp}$)
Article, titre, sous-titre, corps $\in$ TYPE
article $\leq_{tcomp}$ titre
article $\leq_{tcomp}$ sous-titre
article $\leq_{tcomp}$ corps
Au niveau de l'instance du document:
(OS, $<_{comp}$, type)
article, titre, sous-titre, corps $\in$ OS
article $<_{comp}$ titre
article $<_{comp}$ sous-titre
article $<_{comp}$ corps

Figura 5.1: relación de composición de un documento de tipo artículo

b) Relación de secuencia

La relación de secuencia describe la sucesión de las entidades, es decir su sentido en el momento de la lectura en el documento. En el ejemplo de la figura 5.2 podemos observar la relación de secuencia donde un título precede a un sub-título y el sub-título precede a un cuerpo.



Au niveau de type de structure :

Type de structure: (TYPE, $\leq_{tseq}$)

titre, sous-titre, corps $\in$ TYPE

titre $\leq_{tseq}$ sous-titre

titre $\leq_{tseq}$ corps

Au niveau de l'instance du document:

(OS, $\leq_{seq}$ type)

titre, sous-titre, corps $\in$ OS

titre $<_{seq}$ sous-titre

sous-titre $<_{seq}$ corps

Figure 5.2: Relación de secuencia de un documento de tipo artículo

c) Relación de referencia

Las relaciones de referencia describen las referencias entre elementos estructurales. En nuestro ejemplo de la figura 5.3 un sub-título hace referencia a un párrafo y éste a una primera frase y ésta última a una frase2 y finalmente ésta a un objeto1. Para definir formalmente **el tipo de documento**, lo expresaremos de la siguiente forma:

$$ST = (\text{TYPE}, \leq_{tcomp}, \leq_{tseq})$$

ST contiene el conjunto de tipos de elementos de elementos estructurales que forman el documento de notación *TYPE*, así como dos relaciones: la relación de composición de notación $\leq_{tcomp}$ que describe como los elementos estructurales de un cierto tipo pueden ser compuestos, y la relación de frecuencia de notación $\leq_{tseq}$ que describe como éstos elementos pueden encadenarse en la secuencia.

Figure 5.3: Relación de referencia de un documento de tipo articulo

Para el ejemplo de la figura 5.4 consideramos el tipo de estructura de los documentos de **tipo artículo** de notación ST_{article}. Este tipo hace parte del conjunto $TYPE_{\text{article}}$ compuesto por el artículo, el título, el sub-título, el cuerpo del artículo, la referencia, el párrafo, la imagen, el objeto, etc. Las relaciones que tienen influencia en la composición expresan por ejemplo que un elemento de tipo artículo puede estar compuesto por elementos de tipo título, sub-título, cuerpo del artículo o referencia.

$$\text{artículo} \leq_{\text{tcomp}} \text{título}, \text{artículo} \leq_{\text{tcomp}} \text{sub-título},$$
$$\text{artículo} \leq_{\text{tcomp}} \text{cuerpoartículo y artículo} \leq_{\text{tcomp}} \text{referencia}$$

Para describir **el documento**, y por consecuente las relaciones que organizan sus elementos de estructura, consideramos el « cinco-uplet » D ligado a un tipo de estructura ST por la relación ς.

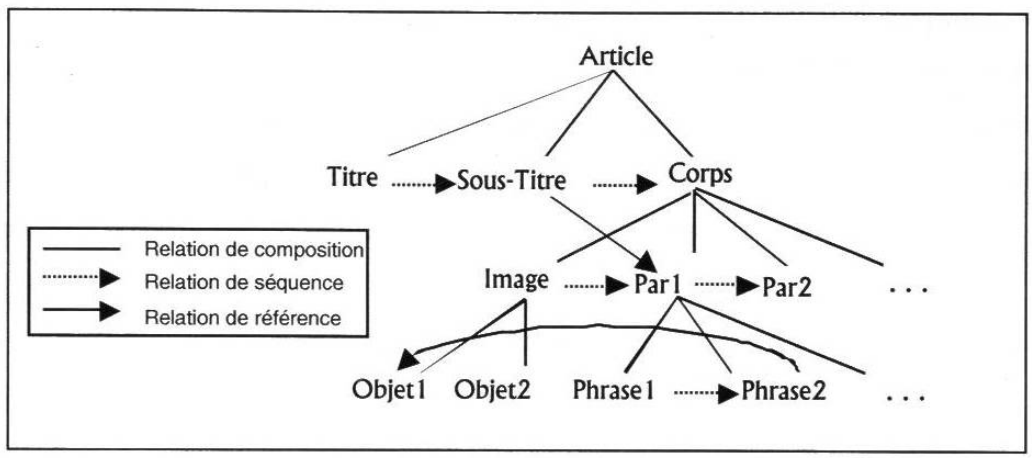


Figura 5.4: Organización sintáctica de un documento D_1 de tipo artículo

Para el documento D_1 de la figura 5.4, tenemos entonces, $\varsigma\,(TYPE_{\text{article}}, D_1)$:

$$D = (OS, <_{\text{comp}}, <_{\text{seq}}, <_{\text{ref}}, type)$$

El elemento OS describe el conjunto de elementos de la estructura que forma el documento D_1, las relaciones $<_{\text{comp}}$, $<_{\text{seq}}$, y $<_{\text{ref}}$ describiendo respectivamente las relaciones de composición, de secuencia y de referencia entre los elementos de OS. La función type permite asignar a cada uno de los elementos de OS un tipo tomado dentro del conjunto del tipo de estructura del documento ($TYPE_{\text{article}}$ de ST_{article}).

5.2.2 El acceso a través de los atributos

Consiste en tener acceso a un documento o a una de sus elementos especificando el tipo de información que le caracteriza, por ejemplo sus autores, su fecha de publicación, etc. La búsqueda por atributos requiere la presencia de atributos dentro del modelo del documento.

Retomaremos para ***la organización de los atributos*** el enfoque clásico propuesto para los lenguajes de representación de un documento estructurado (los elementos de la estructura), tomando en cuenta que la descripción de los elementos que componen el documento está guiada por el tipo de sus elementos. Generalmente, un atributo se describe por su nombre así como por su dominio de valores. Para los atributos de los documentos estructurados, además determinar cuales son los tipos de elementos de estructura susceptibles de aceptar un atributo.

Describimos de forma tradicional el triple:

A= (NOM, DOMAINE, domaine)

NOM define el conjunto de nombres de atributos, DOMAINE es el conjunto de los dominios de valores disponibles, y la función *domaine* asocia a cada nombre de atributo un dominio.

5.2.3 El acceso a través del contenido semántico

Este tipo de acceso necesita de una fase de indexación, es decir la explicitación y la representación del contenido semántico de los documentos. Es necesario entonces, para los documentos estructurados, indexar los elementos que componen el documento para que éstos puedan ser localizables. Esto consiste, en nuestro modelo, en utilizar la estructura original realizada por el autor, donde los elementos de tipo texto son indexados.

Organización del contenido semántico.

Dos relaciones son principalmente puestas en juego: la relación de dominio y la relación de intención para la primera relación: cada parte del documento corresponde a

un conjunto de temas, y la relación de dominio significa que éstos temas se agregaran en los elementos que lo componen.

Como ya lo hemos explicado, la representación de la información concerniente a cada uno de los elementos de la estructura es realizada a través de los atributos. Para ello, representaremos de manera similar el contenido semántico de éstos elementos por un atributo en particular, destacando como característica el depender de las relaciones de composición y de referencia. Esta dependencia permite ampliar la organización semántica del documento.

5.3 Cobertura y dependencias.

A partir del documento inicial de la figura 5.5, encontramos diversos problemas. [Fou98].

1) la presencia de información implícita en el documento inicial.
2) los elementos accesibles vía consultas.
3) las ligas entre los valores de los atributos.

La presencia de información implícita. Si consideramos que el atributo "status" definido para el elemento Par1 tiene por valor "borrador", intuitivamente pensaremos que los elementos **Artículo** y **CuerpoArtículo** se encuentran también dentro del estatus de "borrador", o que por lo menos uno de sus componentes tiene por estatus "borrador". Lo que significa que ésta información, que puede ser deducida a partir del documento inicial, no está representada por D_1. Juzgamos pertinente hacer notar la importancia de proporcionar explícitamente la información presentada de manera implícita para el documento inicial.

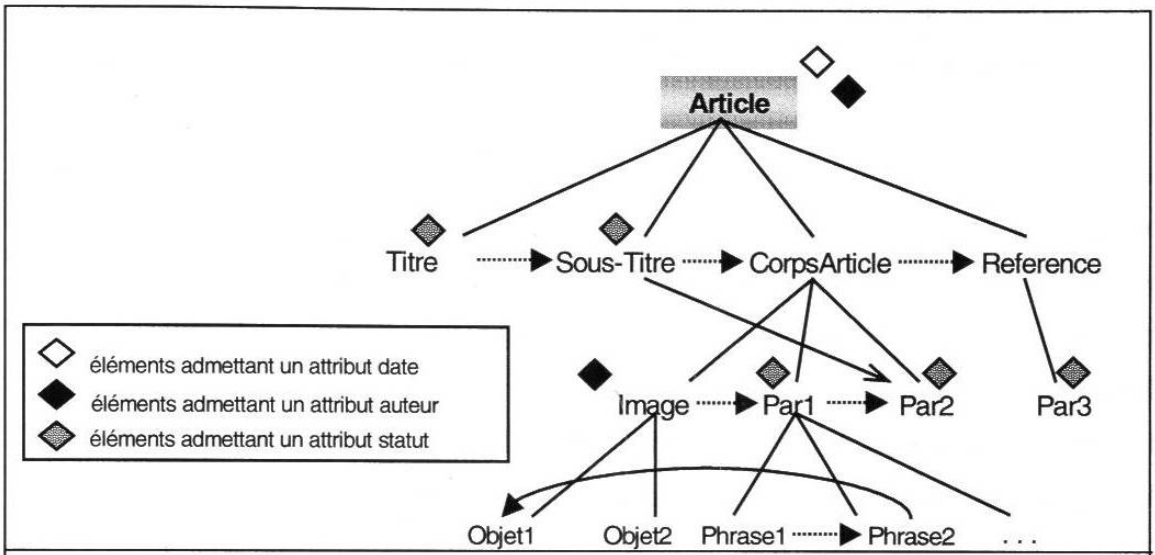


Figure 5.5: Organización de los atributos para el documento D

Los elementos accesibles. Si tomamos como ejemplo la figura 5.5 y deseáramos formular una consulta que comprenda una conjunción de criterios sobre los atributos tales como: `autor, fecha, y estatus;` el documento D_1 no contiene los elementos necesarios para ser descritos simultáneamente, y por lo tanto no será posible dar una respuesta a ésta consulta, aunque los tres atributos estén presentes en el documento.

Relaciones entre valores y atributos. Hemos visto que con la organización semántica se crean dependencias explícitas entre las descripciones semánticas de los elementos de estructura, éstas están basadas sobre las relaciones de estructura de la organización sintáctica. En el ejemplo del documento inicial D_1 éstas dependencias no son consideradas, y ésta es la motivación para crearlas utilizando las relaciones de estructura de la organización sintáctica, como ser la relación de composición, la relación de secuencia y la relación de referencia.

Deseamos creas dependencias para los atributos del contenido semántico así como para los demás (autor, fecha, estatus, etc). Ahora describiremos con mas precisión los elementos requeridos para tratar una consulta para los documentos estructurados. Esta

necesidad nos lleva a concentrarnos en dos nociones primordiales: **la cobertura y la dependencia** de los atributos que corresponden a dos procedimientos diferentes:

d) El primer procedimiento consiste en crear atributos para transformar las informaciones implícitas en informaciones explícitas y definir claramente los elementos accesibles para cada atributo. Definimos entonces, *la cobertura de un atributo definido sobre un elemento* como sobre el conjunto de los elementos concernidos por un atributo en particular. Dicho de otra forma, la cobertura total expresa la adecuación entre el conjunto de las informaciones que describen el documento y el conjunto de los elementos de estructura que representan el documento.

e) El segundo proceso consiste en determinar por un lado como seran valorizados los atributos creados y de donde provendran sus valores, y por otro lado en explicitar las dependencias entre los valores de los atributos en documento estructurado.

Para ello la noción de alcance de los atributos es definida en un documento con el objetivo de responder a éstas dos nociones.

5.4 El alcance de los atributos

El alcance de los atributos en el documento estructurado permite definir las partes del documento concernidas por un atributo, es decir que éste define los elementos que poseen las mismas propiedades. Además, el alcance haces que las dependencias sean explícitas entre los valores de los atributos en el seno de éste documento.

El alcance de un atributo α describe un elemento estructural o de un documento inicial D concernido por el valor del atributo α del elemento o. Los elementos de éste conjunto (cobertura de un atributo α del elemento o) comparten una misma fuente de información, el atributo α del elemento estructural o y su valor.

El alcance de un atributo α sobre un elemento de estructura o es definido según la relación sintáctica (composición, secuencia, referencia), y según una categoría de propagación que indica el sentido que la relación sintáctica deberá seguir. El conjunto que representa la cobertura de un atributo está construido utilizando la relación sintáctica y una categoría de propagación.

Sea un valor inicial del atributo α, es decir el valor fuente que será compartido por los atributos α de los elementos estructurales del alcance, la función de propagación determina el comportamiento entre los valores iniciales y la relación sintáctica. Esta función permite destinar un valor a los atributos α de los elementos del alcance.

Podemos expresar el conjunto de las condiciones que deben llenar los elementos estructurales para pertenecer al alcance. Esto permite, por ejemplo, especificar que un atributo no concierne mas que un cierto tipo de elemento de estructura, o bien que la profundidad del alcance es limitada. A través de éstas condiciones que se puede, por ejemplo controlar los elementos que admitirán un atributo de contenido semántico.

La definición del alcance de un atributo explícita la cobertura éste atributo así como los valores que serán asignados a los elementos de ésta cobertura. A partir de ésta definición varias clases de atributos pueden ser definidas según las relaciones de estructura utilizadas y según la categoría de propagación:

(1) **atributo estático**: se trata de los atributos tradicionales de los SGBD. La información concierne únicamente al element de estructura al cual el atributo está asociado (cat = static).

(2) **atributo de composición ascendiente** ***(descendiente)*** : el atributo y su valor se propagan **según la relación de composición** hacia los predecesores (sucesores) del elemento estructural. El atributo de contenido

semantico y el atributo estatus son atributos de composición acsendientes, mientras que los atributos fecha y autor son descendientes.

(3) **atributo de secuencia hacia atrás *(antes)***: el atributo y su valor se propagan según **la relación de secuencia** hacia los predecesores (sucesores) del elemento estructural.

(4) **atributo de referencia**: El atributo y su valor se propagan según **la relación de referencia** y exclusivamente hacia el sucesor del elemento estructural. Toda clase de atributos pueden ser clasificados en ésta categoría : **contenido semantico**, estatus, fecha, autor, etc.

Hemos definido el alcance para un atributo y para un elemento estructural. Ahora debemos considerar el conjunto de los alcances de los atributos de todos los elementos estructurales del documento D inicialmente definido por un atributo α, el alcance de α para cada uno de éstos elementos y finalmente una función de combinación, "$combine_{\alpha}$"

La presencia de ésta función de combinación se explica por hecho de que: i) cada elemento de estructura admite un atributo α único, y ii) Es posible que un elemento de estructura de un documento pertenezca a varios alcances de un mismo atributo. Es entonces necesario combinar las informaciones que provienen de éstas múltiples fuentes con el fin de obtener un valor único para el atributo del elemento concernido.

La función de combinación "$combine_{\alpha}$" calcula, a partir de un conjunto de valores del atributo α, el valor que el atributo α recibirá del elemento sobre el cual existe un conflicto. Permite en especial, el generar una dependencia entre el valor calculado y los valores provenientes de los elementos que son la fuente de la información.

5.5 Proceso de propagación de los alcances

Consideremos el atributo de contenido semántico definido sobre los elementos de tipo **artículo**. La descripción del alcance de éste atributo de composición ascendiente sobre el elemento **imagen** es la siguiente:

Image

$P_{contenu, comp} = (\texttt{contenu, image, comp, pred, image.contenu, } id_{contenu}, 0)$

La anotación **Image.contenu** representa el valor original del atributo de contenido semántico sobre el elemento imagen. La función de propagación $id_{contenu}$ corresponde a la identidad, es decir que los elementos del alcance reciben las informaciones contenidas en **Image.contenu**. No consideramos las condiciones de pertenencia. Por éste alcance, sabemos a partir de éste momento que los elementos **CorpsAticle** y **Articlе** del documento D_1 admiten un atributo de contenido semántico y reciben las informaciones que provienen de la descripción semántica de la imagen.

La primera etapa del proceso es el crear atributos de contenido semántico sobre los predecesores de éstos tipos de elementos. Para el documento de la figura 5.6, se crean atributos sobre los elementos **artículo, CorpsArticle y Reference.**

La segunda etapa consistirá en destinar valores a éstos nuevos atributos utilizando las funciones de propagación y de combinación. Es evidente que las funciones dependen del lenguaje de indexación utilizado. Si consideramos como ejemplo que el atributo de contenido semántico tiene por valor un conjunto de palabras claves, la función de combinación es entonces el operador de unión:

Corps-Article.contenu = Image.contenu ∪ Par1.contenu ∪ Par2.contenu
Reference.contenu = Par3.contenu
Article.contenu = Titre.contenu ∪ Sous-titre.contenu
∪ CorpsArticle.contenu ∪ Reference.contenu

Una relación de inclusión entre los valores de los atributos de contenido semántico resulta de la aplicación del operador de unión como lo indicamos en la figura 5.6

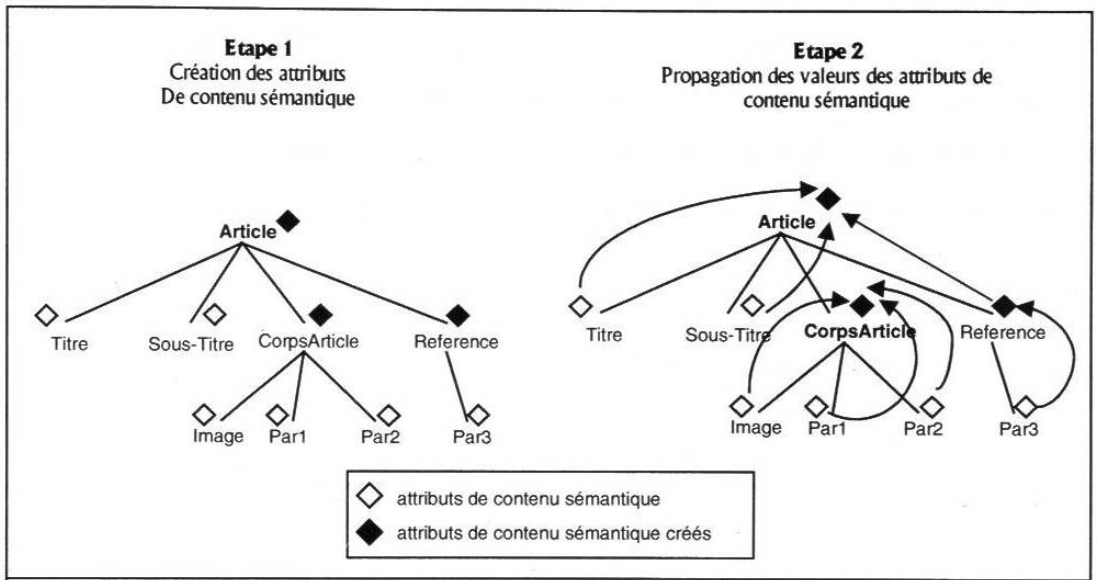


Figura 5.6: Aplicación de los alcances del atributo de contenido semantico

La aplicación de los alcances se establece entonces en dos etapas. La primera consiste en representar la cobertura, es decir en crear los atributos en función de del alcance. La segunda etapa pone en valor los atributos creados. Esta valorización se hace ya sea por la función de propagación si no hay conflictos, ya sea por la función de combinación cuando resolvemos los conflictos.

5.6 Indexación estructural

El alcance corresponde a una herencia de propiedades, pero contrariamente a la herencia de modelos con objetos que se hacen la relación IS-A, la herencia es más compleja ya que se encuentra en función de las relaciones de estructura de la organización sintáctica. Además la herencia se completa por las funciones de propagación y de combinación sobre los valores de los atributos.

Con el fin de saber mas sobre el proceso de indexación estructural, debemos voltearnos hacia las consecuencias de la aplicación de los alcances con relación al modelo inicial. En el modelo inicial, especificamos los tipos de elementos que pueden recibir los atributos. Con la aplicación de los alcances fuimos mas allá de éstas especificaciones y éstas ya son verificadas por consecuente. El documento final ya no es conforme al modelo inicial y para garantizar la coherencia del documento inicial debemos presentar un modelo con el cual sea conforme el documento que resulte de la indexación estructural.

El proceso completo de indexación estructural toma la forma que representamos en la figura 5.7, y se hace en dos etapas:

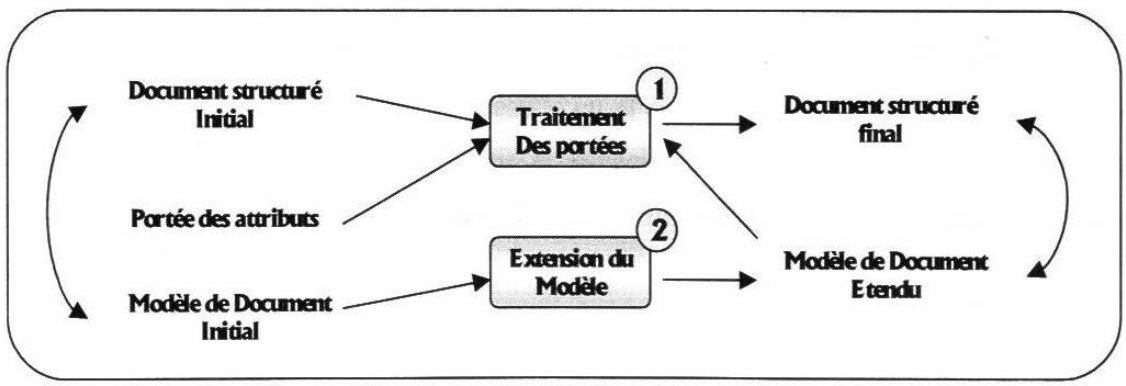


Figura 5.7: Proceso completo de indexación estructural

Etapa 1: realizar la extensión del modelo de documento estructurado inicial utilizando las definiciones de los alcances de los atributos, luego generar reglas al nivel de los tipos de elementos de estructura y extender la relación $\Re_{ST}$ de manera que el documento que resulta de la aplicación de los alcances sea conforme al modelo de documento extendido.

Etapa 2: aplicar los alcances de los atributos, es decir la creación de los atributos y la propagación de los valores. Se trata, como lo dijimos anteriormente, de resolver los conflictos que se presentan y aplicar las funciones de propagación y de combinación con el fin de generar dependencias entre los valores de los atributos.

El resultado de éste proceso de indexación estructural conlleva dos principales aportes para la investigación de documentos estructurados:

a) aumentar la cobertura del documento, es decir la información implícita en el documento inicial fue explicitada, podemos entonces utilizarla durante el proceso de investigación.
b) introducir dependencias entre los valores de los atributos, útiles para el proceso de investigación.

5.7 Implementación y contribuciones al modelo

Primeramente describiremos el uso de una tabla de "bits" con dos niveles en conjunción con un archivo inverso (basado en el enfoque B-árbol) tomando la forma de vectores de bits con el fin de reducir el espacio de búsqueda y asi poder aumentar considerablemente el desempeñó de búsqueda, es decir una recuperación más efectiva de los documentos estructurados.

La implementación está basada en el enfoque del archivo inverso, un enfoque clásico utilizado a menudo por los SRIs para indexar las bases de datos. La implementación de los dos niveles de representación tiene por objetivo el reducir lo más posible el espacio de búsqueda. Consideremos la consulta siguiente: A=a1 AND B=b1, y un documento estructurado D. Si uno de los valores de los atributos requeridos no se encuentra en ninguna de las partes del documento, éste es eliminado en el primer nivel, sin considerar cuales sub-partes de D caracterizan éstos atributos.

5.7.2 La organización de los atributos

Para cada atributo dinámico consideramos una tabla normalizada compuesta de dos campos: el primero representa el valor del atributo (O,1) y el segundo es un conjunto

de señaladores que representan un valor para los documentos estructurados. Incluso si consideramos que cualquier parte del documento puede ser una respuesta potencial los valores del B-árbol no harán ninguna referencia para todos los elementos que tengan un valor de atributo. La cantidad de elementos es grande a comparación con el número de documentos (mas de 400 elementos por documento sobre nuestra colección de documentos). De hecho, todos los documentos están señalados, si al menos, una de sus partes contiene un señalador al valor de un atributo.

La primera fila permite filtrar potencialmente los documentos pertinentes sin tomar en cuenta cual de las sub-partes son pertinentes para el proceso de búsqueda. Entonces consideramos un segundo nivel en el cual las partes del documento están representadas, es decir que para cada documento encontrado tendremos un conjunto de señaladores para representar las partes que contienen el valor del atributo buscado.

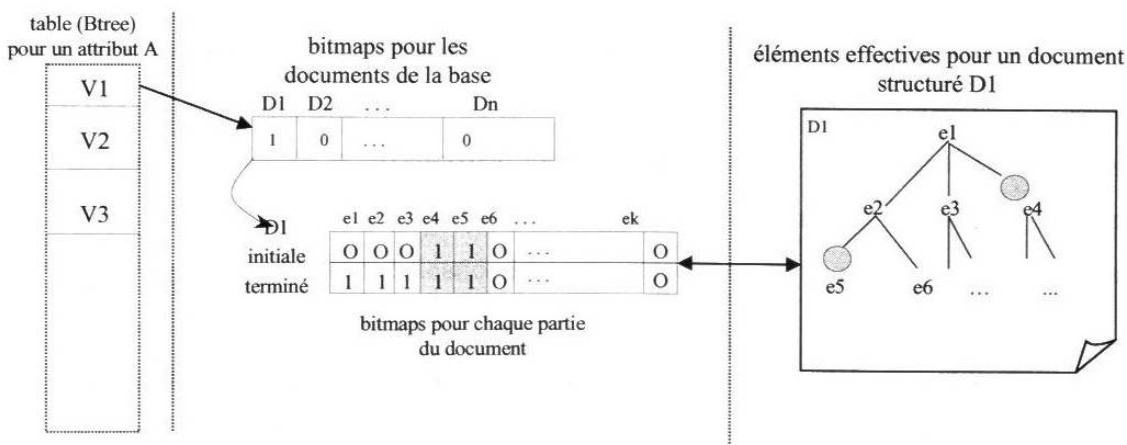


Figura 5.8:*Implementación de la representación de los atributos*

En la figura 5.8 presentamos como implementamos un cuadro de correspondencia basado en el enfoque B-Arbol por un atributo α, donde cada elemento B-árbol tiene un valor de referencia para todos los documentos. En el primer nivel de representación los documentos están representados por O's y 1's es decir por "bits": el bit `ith` del bit es 1 (de notación $S_{A=V1}[i] = 1$) si el documento `Di` está compuesto por lo menos de una de sus

partes, las describiremos por el valor del atributo v1. en el segundo nivel de representación de los bits, cada parte del documento está representada por el bit `jth` del bitmap de notación ($S_{A=v1}[i].D_{initial}[j]=1$), si la parte `ej` de éste documento está descrita por el valor v1. Podemos observar en la figura 5.8 que el cuadro está construido por dos niveles de "bits", el primero llamado inicial, indica cuales partes tienen un valor de referencia en el documento estructurado mientras que el segundo nivel, llamado completo muestra el estatus final obtenido después del proceso de los alcances.

5.7.3 Acceso a través de consultas

Considerando la combinación de accesos: el acceso via consulta y el acceso via navegación (anconsejable frente a grandes colecciones de documentos estructurados). En nuestro prototipo los dos modos de acceso fuéron retomados en base a los resultados del proceso de indexación estructural.

Las funcionalidades implementadas en nuestro prototipo son las siguientes:

a) **consultas sobre la estructura:** éstas son tratadas de manera clasica, a través de rutas de acceso dentro de la estructura que permiten acceder a cualquier elemento ya sea especificando su tipo, ya sea usando una de las tres relaciones de la organización sintáctica.
b) **consultas sobre los atributos:** de acuerdo a nuestro modelo éste tipo de consultas caracterizan a los SGBD ya que los atributos se utilizan para describir los elementos de la estructura así como los del contenido semántico. En el caso de los atributos no estáticos, el tratamiento se efectúa con la ayuda de las dependencias generadas por el porceso de indexación estructural.

5.7.4 Acceso a través de la navegación

La consulta por navigación permite al usuario de "hojear" toda la base de documentos.

a) **navegación dentro de la estructura :** para esta navegacion utilizaremos las tres relaciones de la organización sintáctica para poder ofrecer un acceso equivalente a una liga de tipo hipertexto. Consideraremos las relaciones de secuencia y de referencia para construir éstas ligas hipertexto.
b) **navegación dentro de las respuestas :** puésto que el objetivo es construir ligas hipertexto entre los elementos pertinentes del documento, utilizamos la estructura original de los documentos. El usuario nevega en la base a través de ligas originales sguiendo las ligas entre los artículos. Las ligas consruidas en éste proceso de corresponcia proveen una estructura de navegación dentro de las respuestas.

5.8 Otras contribuciones

Entre las técnicas de optimisación de las consultas como son : la reinscripción algebraica, la utilización de los índices texto completo (en inglés "full text") y finalmente la estructuración de esquemas, preferimos la utilización de índices con el fin de reducir el espacio de búsqueda y proveer una recuperación mas efectiva de los documentos estructurados. Hemos implementado por consecuente el enfoque B-árbol.

Una de nuestras contribuciones es el introducir la semántica de expresiones de ruta **PE's** (en inglés " Path Expression ") en el lenguaje de consultas OQL. Estas expresiones de rutas son útiles esenciales para la navegación dentro de los objetos complejos que caracterizan la orientación a objetos y que constituyen realmente un fundamento para un lenguaje de consultas.

El concepto ó "approach" de expresión de ruta no es nuevo [Mylopou80], [Acc+97], [KKS92] y [Chr94].

5.8.1 Características

Las expresiones de ruta pueden contener variables que modifiquen las clases, los atributos, los métodos. Por consecuente se hace imposible efectuar una consulta sobre un dato sin tener un conocimiento completo del esquema.

Las expresiones de ruta pueden contener seleccionadores que toman un dato una porción del esquema. (por ejemplo juan.residencia.ciudad, juan juega el rol del *seleccionador*, mientras que residencia y ciudad son *expresiones de atributos*).

5.8.2 Semántica

La principal dificultad para implementar las consultas tiene que ver con la evaluación de las expresiones de ruta por ejemplo el tratamiento de las variables de atributos y de rutas. Un **PE's** define un ruta de un objeto a otro utilizando las relaciones de herencia. Proponemos un simple **PE's** orientado hacia la manipulación de documentos estructurados, inspirado en los trabajos de [Acc+97] y [Sen97].

Definición 1: Expresión de ruta

Una expresión de ruta nos permite describir un ruta de un objeto y acceder a uno de sus componentes. La definición de una expresión de ruta se hace utilizando anotaciones como son : " $\rightarrow$ " ó ".", indistintamente para navegar en el interior de los objetos o seguir relaciones. Una ruta es de la forma $\mathbf{c} \rightarrow \mathbf{a_1} \rightarrow \mathbf{a_2} \ldots \rightarrow \mathbf{a_n}$, donde $\mathbf{c} \in \mathbf{C}$ y $\mathbf{a_1}, \ldots \mathbf{a_n}$ ($\forall \mathbf{i}, \mathbf{a_i} \in \mathbf{A}$) son nombres de atributo. Se dice entonces que el ruta se define en **c** y lo podemos representar como: $\overset{c}{\rightarrow}$.

En general, una expresión de ruta es una secuencia que contiene:

a) los nombres de los atributos (por ejemplo: a) que pertenece a un n-uplet o a una unión de tipos.

b) índices de listas (por ejemplo [i])

c) referencias de los valores de objetos (por ejemplo →)

d) "traversées" de conjuntos (por ejemplo {v})

5.8.3 Dominios

nombres de atributos, **A** = { a_1, a_2, ...} ;

identificadores de objetos, **O**= {o_1, o_2, ...} ;

nombres de clases, **C** = {c_1,c_2 , ...} ;

nombres de métodos **M** = {m_1,m_2, ... } ;

K unión de los valores de los dominios atómicos : integer, string, boolean y float

Los conjuntos **A, O, M** y **K** constituyen alfabetos de símbolos.

5.8.4 Tipos y Clases

La estructura de un valor se describe por su tipp. Los tipos se definen usando constructores list, set y tuple, a partir de nombres de clase.

1. Los dominios atómicos : integer, string, boolean y float
2. Los nombres de clase en C son tipos
3. Si t est un type, {t} et {t} sont des types (liste et ensemble) ;
4. Si t_1, ... t_n sont des types et a_1, ... , a_n nombres distintos de atributos en **A**, [a_1 : t_1, ... , a_n : t_n] es un tipo (n-uplet)

Remarcamos que *T* es el conjunto de los tipos

Définition 2 : Classe

Definimos la clase como la asociación entre el nombre de la clase y un tipo.

Ejemplo (Tipos y Clase)

Consideremos la definición de las clases siguientes:

Un_Elemento : [dirección :string, tipo :string, ascendiente :Un_Elemento, descendientes Lista_Elemento, prev : Un_Elemento, next : Un_Elemento, contiene :multimedia]

Definición 3 : Función Tipo

La función tipo: $C \cup A \rightarrow T$

Permite identificar el tipo (c) $\in T$ de una clase o de un atributo.

Ejemplo (Función Tipo)

tipo(artículo) = [nombre : string, fuénte:boolean, nodo :list, pred :list, succ :list]

5.8.5 Valores y Objetos

Dado un conjunto o (de oids), sub-conjunto de O un valor o se define de la forma siguiente:

1. Cada elemento de O donde K es un valor
2. Si $v_1, \ldots, v_n$ son valores entonces $[a_1 : v_1, \ldots, a_n : v_n]$ es un valor de n-uplet
3. Si $v_1, \ldots, v_n$ son valores entonces $[v_1, \ldots, v_n]$ es un conjunto de valores finitos

Definición : Extensión de la función tipo

Extendemos la función tipo sobre el conjunto de todos los valores tipo: O $\rightarrow T$

$\forall$**v** $\in$ **O, tipo(v)** $\in$ **{boolean, string, integer}**

Ejemplo **(Atributo de tipo lista)**

Supongamos que tenemos una raíz de persistencia llamada LosDocs de tipo List y que que deseamos obtener el título del primer documento de la base, entonces la sintaxis de la consulta será la siguiente:

LosDocs[0].title

Ejemplo **(Expresión de camino)**

Para acceder a los elementos de la estructura lógica o a los atributos de un documento el usuario debe especificar ciertas expresiones de caminos para navegar a través de la estructura de los objetos o de los valores.

Proponemos la formulación de una consulta del tipo expresión de camino para navegar en la base accediendo a los valores de la estructura y del contenido de los documentos :

LosDocs $\rightarrow$ Un_Elemento $\rightarrow$ contenido es una expresión de caminp, aplicada a un objeto de la clase LosDocs permite acceder al valor de su atributo Un_Elemento y después al valordel atributo contenido. Setrata de un camino sobre LeDocs ($\overset{\text{LesDocs}}{\rightarrow}$).

El usuario puede formular una consulta utilizando el filtro select-from-where (SFW). El resultado de una consulta es un conjunto de objetos o de valores que satisfacen los predicados de selección. El resultado de una consulta es pressentado a través de métodos de presentación genérica disponibles en O_2 .

Proponemos formular las consultas utilizando el filtro SFW para navegar en la base accediendo a los valores de la estructura y del contenido de los documuentos.

Podríamos utilizar las sintaxis siguiente:

select [c.name | var | *
 from var in c.name
where var.condition]

Q1. Dar los documentos de tipo artículo que contengan palabra "matis"

select e->article
 from LosDocs e->Los_Elementos **e**
where e->contenido="matis"

Q2. Dar los títulos de una tesis en la base

select e.título
 from Los_Docs e
where e->type="tesis"

Las consultas propuéstas son formuladas con la ayuda de un filtro select-from-where. En la cláusula **from** el usuario declara una lista de etiquetas par especificar los caminos de acceso a los objetos de una base y eventualmente las variables implicadas en la evaluación de una consultas. La cláusula where se utiliza como de costumbre para expresar las condiciones sobre los objetos de la base.

5.9 Conclusión

Los lenguajes de consulta de sistemas de búsqueda de información son privilegiados en la expresion en lenguaje natural, ya que estos lenguajes tratan los sistemas de adminisración de base de datos, y particularmente los que manipulan los documentos estructurados, son declarativos y demandan el conocimiento de un usuario a

nivel experto. (al menos debe tener conocimiento del lenguaje y la estructura de los documentos). OQL nos ofrece una serie de operadores de comparación sobre los valores alfanumericos (=, < , >, !=, etc), textuales (like) y de conjuntos (in).

Las reflexiones actuales sobre la flexibilidad de los lenguajes de consultas de tipo declarativo, introducen variables sobre los atributos o bien la noción de ruta de acceso parcial en los sistemas orientados a objetos. Por otro lado, la integración de éstos lenguajes en módulos de interrogación mas naturales (por ejemplo, la búsqueda de texto completo y la búsqueda temática) muestra la necesidad de un acercamiento entre los lenjuages naturales de SBI.

Una mejor integración para el acceso a una base de datos que contenga estructuras de tipo B-árbol (B-tree en inglés), y de acceso específico a textos (archivo en modo inverso). Consideramos, sin embargo, la posibilidad de una falta de acoplamiento entre la base de datos y el mecanismo de indexación completa del texto que permite la recuperación de la ruta de la base de datos y la utilización de varios niveles de granularidad de indexación adaptados a la necesidad de la aplicación. El índice de todo el texto puede ser llamado directamente a través de un operador algebraico y utilizar dos tipos de estructura de acceso.

Capítulo 6

Presentación del Prototipo

6.1 Introducción

Los sistemas de administración de bases de datos orientadas a objetos proveen hoy en día una variedad de tipos de datos: atómicos, tipo texto, tipo multi-valores, etc., y constructores de tipo complejo (tipo lista), capaces de representar la estructura compleja de los documentos, así como de los potentes lenguajes de consulta basados en las nuevas propiedades del modelo orientado a objeto (identidad de objetos, composición, herencia, encapsulación, etc.),

6.1.1 Presentación del prototipo

El prototipo de sistema de búsqueda de informaciones sobre el cual trabajamos está basado en una base de datos constituida de cotidianos de información: *"my Personal Daily News"* myPDN [Fou98], que fue definida para representar y manipular documentos estructurados en el seno de un SABDOO - Sistema de Administración de Bases de Datos Orientadas a Objetos [$O_2$97]. Esta tecnología nos ofrece la facilidad de almacenar los datos estructurados y los datos multimedia. Veremos a continuación, de forma mas detallada las características de ésta tecnología. Disponemos además, de herramientas de comunicación suficientes como los servidores HTTP para consultar e

interrogar un cuerpo de documentos a partir de los clientes WWW. La idea de myPDN es poder consultar distintos periódicos haciendo búsquedas en base al contenido y estructura.

6.2 Características del sistema O_2.

En ésta sección presentaremos a groso modo las características principales del sistema O_2, que adoptamos para nuestro prototipo. Ahora repasaremos los conceptos básicos de un SBDOO - Sistema de Dase de Datos Orientado a Objetos: a) la noción de esquema, b) el concepto de clases, sub-clases y tipos de datos. En un segundo plano daremos el significado de objeto, valor, herencia, etc.

6.2.1 Esquemas, tipos y clases

Explicaremos de forma breve lo que es un conjunto de clases en el sistema O_2 y lo que éste representa. El esquema O_2 define el conjunto de las clases ligadas por la relación de herencia o de composición. Define así mismo los métodos y la encapsulación de los objetos.

En el sistema O_2, una clase no es un objeto, una clase agrupa los objetos que utilizan el mismo método, con el mismo tipo de datos y el mismo valor. Después de haber definido una clase, declaramos el tipo de la estructura de objetos a través de una aplicación recursiva de los operadores n-uplets, conjuntos y listas (tuple, set y list). Podemos incluir las firmas de los métodos aplicables a los objetos de la clase.

```
class  Un_Elemento
type tuple  (filename:string, content:multim, type:strin … )
method public  …
public init
   end;
```

En la definición de la clase `Un_Elemento` las palabras clave se encuentran escritas en negrilla, la encapsulación del objeto se realiza gracias al término **public**.

El sistema O_2 no permite acceder directamente al conjunto de objetos de una clase cualquiera (por ejemplo la clase `Un_Elemento`). Si queremos ver el contenido de una clase debemos codificarlo en el método.

6.2.2 Objetos, Valores y objetos nombrados

La información en O_2 es organizada en objetos y valores. Un objeto O_2 es una tupla (identificador, valor) donde el identificador del objeto es independiente del valor que contiene. Un objeto tiene tres características :

- Una identidad (oid)
- Un valor (es el valor del objeto)
- Un método (describe el comportamiento de datos y valores descritos por un conunto de programas, mudulos y métodos).

Un objeto se dice que encapsula los datos y su comportamiento. Para crear nuevos objetos utilizaremos la instrucción `new` de O_2. Si retomamos como nombre el ejemplo de la clase `Un_Elemento`, y `var` como el nombre de una variable obtendremos la asignación siguiente : var = new Un_Elemento;

La administración de la persistencia de datos se hace a través de raíces de persistencia de objetos o de valores persistentes. La administración de un conjunto de objetos persistentes de una misma clase es posible a través de una raíz o conjunto de persistencia que puede ser, por ejemplo un valor de tipo `set(Un_Elemento)`. Cada nuevo objeto de `Un_Elemento` será entonces insertado en éste conjunto a través de un método escrito por el programador, como el método `init`.

Las raíces de persistencia son objetos o valores nombrados, en nuestro ejemplo de la clase `Un_Elemento` definimos un objeto nombrado (`nombre_Elemento`) y un valor nombrado (`Los_elementos`). Podemos codificarlos así:

```
name  nombre_Elemento  : Un_Elemento;
name  Los_Elementos : set(Un_Elemento);
```

6.3 La arquitectura

En ésta sección efectuaremos un sobrevuelo de la arquitectura general del prototipo MyPDN [Fou98]. Este prototipo está basado en un SABDOO - Sistema Administrador de Base de Datos Orientado a Objetos (O_2) y se organiza alrededor de varias entidades de las cuales daremos las principales funcionalidades. En la figura 6.1, presentamos los componentes que describen la arquitectura general de la aplicación.

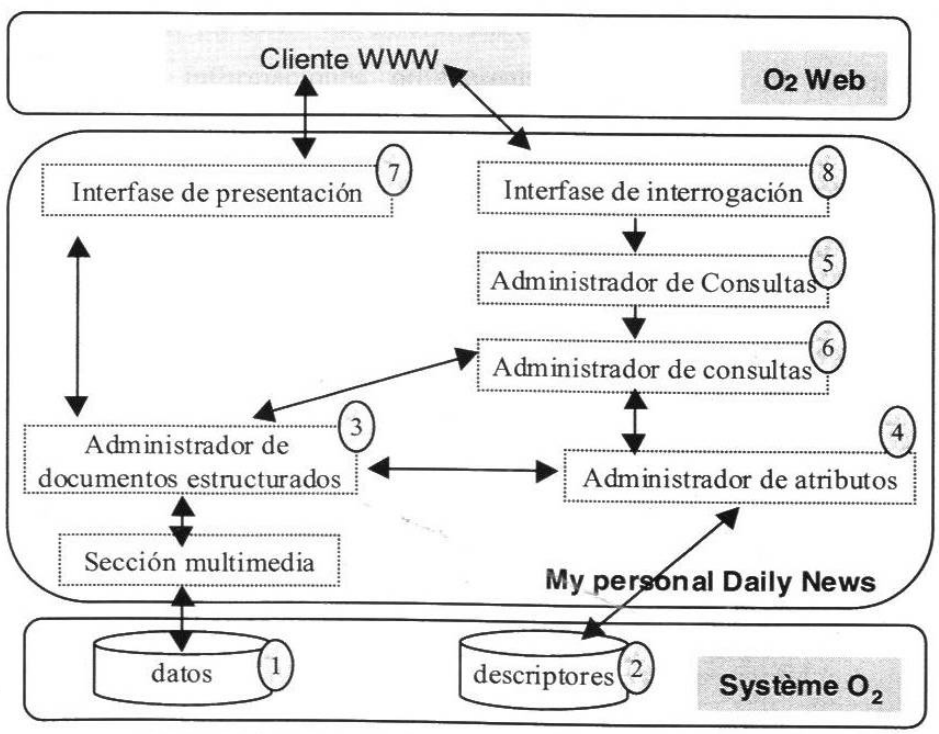


Figura 6.1 Arquitectura general de myPDN

- **Un núcleo de administración de documentos estructurados:** Ayuda a administrar las funcionalidades esenciales de manipulación de los documentos estructurados, respetando nuestro modelo de representación del documento estructurado. Cada una de las tres relaciones de estructura que presentamos para articular la organización de los elementos de la estructura se encuentra representada en ésta entidad, así como las funcionalidades de búsqueda de la arborescencia estructural.
- **Un núcleo de administración de los atributos dinámicos:** En myPDN, separamos explícitamente los documentos de sus atributos. Así, la administración de los atributos dinámicos es independiente de la administración de los documentos. A nivel aplicación establecemos una comunicación entre los atributos y los elementos de estructura que éstos representan, en lo que concierne al modelo, establecemos las relaciones $\Re_{ST}$ y $\Re_{SM}$ (capítulo 5), así como la función **value** que provee el valor de un atributo para un elemento estructural dado. Esta separación permite también conservar las informaciones provenientes del modelo de documentos extendido, y por consecuente de la indexación estructural.
- **Una interface de interrogación básica y una interface de presentación:** Utilizamos para ello el módulo de O_2, que permite interactuar entre una base de datos O_2 a un servidor Web (O_2Web). Presentamos en la figura 6.2, los principales elementos de la arquitectura de O2Web y el servidor de nuestra base.

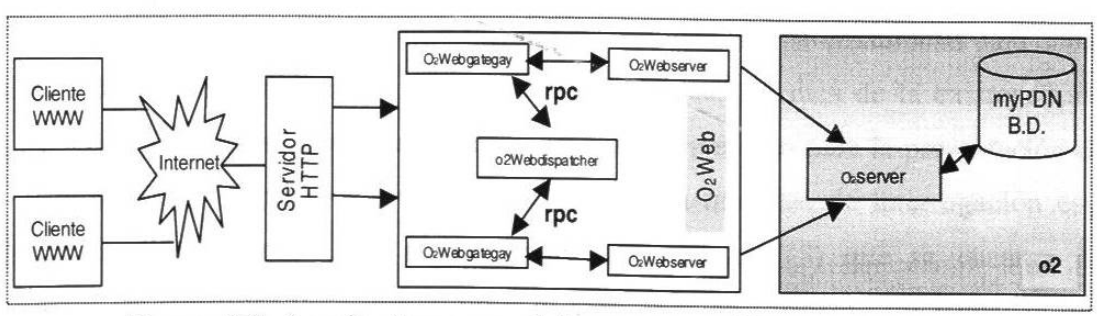


Figura 6.2 Arquitectura general d acceso a myPDN vía o2Web

- **O2Webgateway :** es la puerta de acceso a O_2Web. Cuando un cliente WWW emite una consulta, ésta es transmitida por el servidor HTTP al o2webgateway.
- **O2Webdispatcher** : es el elemento de control para distribuir las consultas de los o2webgateway.
- **O2Webserver** : los servidores O_2Web reciben las consultas de los o2Webgateway, y las ejecutan sobre la base especificada y las transforman en documentos HTML para ser restituidas al cliente del WWW que emitió la consulta. Este componente nos permite visualizar los datos de la base O_2, en éste caso, los documentos estructurados, de todos los clientes del WWW (Netscape Communicator, Microsoft Explorer, etc.). La interfase de presentación hará el llamado al administrador de documentos estructurados para recuperar los datos y transmitirlos al servidor o2Web que los restituirá al servidor HTTP.

Describiremos la arquitectura general que representamos en la figura 6.1 en la cual separamos los elementos de nuestra aplicación y los del sistema O_2 y de la herramienta O_2Web. Según ésta arquitectura general separamos claramente el almacenamiento de los descriptores de los elementos estructurales (2) del almacenamiento del contenido de los elementos estructurales (1). Por otro lado, introducimos dos administradores elementales: el primero, para la administración de los documentos estructurados (3) y el segundo, para la administración de los atributos de éstos documentos estructurados (4). Estos dos administradores se comunican para saber cuales atributos describen cuales elementos estructurales. Además de la existencia de dos módulos de interfase: uno para la interrogación (8) y el otro para la presentación de los documentos y su presentación hipertextual (7). La interfase de interrogación está relacionada directamente a un administrador de consultas (5) que se encarga de interpretar las consultas y de transmitirlas al repartidor de consultas (6) y realiza dos tareas: la repartición de las consultas estructurales y consultas sobre el contenido, así

como la fusión de los resultados otorgados por el administrador de documentos y de atributos. Este repartidor provee los resultados a la internase de presentación que los presenta al usuario y realiza posteriormente los accesos a los documentos para su consultación.

6.4 El administrador de documentos

- **Su presentación:** La representación interna de los documentos estructurados se articula alrededor de una clase única de objetos: la clase Un_Elemento (c.f. figura 6.3). Esta clase, mas allá de la consideración de los descriptores, posee las informaciones necesarias para la representación de las relaciones de estructura de los documentos. Consideramos así dos atributos para la relación de secuencia, ***prev*** y ***next*** y finalmente un atributo para la relación de referencia, ***ref***.

- **La composición:** el atributo ascendiente apunta en la lista los objetos correspondientes a los descendientes estructurales. Las funciones *Prevcomp* y *Nextcomp* proveen respectivamente el conjunto de ascendientes y de descendientes, se instancian bajo la forma de métodos de la clase Un_Elemento.

- **La secuencia:** el atributo ***prev*** apunta el objeto correspondiente al predecesor en la secuencia, el atributo ***next*** apunta el objeto correspondiente al sucesor en la secuencia.
- **La referencia:** el atributo ***ref*** apunta sobre el objeto destino de la relación de referencia.

6.5 La base de documentos

Actualmente, el prototipo myPDN administra 6 periódicos del cotidiano de informaciones *"Liberación"* disponible en Internet †. Hemos tomado y adaptado para nuestra experimentación el sitio del periódico *"Liberación"* en 6 fechas diferentes con el fin de constituir nuestra base de documentos. Los documentos recuperados se presentan bajo la forma de archivos HTML; una vez recuperados, son tratados con el fin de poder ser cargados en la base.

Este tratamiento, en una gran parte manual, consistió en identificar en los archivos HTML los elementos de estructura correspondientes a los tipos de estructura que proponemos, así como en la introducción señaladores para delimitar éstos elementos en los archivos.

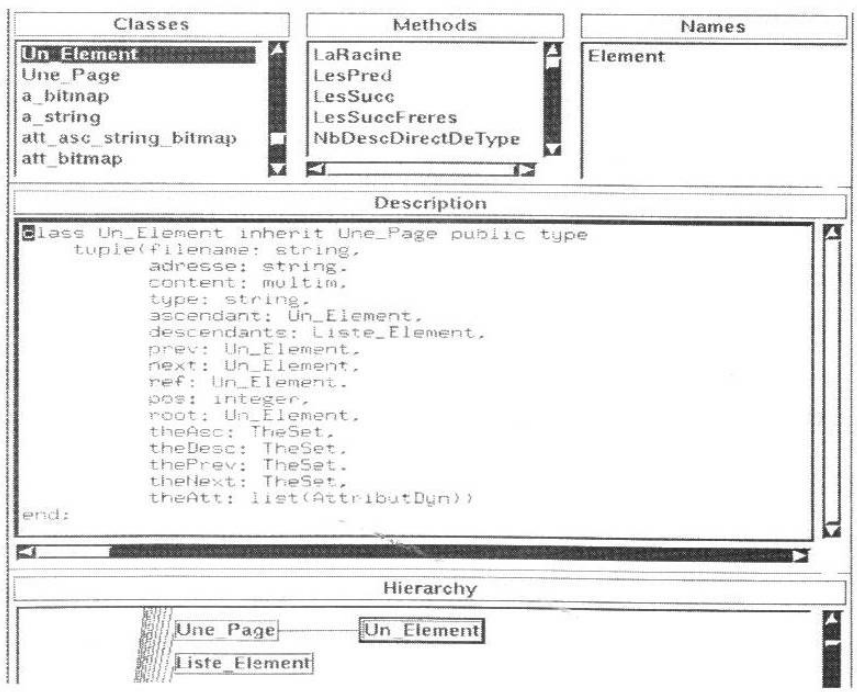


Figura 6.3 - Visualización de la clase UN_Elemento

† Dirección del sitio de Liberación: http://www.liberation.com

El protocolo para cargar la base, adaptado a los documentos que manipulamos, se establece de la manera siguiente: un objeto de la clase Un_Elemento (ver fig. 6.3) se crea para cada elemento de estructura reconocido en un archivo HTML, los atributos de éstos objetos que describen las relaciones de composición y de secuencia, que son instanciados posteriormente. Tuvimos que escoger un modo de almacenamiento de datos físicos de los documentos. Es evidente que la separación entre la estructura y los datos debe ser explícita en la base. El almacenamiento de los documentos se hace bajo la forma de una arborescencia de elementos estructurales. Conservamos así las diferentes partes del documento y representamos, gracias a la agregación o a las listas, las relaciones estructurales entre los elementos. Este enfoque necesita un módulo particular para la visualización del documento que permite reconstruirlo a partir de sus componentes.

Por otro lado, para representar la organización de los atributos que describimos en la sección 5.7.2 del capítulo 5, definimos las clases de objetos para tratar los atributos dinámicos y para representar toda la organización del documento (Btree mas dos niveles de Bitmaps). Estas clases que representan los atributos dinámicos heredan propiedades genéricas de dos clases principales: la clase a_bitmap para manipular los dos niveles de bitmaps y la clase att_domain para manipular los diferentes atributos del dominio tales como texto, fecha, etc. La principal diferencia entre éstas es su método, que se hace cargo de la propagación.

Para cargar los atributos del contenido semántico, hemos indexado los elementos hojas de la estructura de composición de los documentos bajo la forma de palabras clave que cargamos luego en un cuadro para representar el contenido semántico, posteriormente creamos los bitmaps del documento de referencia y los bitmaps para los elementos de la estructura. Después de ésta etapa, el proceso de las características de los atributos se hace para hacer llamado al método que concluirá el segundo nivel de los elementos de la estructura de bitmaps.

Conclusión General

I. Este trabajo es el fruto de múltiples reflexiones que dan seguimiento al conjunto de discusiones ligadas a la problemática y funcionalidades en la manipulación de los documentos: modelización, lenguajes de consultas, acceso concurrente, etc. Estas reflexiones concernientes sobre la buena integración de técnicas de los SBIs y los SMBDs son primordiales en el avenir y en la función de los sistemas de información sobre el marco de los SBI. Sin embargo, éste trabajo de tesis no aborda en forma exhaustiva todos los problemas relativos a búsqueda de información, dejando abierto éste campo de investigación.

II. En el primer capítulo hemos presentado de forma general la motivación de nuestra investigación, y nuestros objetivos de trabajo. En el capítulo segundo hemos realizado un sobrevuelo sobre los conceptos de base de los sistemas de búsqueda de información, de las técnicas de estructuración y de búsqueda de información. Además, hemos presentado de forma general los diferentes tipos de búsqueda y por añadidura, de manera indirecta, los tipos de estructuración que ellos realizan.

El capítulo 3 lo hemos consagrado a la presentación de dos normas: SGML y ODA, éstas presentan la ventaja de ser fácilmente comprensibles para describir los tipos de elementos estructurales y las informaciones descriptivas dadas en forma de atributos a nivel de tipos (ejemplo: tipo Libro, Periódico, Tesis, etc.). Somos partidarios de ésta ventaja durante la fase de indexación estructural utilizada en la definición de nuestro modelo. Enseguida, en el capítulo 4 hemos estudiado los prototipos de acuerdo a dos puntos de vista diferentes: Los Sistemas de Búsqueda de Información (SBI) y los Sistemas Manejadores de Base de Datos (SMBD).

Hemos abordado en el capítulo 5, la presentación del modelo propuesto por Fourel [Fou98], para la manipulación de documentos estructurados, así como la integración de técnicas de SMBD y de SBI en el modelo, ofreciendo facilidades de acceso para la interrogación. Además, hemos implementado la utilización de una tabla de "bits" de dos niveles tomando la forma de vectores de bits, para filtrar la información con el fin de reducir el espacio de búsqueda, y así poder aumentar considerablemente él desempeñó de búsqueda, es decir una recuperación más efectiva de los documentos estructurados.

No obstante, consideramos el uso de operadores de tipo expresiones de camino, en inglés "path expression". Finalmente en el capítulo 6 describimos el prototipo myPDN "my Personal Daily News", desarrollado para manipular documentos estructurados bajo una plataforma orientada a objetos O2.

Perspectivas

III. En el marco de la búsqueda de información, y basándonos en el modelo de [Fou98], el objetivo a seguir durante este trabajo de tesis ha sido el de proporcionar medios de acceso a los documentos estructurados, y describir el modelo permitiendo proporcionar facilidades de acceso a los documentos estructurados de forma simple y homogénea, es decir respecto a su contenido y estructura, y que sea flexible y transparente para el usuario.

Pensamos que un primer desarrollo de nuestra investigación seria el de integrar nuevas relaciones de estructura con el objetivo de poder tomar en cuenta nuevos tipos de medias, tales como datos espacio temporales, secuencias de vídeo, etc.

Consideramos de igual forma la oportunidad de integrar nuevas técnicas de optimización para resolver consultas sin necesidad de "cargar" toda el documento de la base. En fin, no podemos que desear la puesta en marcha de una reflexión profunda sobre la continuación de este trabajo de investigación ...

Bibliografía

[Acc+97] Abiteboul (Serge), Cluet (Sophie), Christophides (Vassilis), Milo (Tova), Moerkotte (Guido) et Simeon (Jerom).-Querying Documents in Object Databases, in Journal Digital Libraries, vol 1, n° 1, April 1997, pp 5-19.

[Ammar95] Ammar KHEIRBEK .- Modèle d'integration d'un Système de Recherche d'informations et d'un Système Hypermédia basé sur le formalisme des Graphes Conceptuels .-.Application au système RIME .- .-Thèse de Ph.D, Université Joseph Fourier - Grenoble I, 1995.

[Apple86] Apple, - Guide de l'utilisateur HyperTalk. - © Apple Computer Inc. 1988.

[Aqmw+97] Abiteboul (Serge), D. Quass, J. McHugh, J. Widom, J. L. Wiener, The Lorel Query Language for Semistructured Data, in Journal of Digital Libraries 1(1) 68-88,1997.

[Ass91] Association (Danish Standards). - SGML-ODA, Présentation des concepts et comparaison fonctionnelle.- afnor technique, 1991.

[Bal90] Balpe, J.P. Hyperdocuments, Hypertextes, Hypermedia .- Eyrolles, 1991.

[Beg88] Begeman, M.L. Conklin, J. The right tool for the job, Byte 1988, pp. 255-266.

[Ber88] Berrut (Catherine). - *Une méthode d'indexation fondée sur l'analyse sémantique de documents spécialisés.* Le prototype RIME et son application à un corpus médical. - Thèse de Ph.D, Université Joseph Fourier - Grenoble I, décembre 1988.

[Ble91] Bleber, M. Issues in Modeling a Dynamic Hypertext Interface for Non-Hypertext Systems. Hypertext'91, pp. 203-217.

[Bdhs96] P. Buneman, S. Davidson, G. Hillerbrand, D. Suciu, A Query Language and Optimisation Techniques for Unstructured Documents Data, in Procedings of ACM SIGMOD International Conference On Management of Data, Montreal, Canada, pp. 505-516. 1996.

[Bush45] Bush, V. - As WeMay Think, Atlantic Montly, Vol. 176, Juillet 1945, pp. 101-108. Reproduit dans [Greif88].

[Bmn94] K. Böhm, A. Mueller, et E. Neuhold. "Structured Document Handling.- a Case for Integrating Database and Information Retrieval". CIKM'94 pp. 26-33, 1994.

[Bur91] Burkowski (Forbes J.).- The use of retrieval filters to localize information in hierarchically tagged text-dominated database. In RIAO91.- Barcelonne, Espagne, April 1991.

[Bur92] Burkowski, F. J. An algebra for hierarchically organized text-dominates databases.Information Processing & Management. Pergamon Press, New York.- Vol. 28, No. 3, pp. 333-348. 1991.

[Bryan88] M. Bryan . An author's guide to the standard generalized markup language. Addison-Wesley, 1988.

[Bsr92] W.B. Croft, A. Smith, and H.R. Turtle. "A loosely-coupled integration of a text retrieval system and an object-oriented database system". 15th. Ann. International SIGIR Conference, Proceedings : 313-323"

[Bk94] K. Böhm, et K. Aberer. "Storing HyTime Documents in an Object Oriented Database". CIKM '94, pp. 26-33, 1994.

[BRG88] E. Bertino, F. Rabitti et S. Gibbs. Query Processing in a Multimedia Document System. ACM Transactions on Office Information Systems. - Vol. 6, No. 1, pp. 1-41. January 1988.

[Con87] J. Conklin. Hypertexte: An Introduction and Survey. Computer, pp. 17-41, September 1987.

[Chr96] Christophides (Vassilis).- Documents Structurés et Bases des Données Objets. - Thèse de PhD, Conservatoire National des Arts et Métiers, Octobre de 1996.

[Chr94] Christophides (Vassilis), Abiteboul (Serge), Cluet (Sophie) and M. Scholl, from Structured Documents to Novel Query Facilities, in Proc of ACM SIGMOD International Conference on Management of Data, pp. 313-324, May1994.

[Fou98] Fourel, Franck.- Modélisation, indexation et recherche de documents structurés. -Thèse de PhD, Université Joseph Fourier, février 1998.

[Frakes92] Frakes, William Baeza-Yates, editors.- Information Retrieval, Data Structures & Algorithms . Pretince-Hall, 1992.

[GS89] Goeorges Gardarin, Shim Yoon .- Modeling and Querying Structured Hypermedia Documents. December 1994.

[GPV89] Gyssens, M. J., Paredaens, J. 1 Van Guchet. - A grammar-based approach towards unifying hierarchical data models. Proc. ACM SIGMOD International Conference on Management of Data, Portland, Oregon, 1989, pp. 263-272.

[GT87] Gonnet, G. H. 1 Tompa, F.W. Mind your grammar: a new approach to modelling text. Proc. 13th International Conference on Very Large Data Bases (VLD87), Brighton, England,. September 1987, pp. 339-346.

[Güiti84] R. Güiting, R. Zicari, D. Choy, An algebra for structured office documents, in ACM TOIS 7(2), pp. 123-157, 1989.

[Halin89] G. Halin.- Apprentisage pour la recherche interactive et progressive d'images : processus EXPRIM et prototype RIVAGE. Thèse de l'Université Joseph de Nancy I, 9 octobre 1989.

[ISO86] Information Processing - Text and Office Systems - Standard Generalized Markup Language, ISO 8879, 1986.

[ISO88] Information Processing - Text and Office Systems - Office Document Architecture, ISO 8613, 1988.

[ISO89] ISO.- Information Technology-Database Language SQL. Rapport Technique, ISO/IEC, April 1989.

[KKS92] M. Kifer, W. Kim, and Y. Sagiv. Querying Object-Oriented Databases. In Proceedings of ACM SIGMOD Conference on Management of Data. pp. 393-402. 1992.

[Loe94] Loeffen (A.).- Text Databases: A survey of texts models and systems. ACM SIGMOD Records, vol. 23, 1994.

[Mac90] Macleod (I.A.).- Strucutred and Retrieval of Strucutred Documents. Information Processing & Management, Vol. 26 n° 2, 1990. pp. 197-208.

[Mac91] Macleod (I.A.).- A query language for retrieving information from hierarchical text structures. Computer Journal, vol. 34, n° 3, 1991, pp. 214-222.

[Mec97] Mechkour M., Mulhem P., Fourel F., Berrut C., PRIME-GC : A Medical Information Retrieval Prototype on the Web. In Seventh International Workshop on research Issues in Data Engineering (RIDE 97), Birmingham, U.K., avril 1997.

[Mylopou80] Mylopoulous J., Bernstein P.A., Wong H.K.T., A Language Facility for Designing Database-Intensive Applications, ACM TODS, 5:2, pp. 185-207.

[Mul93] Mulhem P, Bruandet M-F., A way to compare Objects. In Second Internbational Conference on Information and Knowledge Management (CIKM93), Arlington, USA, november 1993.

[Mul95] Mulhem P, Berrut C., The RIME prototype. In MIRO final workshop, Glasgow, UK. 18-20 septembre, 1995.

[Nav96] Navarro (Gonzalo).- A language for Queries on Strucfure and Contents of Textual Database.- Santiago, Chile, Thèse de PhD, University of Chile, April 1996.

[Nel67] Nelson T. Getting It out of our system . - in Information Retrieval : a critical review, G. Schechter ed. Thompson Books, Washington D.C., 1967.

[Nel80] Nelson, T.H. - Replacing the Printed Word : A complet Literary System, Information Processing 80, S.H. Lavington Edition, North Holland Publishing Company.

[Niel90] Nielsen J. Hypertexte and Hypermedia, ed. Academic Press, San Diego (USA), 1990.

[Nie90] Nie (Jianyun).- Un modèle logique général pour les Systèmes de Recherche d'informations. Application au prototype RIME .-Thèse de PhD, Université Joseph Fourier, 1990.

[ODMG93] ODMG (Objetc Data Management Group).- The Object Database Standard: Release 1.1.- Morgan Kaufmann Publishiers, San francisco California, 1983.

[Ouni96] I. Ounis.- Etude des modèles logiques pour la recherche d'information.- Rapport MRIM - SUR96-002, Octobre 1996.

[O_2 97] O_2 Technology.- The O_2 User's Manual, release 5.0, Novembre 1997.

[Rijs79] C.J. Van Rijsbergen. Information Retrieval.- Butterwork, second edition, 1979.

[Salton83] G. Salton. M.J. Mc.Gill. Introduction to Modern Information Retrieval.- McGraw Hill, New York, 1983.

[Sen97] Arijit Sengupta. DocBase .- A Database Environment for Structured Documents. PhD Thesis, Indiana University, December 1997.

[SGML86] G. Shafer, A Mathematical Theory of evidence. Princeton University Press, 1976.

[SGML90] SGML french version, Traitement de l'information-Systèmes bureautiques-Langage normalisé de balisage généralisé (SGML) ISO8879, Afnor 1990.

[Tha90] Thanos C. - *Multimedia Office Filling* : The MULTOS *Approche*.- North Holland 1990.

[TSM91] Tague, J., Salminen, A., & McClellen, C. Complete formal model for information retrieval systems. Proc. 14th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval, Chicago, October 1991, pp. 14-20.

[Vab96] Volz Marc, Aberer Karl et Böhm Klemeus. - An OODBMS-IRS coupling for structured documents. Bulleting of the Technical Commitee on Data Engineering, Vol. 19 n° 1 March 1996, pp. 34-42.

Lista de Cuadros

Lista de Figuras