IMPLEMENTACION DE UN INTERFASE EN LENGUAJE NATURAL PARA CONSULTAR BASES DE DATOS RELACIONALES

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ciencias y Humanidades

Departamento de Computación

IMPLEMENTACION DE UN INTERFASE EN LENGUAJE NATURAL PARA CONSULTAR BASES DE DATOS RELACIONALES

EDWIN ESTUARDO LOPEZ ZAMORA

Modelo de trabajo profesional presentado para
optar al grado académico de
Ingeniero en Ciencias de la Computación,
en el grado de Licenciado

Guatemala

1992

Vo. Bo.:

(f) _____
Ingeniera Berta Patricia Castillo Avila
Asesor

Tribunal:

(f) _____
Licenciada Marta Julia Fernández

(f) _____
Ingeniera Berta Patricia Castillo Avila

(f) _____
Licenciado David N. Alvarez Zeceña

Fecha de aprobación: 10 de septiembre de 1992.

A mi amada esposa, por ser la mayor fuente inagotable de inspiración, ánimo y comprensión necesarios para realizar este trabajo,

gracias Maggie

A mis padres, por brindarme con mucho esfuerzo la oportunidad de estudiar

A mis amigos y en especial a aquellos que colaboraron conmigo en una u otra forma

CONTENIDO

LISTA DE TABLAS Y GRAFICAS

RESUMEN

El uso de lenguaje natural para accesar bases de datos de información constituye una de las mayores áreas de aplicación de la tecnología que involucra el entendimiento del lenguaje natural por medio del computador. El presente trabajo establece el estudio de los procedimientos, así como el desarrollo de un interfase en lenguaje natural para consultar bases de datos relacionales, que utilicen **SQL** como lenguaje de procesamiento.

Para el desarrollo de la investigación se definió un marco teórico sobre el proceso de entendimiento del lenguaje natural, ya que poco se ha escrito en el medio sobre lo que constituye el procesamiento de lenguaje natural por medio del computador, y dicho proceso teórico sirve de base para el entendimiento de la tecnología que involucra el desarrollo del interfase.

Otra parte importante del trabajo lo constituye el diseño e implementación del interfase, en la cual se analizan los requerimientos y cómo se implementaron en el desarrollo del mismo.

Por último se muestran ejemplos obtenidos de la interfase desarrollada, que dan idea del grado de alcance del trabajo.

PREFACIO

Lenguaje Natural es el que los humanos aprenden del medio ambiente y usan para comunicarse con otras personas, así como para expresar emociones y conocimientos.

En los últimos años con el desarrollo de la *Inteligencia Artificial* (**AI**), se han obtenido resultados importantes en la automatización de procedimientos para comprender y resolver situaciones que caracterizan a los humanos.

Una de las investigaciones importantes de la inteligencia artificial lo constituye la comunicación entre seres humanos y computadores por medio del lenguaje natural. Por ello se han creado bases teóricas para el entendimiento y uso del lenguaje natural, con el objeto de definir formatos de comunicación más eficientes y naturales entre programas de aplicación y sus usuarios. Así se han definido interfases de lenguaje natural para procesar instrucciones en el computador, recuperar información de bases de datos y efectuar traducciones de un idioma a otro.

Uno de los mayores "cuellos de botella" en el diseño de programas para el entendimiento de lenguaje natural, lo constituye la adquisición de suficiente conocimiento acerca del dominio en el que se emplee. La tecnología actual está limitada a trabajar con dominios muy bien definidos. Un área

de aplicación que se apega a estos requerimientos es el desarrollo de interfases de lenguaje natural para accesar bases de datos.

A pesar que las bases de datos guardan una gran cantidad de información, ésta es bastante regular y bien definida. Estas características, junto con la utilidad de que una base de datos puede ser accesada por medio de lenguaje natural, hacen de estas interfases una importante aplicación de la tecnología que involucra el entendimiento del lenguaje natural.

El presente trabajo estudia los métodos y procedimientos que esta tecnología emplea para desarrollar dichas interfases, incluyendo el diseño e implementación de un interfase en lenguaje natural, para consultar bases de datos relacionales que utilicen **SQL** como lenguaje de procesamiento.

I. INTRODUCCION

En lenguaje natural, la adquisición de suficiente conocimiento acerca del dominio de trabajo, constituye un problema aún no resuelto hasta el momento. Como contraste, la tecnología actual está limitada a trabajar dominios de aplicación bien definidos, y un área de aplicación fiel a estos requerimientos lo constituye el uso de lenguaje natural para accesar bases de datos.

El uso de lenguaje natural para accesar bases de información constituye una de las mayores áreas de aplicación de la tecnología que involucra el entendimiento del lenguaje natural por medio del computador, y el presente trabajo involucra el estudio de los procedimientos, así como el desarrollo de un interfase en lenguaje natural para consultar bases de datos relacionales, que utilicen **SQL** como lenguaje de procesamiento.

Este proyecto intenta dar una idea del trabajo que constituye el desarrollo de interfases en lenguaje natural para consultar bases de datos; sin embargo no quiere decir que haya tomado en cuenta todos los avances y lo último en tecnología para el desarrollo del mismo.

Para el desarrollo del presente trabajo se definió un marco teórico sobre el proceso que constituye el entendimiento del lenguaje natural, que es vital para el desarrollo del

proyecto, ya que poco se ha escrito en el medio acerca de lo que constituye el procesamiento de lenguaje natural por medio del computador, y dicho proceso teórico sirve de base para el entendimiento de la tecnología que involucra el desarrollo del interfase.

En el capítulo II se estudia el problema que constituye el entendimiento del lenguaje y la distintas gramáticas que se han desarrollado para un mejor entendimiento del mismo.

En los capítulos III, IV y V, se analiza el procesamiento de lenguaje natural por medio del computador, y se define el marco teórico-tecnológico necesario para desarrollar el interfase.

Otra parte importante del trabajo la constituye el diseño e implementación del interfase, en la cual se analizan los requerimientos y cómo se implementaron en el desarrollo del mismo. En los capítulos VI y VII se describe dicho proceso.

El trabajo finaliza con las conclusiones derivadas del proyecto, las cuales se incluyen en el capítulo VIII.

II. LENGUAJE NATURAL

A. Lenguaje natural y su entendimiento

Uno de los objetivos importantes de la ciencia de la computación, y en particular de la inteligencia artificial, lo constituye el procesamiento de lenguaje natural: un proceso computarizado que sea capaz de entender y procesar requerimientos de un usuario cualquiera que emplee un idioma determinado como español, inglés, alemán, ruso u otro.

Según Harris (1985:3-5), siempre que el lenguaje sea hablado o escrito, cada mensaje tiene una estructura y elementos del mismo que se relacionan unos con otros en formas reconocibles. La comunicación verbal es caracterizada por sonidos capaces de ser reproducidos por casi cualquier humano.

El estudio de patrones de sonidos es llamado **fonología**. El estudio de la estructura de las unidades del lenguaje y sus relaciones es llamada **sintaxis**. Fonología y sintaxis son partes importantes de la **lingüística**, que es el estudio del lenguaje y en particular del lenguaje natural. La **semántica** es el estudio de las relaciones entre las estructuras lingüísticas usadas y el significado pretendido.

Los *lingüistas* son los encargados de estudiar el lenguaje. Los métodos usados por los lingüistas para describir similitudes entre los distintos lenguajes se denominan **gramáticas**. Estos métodos son diseñados para mostrar las

relaciones entre las estructuras que componen una oración.

La gramática consiste de elementos permitidos entre oraciones y las reglas para colocar estos elementos juntos. Por ejemplo, en español las oraciones son descritas como una parte sustantiva llamada **sujeto** seguida por una parte verbal llamada **predicado**. Obviamente una definición formal de estos elementos es requerida para describir sujetos, predicados y sus componentes, la cual es usada hasta que los elementos puedan ser definidos como simples palabras.

Las palabras en una gramática son llamadas **vocabulario**. La gramática es hecha de reglas y el vocabulario agrupa conocimientos asociados con las mismas.

En lenguaje natural, los elementos gramaticales pueden ser palabras y frases, y las reglas especifican como los elementos pueden ser combinados.

El estudio de la gramática ha probado ser una de las mejores herramientas para comprender la comunicación humana y ha generado bastantes terminologías, las cuales se han clasificado como **niveles de análisis del lenguaje,** los cuales serán explicados en su oportunidad.

B. Lingüística básica

La **oración** se define como el elemento básico del lenguaje. Nosotros sabemos que la oración es **"un conjunto de palabras que expresan un sentido completo"**. Diferentes tipos de ideas,

objetos y acciones requieren distintas representaciones. Estas representaciones son clasificadas como partes del lenguaje.

Para expresar una idea completa, una oración debe estar compuesta, como mínimo, de dos partes llamadas **sujeto** y **predicado.** Así por ejemplo en la oración:

El niño juega pelota

el niño es el sujeto y *juega pelota* es el predicado.

Un grupo de palabras juntas que contienen un sujeto y un predicado es llamado **cláusula.**

1. **Sujeto.** Es la persona, animal u objeto de quien se habla en la oración y no tiene un lugar fijo. Al sujeto lo conforman varios elementos:

a. Nombre o Sustantivo: es el núcleo o parte central del sujeto, el cual representa a una persona, animal u objeto. En la oración *el niño juega pelota*, *niño* es el núcleo del sujeto. Un sujeto puede tener más de un núcleo, en cuyo caso hay un núcleo principal y el resto tiene la función de modificarlo.

Los nombres pueden dividirse en varias subclases según su significado (semántica), origen y morfología. Según su significado pueden distinguir los *nombres comunes* de los *nombres propios*. El nombre común designa seres o cosas de una misma clase o género. El nombre propio sirve para

distinguir a un ser o una cosa de los demás de su especie.

Los nombres poseen género, los cuales son **femenino** y **masculino**, a los cuales se les agrega el **neutro**. Son masculinos los nombre de varones y animales machos. Son femeninos los nombres de mujeres y animales hembras.

Los nombres también poseen número, que es una modalidad universal en las lenguas por la cual el nombre significa un solo ser o más de uno. Cuando el nombre indica un solo ser está en **número singular**, y cuando indica varios, está en **plural**.

b. **Determinantes:** son elementos constituyentes del nombre y se anteponen al mismo para precisar: su categoría gramatical (**artículos**), su posición (**demostrativos**), su relación con el poseedor (**posesivos**), la cantidad (**numerales**), vagamente (**indefinidos**), una pregunta (**interrogativos**), un sentimiento (**exclamativos**), una relación con el dueño (**relativos**). Ejemplos de determinantes son:

El, unos	**artículos**
ese, esta	**demostrativos**
mi, tu, su	**posesivos**
qué, cual, cuya	**interrogativos, exclamativos y relativos**
dos, cien, mil	**numerales**
todos, varios	**indefinidos.**

c. **Adjetivos:** son elementos ocasionales del sujeto y califican al nombre indicando propiedad, cualidad, color, forma o manera de ser.

d. **Pronombres:** sirven para reemplazar al nombre. Por su forma y significado los pronombres pueden ser:

   i. **Personales:** *yo, mi, tú, te, vosotros, le, usted, les*, etc.

   ii. **Demostrativos:** *este, ese, aquellos, esta, esas, aquellas, eso*, etc.

   iii. **Posesivos:** *mío, tuya, suyos, nuestras, vuestros*, etc.

   iv. **Numerales:** *uno, tres, primero, quinto*, etc.

   v. **Relativos:** *que, cual, quien, cuyo, cuanto, cuales*, etc.

   vi. **Interrogativos:** *¿quién?, ¿qué?, ¿cuánto?, ¿cuál?, ¿cuáles?*, etc.

   vii. **Indefinidos:** *alguno, ninguno, todo, alguien, nadie, algo*, etc.

e. **Preposición:** establece la relación entre un sustantivo y alguna parte de la oración. Algunas preposiciones separables son:

a	*con*	*en*	*para*	*so*
ante	*contra*	*entre*	*por*	*sobre*
bajo	*de*	*hacia*	*según*	*tras*
cabe	*desde*	*hasta*	*sin*	

f. **Conjunciones:** unen palabras o grupos de palabras. Pueden

ser *coordinadas* y *subordinadas*. Conjunciones coordinadas conectan palabras de igual rango, mientras que las subordinadas de distinto. *Y, e* (adición), *o, ni* (disyunción), *pero, sino* (oposición), *ahora bien, porque, pues, puesto que* (causa), son conjunciones coordinadas. *Ante, a causa* o *hasta*, son conjunciones subordinadas.

g. **Adverbio:** es una palabra invariable que reemplaza a grupos preposicionales o conjunciones coordinadas.

2. **Predicado.** Es el modificador del sujeto, y a su núcleo se le denomina **verbo**, que es el que lleva la parte de ejecutar la acción de la oración. Así en la oración *el niño juega pelota*, el verbo es *juega*.

Los verbos se puede clasificar de acuerdo con la acción sobre el objeto. Por la capacidad de transmitir acción, con el fin de afectar a seres y objetos, los verbos pueden ser:

a. **Transitivos:** son aquellos con poder de transmitir la acción para afectar directa o indirectamente a un ser u objeto. Ejemplo:

Mi padre *lee* *el periódico*

sujeto **verbo** **objeto directo**

b. **Intransitivos:** son los que no admiten objeto directo. Ejemplo: *Luis nada en la piscina.*

c. **Reflexivos:** son transitivos y se conjugan con el enclítico, que sirve de objeto directo. Ejemplo: *Yo me*

baño.

d. **Copulativos:** son aquellos que unen al sujeto con un atributo (adjetivo o nombre). Ejemplo: *El es inteligente.*

e. **Auxiliares:** son indispensables para conjugar los demás verbos en sus tiempos compuestos. Por ejemplo: *he obtenido, hemos estado,* etc.

El tiempo indica el momento cuando se realiza la acción verbal. Los tiempos del verbo son:

a. **Presente:** es un tiempo absoluto, que expresa coincidencia entre la acción y el momento en que se habla; ejemplo: *En este momento leo la lección.*

momento **acción**

b. **Pretérito:** expresa una acción pasada; ejemplo: *Ayer escribí una carta.*

c. **Futuro:** es un tiempo absoluto que expresa una acción venidera. Por ejemplo: *Juan vendrá mañana a comer.*

d. **Copretérito:** es un tiempo relativo. Expresa un acontecimiento que sucedió al mismo tiempo que otro, por ejemplo: *mientras yo jugaba el dormía.*

C. Clasificación de oraciones

Las oraciones se pueden clasificar de acuerdo a su forma en **oraciones simples** y **oraciones compuestas.** Una oración simple consiste de un sujeto y un predicado; una oración

compuesta combina dos o más cláusulas con conjunciones coordinadas o con comas.

Además podemos clasificar a la oraciones de acuerdo a su uso como: **declarativas, interrogativas, imperativas y exclamativas.** Una oración declarativa hace una afirmación, una interrogativa una pregunta, una imperativa expresa órdenes y una exclamativa expresa asombro.

Los siguientes ejemplos dan una idea clara de las distintas clasificaciones que poseen las oraciones:

Oración simple	*Todos los animales están en el zoológico.*
Oración compuesta	*La muchacha tiene licencia, sin embargo no sabe manejar.*
Oración declarativa	*El auto es japonés.*
Oración interrogativa	*¿Quién es ud.?*
Oración imperativa	*Quítese el sombrero.*
Oración exclamativa	*¡El avión se estrelló!*

D. Niveles de análisis

Como lo indica Luger et al (1989:379):

> "El lenguaje es un fenómeno complicado que involucra procesos variados, como el reconocimiento de sonidos o cartas escritas, análisis sintáctico, inferencias semánticas de alto nivel, y siempre la comunicación de contenido emocional a través de ritmo e inflexión".

Para manejar esta complejidad, los lingüistas han definido

distintos niveles de análisis para el lenguaje natural:

1. **Prosodia**. Se relaciona con el ritmo y entonación del lenguaje. Este nivel es dificultoso de formalizar y a menudo no es tomado en cuenta, sin embargo es importante en el efecto de la poesía y los cantos religiosos.

2. **Fonología**. Examina los sonidos que son combinados para formar el lenguaje. Este campo de la lingüística es importante para trabajar en reconocimiento y generación de lenguaje hablado por medio del computador.

3. **Morfología**. Involucra el significado de los componentes (morfemas) que fabrican palabras. Esto incluye las reglas que gobiernan la formación de palabras. El análisis morfológico es importante en la determinación del juego de palabras en una oración.

4. **Sintaxis**. Estudia las reglas para combinar palabras en frases y oraciones, y el uso de estas reglas para analizar y generar oraciones. Este ha sido el nivel automatizado más exitoso de análisis lingüístico.

5. **Semántica**. Estudia el significado de las palabras, frases y oraciones y la forma en que son combinados en

el lenguaje natural.

6. **Pragmática.** Es el estudio de la forma en que el lenguaje es usado y los efectos que produce en el que lo escucha. Por ejemplo pragmática indica las razones por las cuales *si* es una respuesta inapropiada para preguntas como *¿qué hora es?*

7. **Conocimiento del mundo.** Incluye el conocimiento del mundo físico y real. Este conocimiento es esencial para entender el significado completo de un texto o una conversación.

E. Gramáticas existentes

A continuación se presenta una sinopsis de algunas de las gramáticas propuestas para la interpretación y extracción del significado del lenguaje natural. Se principia con gramáticas definidas sintácticamente, hasta llegar a las gramáticas definidas con base en conocimiento. Dichas definiciones se basan en las investigaciones de Harris (1985:21-48).

Como se dijo con anterioridad, las gramáticas fueron desarrolladas por los lingüistas como un conjunto de reglas para mostrar las distintas estructuras que componen la oración, un proceso muy importante que es útil para entender cómo son generadas las oraciones.

1. Estructuras sintácticas de Noam Chomsky. Noam Chomsky publicó, en 1957, un libro llamado *Syntactic Structures*, que fijó durante algún tiempo la forma de pensar y trabajar de los lingüistas.

Chomsky dice que se debe considerar todo el lenguaje que una persona es capaz de producir, por lo que se concluye que el lenguaje es infinito.

También define al lenguaje como un conjunto de elementos de un vocabulario específico de acuerdo a reglas de gramática bien definidas. Las reglas son las siguientes:

a. Definir categorías sintácticas, que son estructuras que integran la oración, y que actúan como símbolos no terminales. Sustantivo, verbo, adjetivo y conjunciones son ejemplos de ello.

b. El vocabulario constituirá las palabras del lenguaje, que son consideradas como símbolos terminales que se unen para generar oraciones.

c. Las relaciones entre las estructuras que integran la oración y el vocabulario, son especificadas por reglas o producciones.

d. Las producciones generan un conjunto de oraciones que constituyen el lenguaje a producir por la gramática. Las producciones son iniciadas por un símbolo no terminal, el cual causa que todas las producciones subsiguientes generen todas las posibles oraciones que éstas

especifiquen. Por ejemplo, las reglas o producciones para la oración *El niño juega pelota* son:

<Oración> = **<Sujeto> <Predicado>**

<Sujeto> = **<Determinante> <Sustantivo 1>**

<Predicado> = **<Verbo> <Sustantivo 2>**

<Determinante> = *el*

<Sustantivo 1> = *niño*

<Sustantivo 2> = *pelota*

<Verbo> = *juega*

Este simple tipo de gramática generativa es llamada *una gramática de estado finito*. Sin embargo, los lenguajes naturales no son finitos y no pueden ser definidos adecuadamente por gramáticas de estado finito.

Pese a que las teorías fueron aceptadas por una gran cantidad de investigadores, se encontraron algunos problemas con ellas, como los siguientes:

i. El nivel sintáctico es completamente separado del nivel semántico.

ii. Oraciones con igual estructura, pero diferente significado, no son manejadas por estas teorías.

iii. Una oración simple puede tener más de un significado, lo cual no puede ser detectado.

La importancia de que la sintaxis por si sola no es suficiente debe ser enfatizada, ya que aunque es una condición necesaria, no provee suficiente información para entender el

lenguaje.

Estas gramáticas son útiles para definir tanto lenguajes de programación, como segmentos limitados de lenguajes naturales.

2. **Teoría semántica de Katz y Fodor.** Jerrold J.Katz y Jerry A. Fodor desarrollaron algunas técnicas para tratar con los problemas semánticos derivados de las teorías de Chomsky. Dicho trabajo se encuentra publicado en un artículo de 1964, el cual se titula *The Structure of a Semantic Theory*.

El concepto fue continuar con la separación entre la sintaxis y la semántica. Katz y Fodor propusieron dos componentes para una teoría semántica: **un diccionario y un grupo de reglas de producción**. El diccionario proveería para cada palabra en el lenguaje una descripción fonológica, información sintáctica e información semántica. Las reglas de producción estaban diseñadas para producir todas las interpretaciones válidas de una oración.

3. **Teoría estándar de Chomsky.** Noam Chomsky decidió incorpar algunos resultados de investigadores como Katz y Fodor, por lo que desarrolló un nuevo trabajo conocido como la teoría estándar, el cual fue publicado con el título de *Aspects of the Theory of Syntax* en 1965. Según Harris

(1985:31), la gramática consistía de:

a. Un componente sintáctico, dividido en:

1. La base

Reglas de estructura de frases

Lexicón

2. El componente transformacional

b. Componente semántico

c. Componente fonológico.

El componente sintáctico fue conocido como *generativo*, ya que dio origen a un análisis más profundo con base en las reglas y el lexicón. Los componentes semántico y fonológico fueron llamados *interpretativos*, debido a que la combinación del componente sintáctico con el semántico, provee un análisis semántico completo, mientras que con el componente fonológico, se obtiene la secuencia de sonidos.

4. **Gramática de casos.** En la conferencia *Universals in Linguistic Theory*, Charles J. Fillmore presentó un artículo titulado *A Case for Case*. Fillmore trata de dar solución a algunos de los problemas encontrados en las teorías de Chomsky utilizando la forma tradicional de casos, pero en forma mucho más moderna. Como lo indica Harris (1985:36), los casos hacen énfasis en las terminaciones de los nombres; estas terminaciones identifican al nombre e incluyen nominativos, genitivos, acusativos, dativos y ablativos. Estas

terminaciones ayudan a identificar la función que el nombre juega dentro de la oración, con lo que el orden de las palabras no es tan importante.

Fillmore propuso que los sujetos están siempre ligados con los verbos, en una llamada *relación de caso*. Las nociones de caso abarcan un conjunto de conceptos universales, que identifican a ciertos tipos de actitudes de los humanos, debido a eventos que les ocurren. Estos casos incluyen:

Agentivo o agente: instigador de la acción del verbo.

Instrumental: el objeto físico involucrado en el evento.

Dativo: el afectado de la acción del verbo.

Resultado: la entidad resultante de la acción.

Locativo: la identificación u orientación de la acción.

Objetivo: es lo que representa a un sustantivo cuya acción ha sido identificada por el verbo.

Las gramáticas de caso pudieron resolver ciertas ambigüedades de las gramáticas previas:

a. Oraciones idénticas pero de distinto significado pueden ser analizadas, debido a que cada oración pertenece a un caso distinto y no hay unión para casos distintos.

b. A oraciones derivadas de una oración simple, se les puede dar el mismo significado.

III. PROCESAMIENTO DE LENGUAJE NATURAL

Según Obermeier (1987:225), procesamiento de Lenguaje Natural (**NLP**, **N**atural **L**anguage **P**rocessing) es un método por el cual un computador es capaz de entender y procesar requerimientos de un usuario cualquiera que emplee un idioma determinado.

NLP generalmente es dividido en seis áreas:

1. Interfases de lenguaje natural para accesar bases de datos.
2. Traducciones de un idioma a otro por medio de computador.
3. Programas inteligentes de búsqueda e indexación, para operar grandes cantidades de texto.
4. Generación de texto para la producción automatizada de documentación.
5. Sistemas de habla que permiten la interacción con los computadores por medio de la voz.
6. Herramientas para desarrollar sistemas de NLP para distintas aplicaciones.

Estas áreas se pueden dividir en tres grupos básicos:

1. Análisis textual y reconocimiento de voz.
2. Desarrollo de interfases para recuperación de información.
3. Sistemas para efectuar traducciones automatizadas y desarrollo de herramientas para trabajar con lenguaje natural.

Como podemos observar en la fig. 3.1, las estadísticas indican que cerca del 40 por ciento de la actividad de investigación en NLP, está dedicada a desarrollar interfases de lenguaje natural para accesar bases de datos, otro 40 por ciento para traducción automatizada y herramientas para NLP, y el porcentaje restante, que tiene una actividad menos significativa, lo ocupan las demás áreas de investigación.

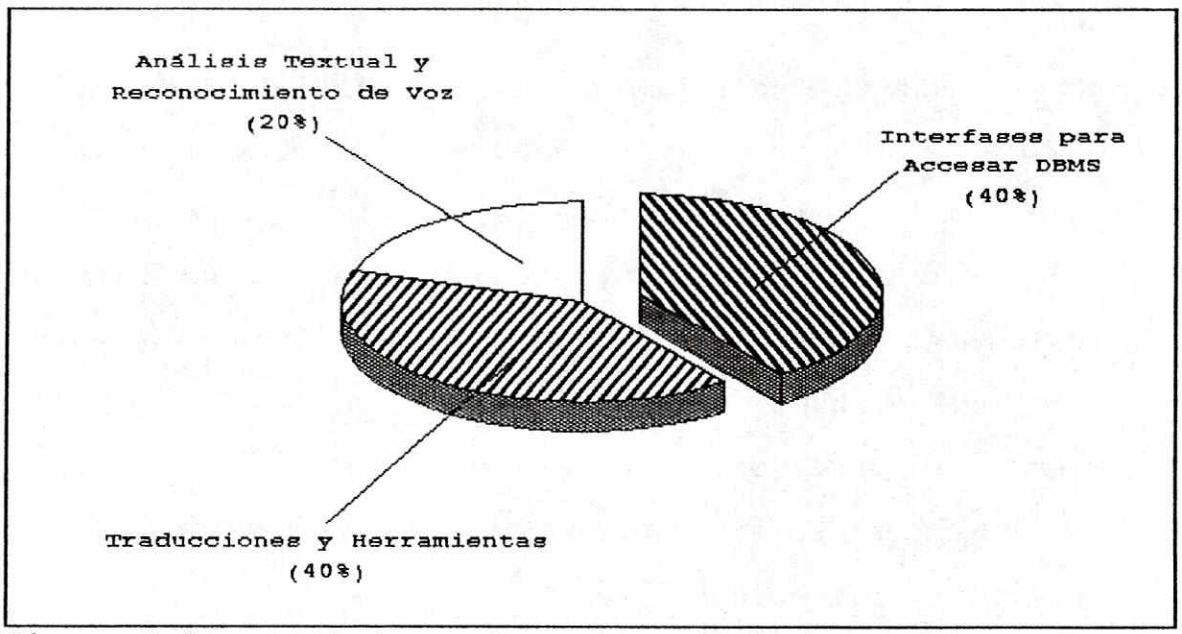


Figura 3.1 Estadísticas sobre el desarrollo de aplicaciones con lenguaje natural. Fuente Obermeier (1987:225).

A. Técnicas para el procesamiento de lenguaje natural

Según Obermeier (1990:217), el problema central al que se enfrentan los sistemas para procesamiento de lenguaje natural, lo constituye el hecho de que transforman ingresos, a veces ambiguos, en una representación (la cual varía de aplicación a aplicación) que es utilizada por un computador.

La transformación de ingresos ambiguos a esta representación es conocida como *parseo*. La palabra *parse* se deriva del Latín *pars orationis (parte del habla)*. Según Obermeier (1987:225-228), cinco diferentes tipos de parsers están involucrados en NLP:

1. **Parser tipo Pattern-Matching.** Los primeros programas para procesamiento de lenguaje natural trabajaban con patrones lingüísticos de la oración, sin recurrir a formalismos de alguna gramática específica. Al efectuarse el análisis, el sistema revisa una posible similitud con un número fijo de patrones; si alguna similitud es encontrada el sistema realiza una determinada acción.

ELIZA, un programa escrito en 1966 por J. Weizenbaum, es un ejemplo clásico de la técnica de pattern-matching. ELIZA consiste de un conjunto de patrones, y cada patrón tiene un número de copias asociadas con él. Cuando un patrón particular es identificado, el programa selecciona un conjunto de réplicas y hace las sustituciones necesarias en la copia.

Este proceso es similar a la técnica empleada por la inteligencia artificial para simular visión humana.

Este tipo de programas sin una gramática básica tiene uso limitado y son usados únicamente para análisis parcial.

2. **Parser basado en gramáticas.** Este tipo de parser es basado en los trabajos preliminares de N. Chomsky y consiste en construir una estructura de árbol a partir de un conjunto de reglas o producciones (ver fig. 3.2). Las terminaciones de las ramas son llamadas *terminales* y los otros símbolos restantes son llamados *no terminales*.

Este tipo de parser es utilizado en forma amplia y con éxito para traducir instrucciones hechas en algún lenguaje de programación, a operaciones ejecutables en el computador.

3. **Parser basado en semántica.** El significado de las oraciones ha hecho que los investigadores se inclinen por efectuar análisis semántico, en vez de sintáctico. Dos técnicas surgen de este trabajo: *gramáticas de casos* y *gramáticas semánticas*.

Gramáticas de casos han sido usadas desde 1968 y la técnica se refiere a que cada oración tiene asociada una representación de su significado. Esta técnica se basa en los trabajos de Fillmore y consiste en trabajar con los casos (agentes, objetos e instrumentos), que indican la relación de

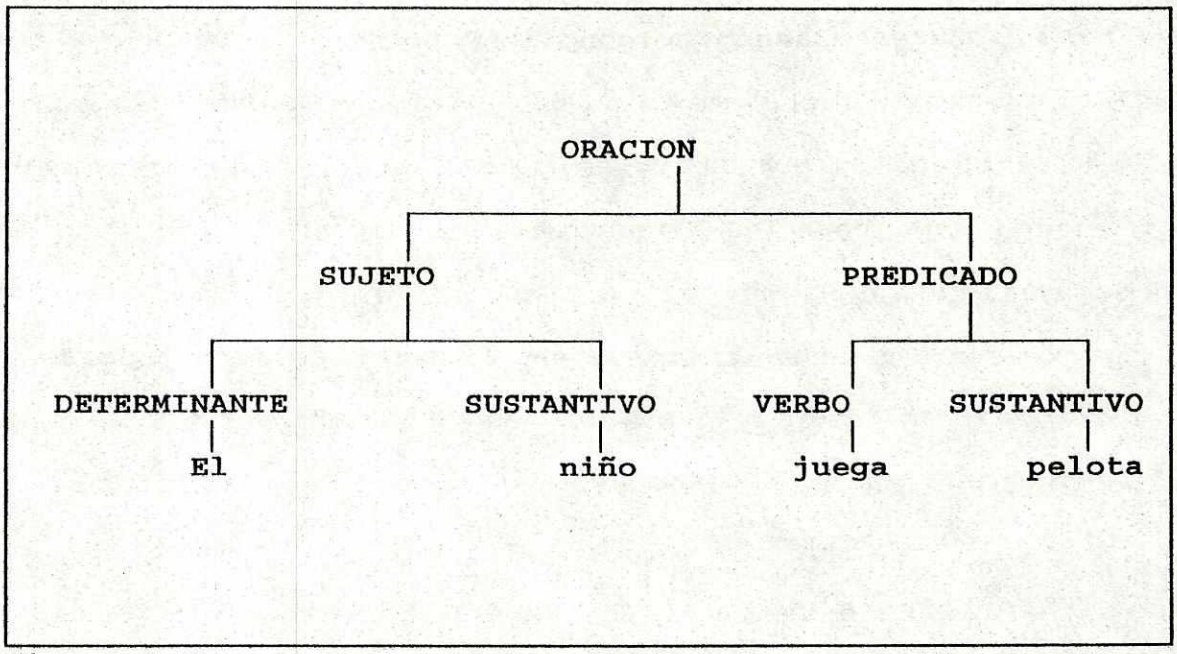


Figura 3.2 **Esquema tipo árbol para ejemplo de una oración generada por reglas o producciones. Fuente Obermeier (1987:226).**

los sustantivos con el verbo y el papel que éste desempeña en la oración (ver capítulo II).

Las gramáticas semánticas consisten de un diccionario denominado *lexicón* y una serie de reglas gramaticales. Esto es similar al análisis sintáctico, con la excepción que los sustantivos y verbos son especificados por su significado.

La ventaja que tiene este tipo de gramática es que es compacta, debido a que se analiza un pequeño número de posibilidades, pero posee como desventaja la dificultad que se presenta al cambiar las reglas gramaticales, ya que éstas determinan la semántica de los elementos del diccionario.

4. **Parser basado en conocimiento.** Algunos sistemas para procesamiento de lenguaje natural no sólo efectúan análisis sintáctico y semántico, sino que también accesan una base de datos de conocimiento, que contiene especificaciones de un dominio sobre el que se está trabajando. Esta técnica es llamada *parseo experto en palabras*, en la cual la palabra es considerada como la unidad básica de la lingüística.

5. **Parsers que utilizan redes neurales.** Una técnica reciente para el procesamiento de lenguaje natural, lo constituye el uso de unidades de procesamiento que actúan en forma similar a las neuronas. Cada unidad tiene un número de entradas, un conjunto de posibles estados y una salida que es producto de las entradas. Este tipo de sistema es llamado usualmente *red neural*.

Obermeier (1987:227-228) indica que el modelo léxico del parser de red neural contiene tres niveles de neuronas. El primer nivel es el nivel léxico, que es el de ingreso de la red. En el segundo nivel, las entradas del nivel léxico son combinadas para activar neuronas que representan el significado de las palabras. En el tercer nivel, que es el nivel lógico, los significados son combinados para formar predicados y objetos.

Aunque el modelo de red neural para NLP tiene como

objetivo modelar el procesamiento lingüístico humano, es un área nueva de investigación que promete mucho, pero también posee dificultades aún no resueltas. Luger et al (1989:585) enumera algunos beneficios y desventajas; entre sus beneficios tenemos los siguientes:

a. Las redes neurales manejan ambigüedades muy bien. Una vez ajustadas, las redes neurales demuestran habilidad para reconocer patrones, aunque los ingresos sean ambiguos o no se tenga suficiente información.

b. Son robustas. A causa de que la información es distribuida, pueden seguir procesando aunque algunos de sus nodos fallen.

c. Las redes neurales implementan paralelismo.

d. Son capaces de aprender nuevos conceptos.

e. Por ser en si mismas una arquitectura del cerebro, modelan el mecanismo de la inteligencia.

f. Tienen éxito en áreas como modelo de la visión humana.

Sin embargo, un número de dificultades deben ser resueltas:

a. Las redes neurales no son cerebros. Los cerebros no sólo aprenden ajustando conexiones, sino también las crean.

b. Las redes neurales no pueden modelar mecanismos cognoscitivos de alto nivel como atención, simbolismo y referencia.

c. Muchos de los procesos del intelecto humano no son en

paralelo, sino en serie. En este caso un sistema experto o de producción puede ser mejor para la solución de problemas.

d. El cerebro se compone de trillones de neuronas, por lo que aunque la cantidad de neuronas implementadas en computadores pueden realizar ciertos trabajos, no es suficiente para crear programas más inteligentes.

B. Técnicas de Parseo

Según Allen (1988:40-65), el procesamiento del lenguaje natural no involucra únicamente la elección de un determinado tipo de parser, también se debe emplear una técnica de parseo apropiada. Las técnicas de parseo se agrupan en: *determinísticas* y *no determinísticas*.

Los parsers no determinísticos se dividen en parsers *top down* y parsers *bottom up*. Los parsers top down inician su trabajo a partir de las reglas que se identifican con un ingreso determinado y recursivamente van hacia las reglas más internas. Los parsers bottom up trabajan al revés, pero tienen la dificultad de generar demasiadas alternativas para un ingreso determinado.

Los parsers determinísticos se diferencian de los no determinísticos, en que cuando un análisis falla, éstos no continúan analizando las posibilidades restantes. Sin embargo, estos parsers tienen la dificultad de que no pueden

usar información sintáctica para analizar los ingresos.

Como se ve, éstas estrategias de parseo son importantes para el procesamiento, tanto de lenguajes de programación, como para el lenguaje natural. A continuación, consideraremos los parsers que usan *redes de transición* que, aunque no son suficientemente poderosos, proveen la base para las *redes de transición aumentadas*, que constituyen una herramienta poderosa para el procesamiento del lenguaje natural.

1. **Redes de transición**. Las redes de transición están basadas en la teoría de grafos y máquinas de estado finito. Una máquina de estado finito es, en teoría, un dispositivo que trabaja con un estado inicial particular y luego cambia de estado cuando una condición específica ocurre. Un grafo es un conjunto de nodos sin dirección unidos por medio de rutas denominadas arcos.

Las redes de transición tienen las siguientes reglas:

a. Todos los símbolos terminales o palabras de la oración están representados por arcos.

b. Un estado o nodo del grafo es el estado inicial y un subconjunto de estados, constituyen el estado final.

Las dos principales redes de transición que se van a analizar son las *redes de transición recursivas* y las *redes de transición aumentadas*.

2. **Redes de transición recursivas.** Una red de transición recursiva (**RTN**) tiene un punto inicial y uno o más estados finales. De esta forma, un arco del grafo puede contener el nombre de otra subred a la cual se le puede transferir el control, para analizar un segmento particular de una oración. Cuando esta subred es terminada de recorrer, el control es regresado al nodo que la llamó y se continúa con el siguiente estado de la red. La recursión se da cuando una subred se llama a sí misma. Ya que varias subestructuras pueden ser analizadas simultáneamente, las RTNs tienen la capacidad de procesamiento en paralelo, aunque frecuentemente, muchas de las subredes llamadas fallan en el análisis de los segmentos correspondientes de la oración.

3. **Redes de transición aumentadas.** Las redes de transición aumentadas (**ATN**) son similares a las RTN, pero a éstas se le agregaron las siguientes características:

a. Almacenamiento en registros de los resultados parciales de las partes analizadas de la oración.

b. Un arco puede ser validado antes de ser procesado, ya que se pueden especificar condiciones asociadas a cada arco.

c. Se definen acciones asociadas con cada arco. Cada vez que un arco es tomado en cuenta, las acciones asociadas con él son ejecutadas.

Las ATN, en vez de aceptar o rechazar cada palabra o frase

que es encontrada, construyen la estructura de la oración conforme las palabras ingresadas encajen con los elementos de la red. Las distintas partes de la oración son retenidas en registros hasta que la estructura entera de la oración pueda ser determinada. Por ejemplo, según Harris (1985:160), cuando un verbo es encontrado, puede ser almacenado en un registro llamado **V**; similarmente todas las palabras de un sustantivo pueden ser guardadas en un registro **NS**.

Las ATN constituyen la técnica de más amplio uso para el procesamiento del lenguaje natural, debido a que al utilizar la habilidad de efectuar validaciones y especificar acciones, se provee un análisis semántico más poderoso. Sin embargo, una desventaja importante lo constituye el hecho de que las gramáticas y los programas que las interpretan pueden llegar a ser bastante complejos.

C. Representación de conocimiento

Cualquier sistema de procesamiento de lenguaje natural que no efectúe análisis estrictamente sintáctico, posee un mecanismo para representar conocimiento. El conocimiento no es simplemente pura información, sino que es un método por medio del cual, con información útil, se puede llegar a la solución de problemas, que a su vez pueden generar nuevo conocimiento.

1. **Almacenamiento de conocimiento.** Un sistema de NLP almacena generalmente su conocimiento en bases de datos, las cuales pueden ser *estáticas* y *dinámicas*.

Según lo indica Harris (1985:286), los primeros sistemas que almacenaban conocimientos usaban bases de datos estáticas, en las cuales la información contenida en ellas no admitía cambio alguno. **LUNAR**, un sistema desarrollado por William Woods, respondía preguntas acerca de las rocas lunares traídas a la tierra por las misiones Apollo. Ya que no se han traído más rocas lunares, no hay razón para cambiar la base de datos. Este es un ejemplo clásico de sistemas que almacenan conocimiento estático.

Una base de conocimiento dinámica, por otra parte, posee un conocimiento inicial almacenado, el cual puede ser modificado y a su vez tiene como característica la de permitir la adquisición de más conocimiento. Este proceso es llamado *aprendizaje*.

Cuando el conocimiento es almacenado como datos puros, se le denomina *declarativo* y cuando el conocimiento es usado para ejecutar código es llamado *procedural*. El conocimiento declarativo es eficiente y fácil de mantener y accesar, mientras que el procedural permite más flexibilidad.

2. **Esquemas de representación de conocimiento.** Durante 25 años, según Luger et al (1989:334), numerosos

esquemas de representación de conocimiento han sido implementados, cada uno con puntos fuertes, así como con debilidades. John Mylopoulos y Héctor Levesque clasificaron estos esquemas en cuatro categorías:

a. **Esquemas de representación lógica:** estos esquemas de representación usan expresiones en lógica formal para representar conocimiento. Ejemplo: la estructura de reglas de **PROLOG**.

b. **Esquemas de representación procedural:** en estos esquemas el conocimiento es representado como un grupo de instrucciones, que se usan para resolver algún tipo de problema. Ejemplo: conocimiento almacenado en sistemas de producción.

c. **Esquemas de representación por medio de redes:** estos esquemas capturan conocimiento por medio de grafos, en que los nodos representan objetos y conceptos de un dominio específico, y los arcos representan las relaciones asociadas entre ellos. Ejemplo: redes semánticas y dependencia conceptual.

d. **Esquemas de representación estructurados:** estas representaciones extienden las redes ya que permiten que cada nodo conste de atributos y valores, los cuales pueden ser numéricos o datos simbólicos, punteros a otros nodos, o procesos que realizan una tarea específica. Ejemplo: frames y scripts.

De estos esquemas, los últimos dos se consideran como los más importantes, por lo que a continuación procederemos a explicar más detalladamente cada uno de ellos.

3. **Representación de conocimiento por redes.** Los grafos son un método útil para representar conocimiento, ya que por medio de arcos y nodos, se representan las distintas relaciones que se derivan del dominio específico de conocimiento al que se le apliquen. Luger et al (1989:336-359) indica que los tipos de representación por redes más importantes son las *redes semánticas* y *la teoría de la dependencia conceptual*.

Las redes semánticas representan conocimiento como grafos, con nodos correspondientes a conceptos y arcos representando asociaciones entre ellos.

La teoría de la dependencia conceptual desarrollada por Roger Schank, por su parte, ofrece un conjunto de cuatro conceptualizaciones primitivas del mundo al que se le incorpore. Todas estas son iguales e independientes. Estas, según Luger et al (1989:345), son definidas así:

ACTs representan acciones

PPs representan objetos

AAs modificadores

PAs modificadores de objetos.

Todas las acciones son reducidas a una o más primitivas de

tipo ACTs. Estas, mostradas a continuación, toman los componentes básicos de la acción:

ATRANS transferencia de una acción (dar)

PTRANS transferencia de una localización física de un objeto (ir)

PROPEL aplicar fuerza física a un objeto (presionar)

MOVE mover un objeto (mover)

GRASP agarrar un objeto (agarrar)

INGEST ingerir un objeto (comer)

EXPEL expulsar un objeto (expulsar)

MTRANS transferir información pensada (hablar o decir)

MBUILD crear mentalmente nueva información (decidir o pensar)

CONC conceptualizar o pensar una idea (pensamiento)

SPEAK producir sonidos (hablar)

ATTEND escuchar o poner atención.

Estas primitivas son usadas para definir *relaciones de dependencia conceptual*, reglas que describen estructuras de significado como relaciones de caso o la asociación de objetos y valores.

La dependencia conceptual constituye una teoría formal sobre la semántica del lenguaje natural y provee una forma canónica del significado de las oraciones. Todas las oraciones con significado igual son representadas como sintácticamente idénticas, mientras que los grafos representan

los equivalentes semánticos.

4. <u>Representaciones estructuradas</u>. Las principales estructuras que se utilizan para representar conocimiento son las siguientes:

a. Frames: El concepto de frames fue expuesto por Marvin Minsky en 1975 en el artículo titulado "A Framework for Representing Knowledge", en *The Psychology of Computer Vision*. Un frame puede ser visto como una estructura estática que representa situaciones estereotipadas, bien definidas. Unido a cada frame hay otro tipo de información, la que se refiere al uso del frame, que es utilizada para indicar qué situación se espera que ocurra e indica si las expectativas son o no confirmadas.

Un frame puede ser una red de nodos relacionados. Hay niveles superiores que permanecen fijos y representan situaciones que siempre son verdaderas bajo ciertas circunstancias. Los niveles inferiores tiene varios terminales (llamados "slots"), a los que les son asignados ejemplos y valores específicos. Cada terminal tiene condiciones específicas para sus asignaciones, que se deben cumplir. Estas asignaciones constituyen subframes.

Harris (1985:289) nos indica que Minsky dio una lista de posibles tipos de frames comunes:

i. Frames de sintáctica superficial: almacenan casos

para verbos e indicadores preposicionales.

ii. **Frames de semántica superficial:** agrupan el significado de palabras, calificadores y relaciones entre participantes, acciones y objetivos.

iii. **Frames temáticos:** agrupan temas y entendimiento de problemas.

iv. **Frames narrativos:** historias, explicaciones y argumentos son almacenadas en este tipo de frames.

En la fig. 3.3, podemos observar cómo un cuarto de hotel puede ser descrito por un número de frames individuales. Un frame representa la cama, otros la silla y el teléfono, con valores asignados individualmente. Cada frame individual es visto como una estructura de datos, similar a un "record", que contiene información relevante a entidades estereotipadas.

Los slots de cada frame pueden contener la siguiente información:

1. Información que identifica al frame.
2. Relación del frame con otros frames. Ejemplo, el frame que identifica a la cama se relaciona con el del colchón.
3. Descriptores de requerimientos o partes del frame. Por ejemplo en el descriptor de la silla se indica que ésta posee cuatro patas.
4. Información sobre el uso del frame, que constituye una

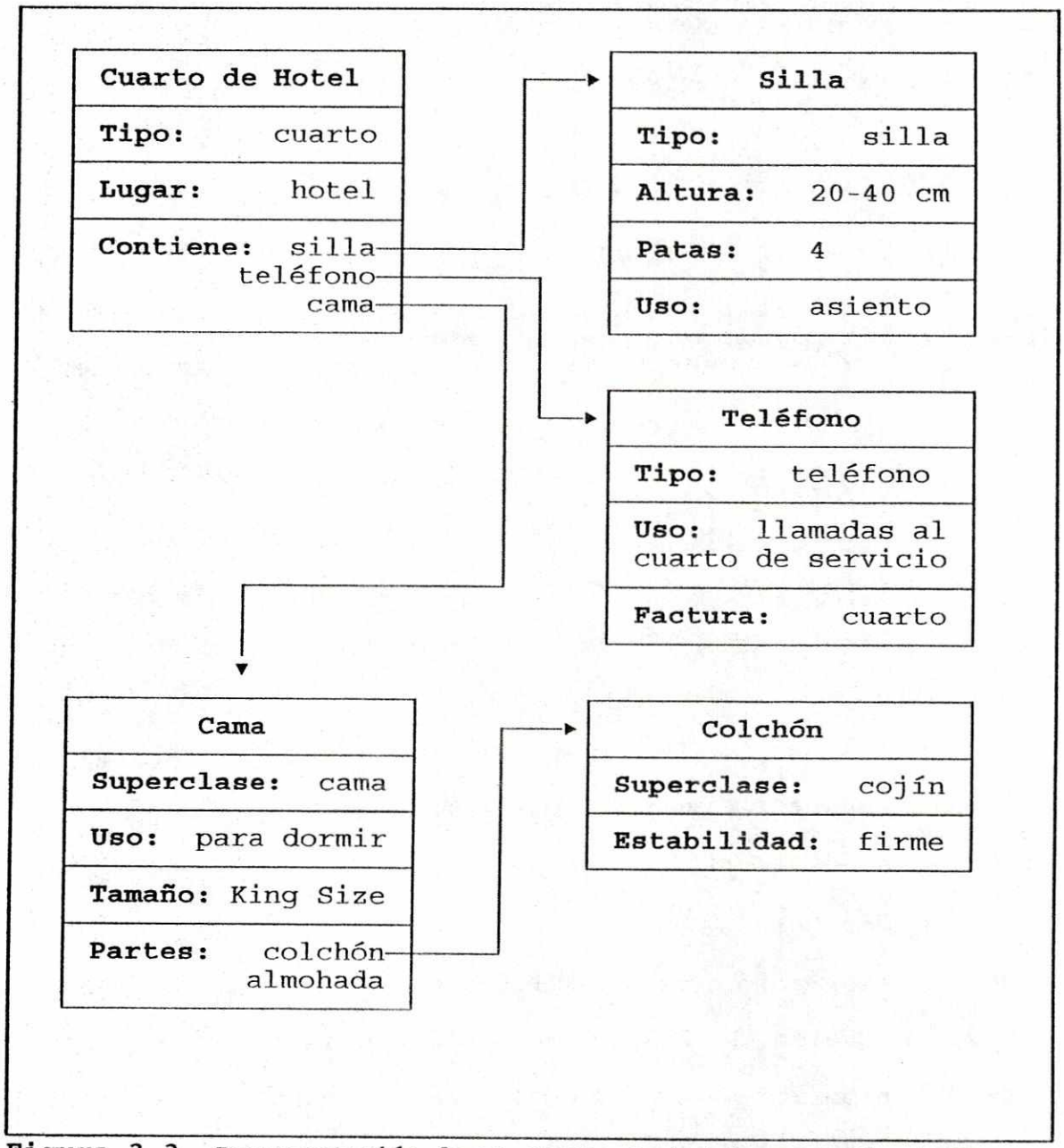


Figura 3.3 **Representación de un cuarto de hotel por medio de Frames. Fuente Luger et al (1989:360).**

importante característica. Ejemplo, el frame sobre la

silla indica que ésta es para sentarse.

5. Información por default. La silla, por ejemplo, tiene cuatro patas desde su fabricación.
6. Información nueva puede surgir y, hasta que no se necesite, puede no ser definida.

A pesar de su enorme utilidad, los frames poseen ciertas dificultades:

1. Un sistema de frames no puede representar las diversas situaciones prototipo, ya que la mayoría son producto de la interrelación entre sus componentes.
2. Los frames generalmente representan situaciones estáticas, es decir que no varían ante determinados eventos.

b. **Scripts:** Los scripts representan situaciones dinámicas, que cambian con determinados eventos. La teoría de los scripts fue desarrollada por Roger Schank y Robert Albenson, basada en su insatisfacción con el uso de la memoria semántica, que consiste en el almacenamiento de palabras organizadas jerárquicamente. Para Schank y Albenson, la memoria humana no consiste de palabras almacenadas, sino de conceptos organizados.

La teoría se basa en el hecho de que muchas situaciones son consideradas normales y predecibles, por lo que se pueden definir expectativas y objetivos que representen estas situaciones.

Los scripts organizan estructuras de dependencia conceptual, en descripciones de situaciones típicas. Por ejemplo, según Harris (1985:300) y Luger et al (1989:365), un script puede proporcionarnos información sobre las acciones o eventos que pueden ocurrir en un restaurante/1/:

Script: **RESTAURANTE**
Area: Cafetería
Componentes básicos: mesas
sillas

comida
cuentas
dinero

Actores: **A** = Consumidor
B = Camarero
C = Cocinero

Condiciones de entrada: **A** tiene hambre
A tiene dinero

Resultados: **A** tiene menos dinero
A no tiene hambre
A está satisfecho

Escena 1: Ingreso

A PTRANS A ingresa al restaurante
A ATTEND ve mesas disponibles

A MBUILD donde sentarse
A PTRANS A hacia la mesa
A MOVE A para sentarse en la silla

Escena 2: Ordenando

A PTRANS menú a **A**
B PTRANS B hacia la mesa

A MBUILD elige su comida
A MTRANS ordena a **B**
B PTRANS B a la cocina
B MTRANS (ATRANS C) en la cocina
C hace script **PREPARAR COMIDA**
C ATRANS comida a **B**
B PTRANS B hacia la mesa
B ATRANS comida a **A**

/1/ La definición de las primitivas para las acciones contenidas en este script fue descrita en la representación de conocimiento por redes, en este mismo capítulo.

Escena 3: Comiendo

A PTRANS A la comida
A INGEST A la comida

Escena 4: Salida del restaurante

A MTRANS pide cuenta a **B**
B MBUILD la cuenta
B ATRANS la cuenta a **A**
A ATRANS paga a **B**
A PTRANS A fuera del restaurante

Los componentes del script anterior son:

Condiciones de entrada o descriptores del mundo, verdaderos para el script llamado. En el ejemplo, es la información sobre el restaurante y el consumidor.

Resultados o hechos que son verdaderos a los scripts finalizados. En el ejemplo puede ser la satisfacción del cliente.

Objetos que intervienen en el script. Por ejemplo sillas, mesas y menús.

Roles o acciones que los individuos realizan. Por ejemplo, el consumidor ordena, come y paga.

Escenas. Schank divide a los scripts en secuencias que representan situaciones temporales que acontecen.

Debido a que los scripts son dinámicos, extienden más aún el entendimiento del lenguaje natural por medio del computador; sin embargo los scripts tienden a representar eventos que pueden no indicar el comportamiento normal de una

situación tratada.

IV. SISTEMAS PARA PROCESAMIENTO DE LENGUAJE NATURAL

Según Harris (1985:315), cualquier sistema de lenguaje natural diseñado para la manipulación y entendimiento del lenguaje, debe ser capaz de: **(a)** aceptar ingresos en texto escrito en lenguaje natural, **(b)** almacenar conocimiento relacionado al dominio de aplicación, **(c)** trazar inferencias de este conocimiento, **(d)** responder a preguntas basadas en el conocimiento y **(e)** generar respuestas apropiadas. Para la comprensión de la estructura y operación de dicho sistema, se presenta a continuación una descripción de las partes que lo integran.

A. Descripción general

Un sistema para el procesamiento de lenguaje natural (**NLPS**, **N**atural **L**anguage **P**rocessing **S**ystem), es un sistema basado en conocimiento para el entendimiento del lenguaje. El conocimiento es almacenado en una base de datos, la cual es precompilada, es decir que el conocimiento del dominio que se esté trabajando existe antes de que el procesamiento ocurra.

El sistema acepta ingresos que son manipulados para asegurar un entendimiento apropiado, declaraciones que representan conocimiento a ser aprendido, y preguntas a ser respondidas accesando este conocimiento. La salida del

sistema son respuestas apropiadas a estos ingresos.

Los módulos principales del sistema son *el parser, el traductor* y *el generador*. El parser acepta las oraciones de ingreso y las mapea en una estructura interna (**MREP**, **M**eaning **REP**resentation), compatible con la base de conocimiento. El generador procesa esta estructura interna para generar salidas apropiadas. El traductor accesa la base de conocimiento para obtener datos del mismo, trazar inferencias o simplemente para agregar conocimiento (ver fig. 4.1).

El conjunto de datos necesitados por el NLPS incluyen la base de conocimiento, el lexicón, los textos de entrada y salida, y la representación semántica para las declaraciones de entrada. La base de datos representa la memoria del sistema y el lexicón contiene el vocabulario del mismo.

1. **El Parser**. El parser acepta las oraciones que han sido ingresadas por el usuario como líneas de caracteres, los cuales son divididos en elementos sintácticos. Posteriormente, se realiza el análisis morfológico para identificar formas regulares y la información del lexicón es accesada para cada palabra analizada. El objetivo de esta fase es convertir el ingreso en una representación que pueda ser procesada. Los distintos tipos de parsers y las distintas técnicas de parseo que fueron analizadas en secciones anteriores, ayudan a un análisis más profundo del texto

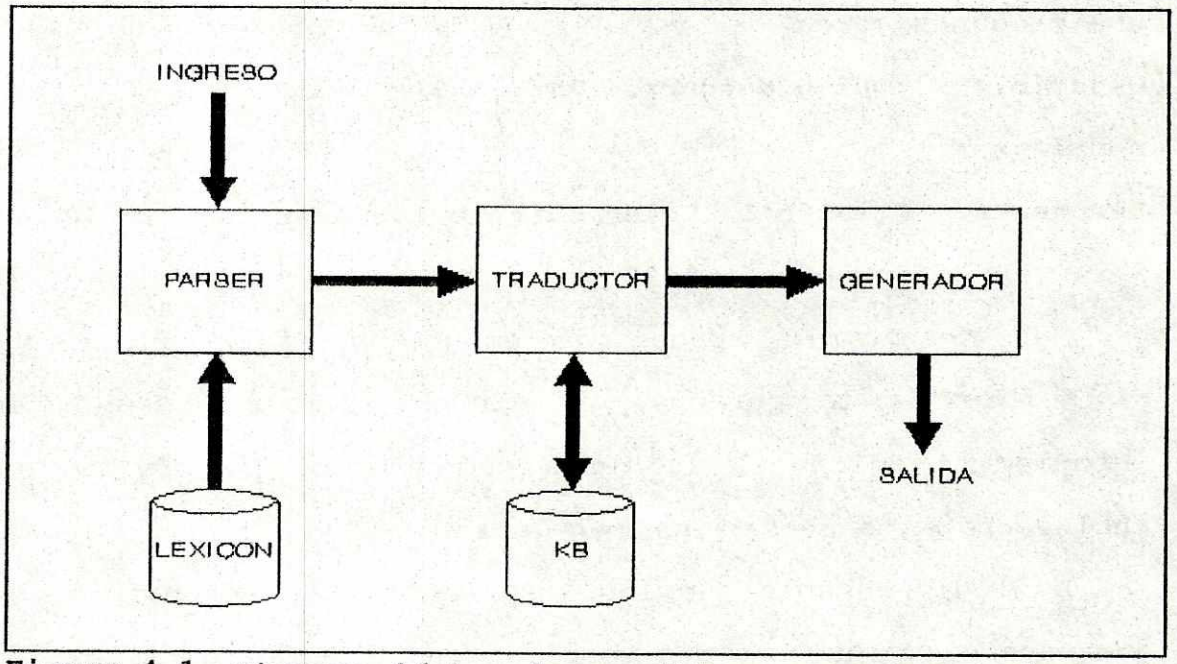


Figura 4.1 **Diagrama básico de un Sistema para Procesamiento de Lenguaje Natural. Fuente Harris (1985:316).**

procesado.

2. **El Generador.** El generador produce las salidas para el usuario. En ciertas circunstancias son respuestas que indican que el ingreso ha sido aceptado y procesado. En otras son respuestas apropiadas que se refieren a la información que ha sido procesada. Por ejemplo, la respuesta a una declaración que agrega conocimiento a la base datos de conocimiento puede ser:

Respuesta: Ingreso Aceptado

Y según Kerschberg (1986:617), una respuesta apropiada a

una pregunta como

Usuario: ¿Quién descubrió América?

podría ser

Respuesta: Cristóbal Colón descubrió América el 12 de octubre de 1492.

En este ejemplo, aunque la pregunta se refiere únicamente al nombre de la persona, el sistema también posee como información adicional la fecha del evento, por lo que se incluye para completar aún más la respuesta.

Es indispensable que las respuestas sean apropiadas, ya que un sistema que sea más cooperativo provee más información al usuario que lo emplee.

3. **El Traductor.** El traductor tiene como función el acceso a la base de datos de conocimiento, incluyendo las adiciones o modificaciones necesarias. La función a desempeñar por el traductor depende del tipo de declaración ingresada. Si la declaración es una pregunta, el traductor determina cómo ésta deberá ser respondida. La recuperación de información es obtenida accesando la estructura de la base de conocimiento y, las inferencias de la información, son llevadas a cabo operando las reglas definidas en la estructura de conocimiento.

4. La Base de Conocimiento. La base de conocimiento (**KB**, **K**nowledge **B**ase), según Harris (1985:318), contiene básicamente tres tipos de datos: *entidades*, *eventos* y *situaciones*, los cuales son llamados *paquetes*.

Las entidades son objetos, personas o lugares, algo que pueda ser descrito o de que se pueda hablar en la oración. Cada entidad tiene un nombre y un tipo que la identifican, pero son definidas por su lista de atributos. Los atributos son algún tipo de información que pueda ser descrita, como localizaciones, tamaños, duraciones o relaciones con otras entidades.

Los eventos son acciones que ocurren involucrando actores y objetos, objetivos, fuentes e instrumentos. Las situaciones combinan entidades y eventos, con condiciones y resultados.

5. El Lexicón. El lexicón contiene el vocabulario del NLPS, y además la información sintáctica, semántica y pragmática/2/, que es necesaria para el análisis de cada elemento léxico por medio del sistema.

La función primaria del lexicón es la de asistir al parser

/2/ Información pragmática se refiere a condiciones que indican cuando una frase u oración funciona apropiadamente dentro del contexto del lenguaje natural (ver Niveles de análisis en el capítulo II).

en la traducción del texto de ingreso, a una representación interna a ser procesada por el computador. Cualquier palabra usada en el ingreso debe ser localizable en el lexicón.

B. Implementación de un NLPS

Según Harris (1985:325), debido a la complejidad que involucra el procesamiento de lenguaje natural, la forma de implementación puede ser de la siguiente forma:

1. **Modo de parafraseo o traducción.** En este tipo de implementación, los ingresos son analizados y almacenados en una representación que expresa su significado, para inmediatamente ser producida como salida. Si la salida involucra un lenguaje distinto al que se procesa, al proceso se le denomina *traducción*. Por ejemplo, si se ingresa una oración en español y la salida es en inglés, el NLPS es un traductor español-inglés.

2. **Modo de inferencia.** Existen tres tipos de inferencia que son posibles en un NLPS: **(1)** inferencia que asigna o completa estructuras (como frames y scripts), **(2)** inferencia que se efectúa sobre las estructuras de la base de datos de conocimiento y **(3)** inferencia sobre las afirmaciones que se hacen respecto del sujeto.

El primer tipo de inferencia analiza las oraciones

tratando de asignar valores a estructuras vacías asociadas con un caso particular, como las oraciones que incluyen pronombres.

El segundo tipo de inferencia accesa la base de conocimiento y analiza situaciones que representen varios pasos, los cuales se encuentran representados en la base. Como por ejemplo, podemos considerar un tipo de actividad deportiva para la cual se incluye información sobre las reglas del juego, popularidad y estrellas o protagonistas.

El tercer tipo de inferencia analiza las afirmaciones o declaraciones contenidas en la base de conocimiento. Estas pueden ser las entidades o lista de atributos. Un ejemplo sería, según Harris (1985:330):

Todos los negocios bien administrados generan ganancias. Este puede ser un atributo perteneciente a la lista de atributos sobre actividades económicas, que a su vez se relacionan con administración.

3. **Modo de respuesta a preguntas.** Las preguntas formuladas formuladas son analizadas para determinar su tipo, y luego se accesa la base de conocimiento para encontrar su respuesta. En el caso de un sistema basado en reglas semánticas, las reglas pueden representar las posibles preguntas a las que puede responder el sistema.

4. **Modo de aprendizaje.** Este paso consiste en agregar conocimiento a la base de datos. El conocimiento es adquirido para cada declaración que se ingrese, el cual es almacenado en la base de conocimiento. El proceso principia convirtiendo la declaración a una representación interna de su significado (MREP), la cual es pasada al traductor que determina si la información contenida en la MREP es relevante en la base de conocimiento (KB). Si esta información es relacionada con algo existente en las listas de conocimiento de la KB, es agregada a dichas listas, de lo contrario, se crean nuevas listas que relacionan este nuevo conocimiento.

C. Estructuras de control para un NLPS

Como lo indican Harris (1985:339) y Kerschberg (1986:620), los módulos del NLPS (parser, traductor y generador) implican que un control secuencial es empleado. Los ingresos son manejados por el parser, el cual pasa una representación al traductor, que la emplea para accesar la base de conocimiento y éste a su vez pasa la información al generador, que produce la salida apropiada. Unicamente cuando este ciclo es completado, nuevos ingresos pueden ser aceptados. En los sistemas actuales, este tipo de control no siempre es apropiado, ya que en ocasiones el parser también necesita información que se encuentra en la base de conocimiento.

Harris (1985:339) presenta una breve descripción de

algunas estructuras de control que se han implementado en el procesamiento de lenguaje natural.

1. **Control Secuencial.** Este control sigue la lógica propuesta por las teorías de algunos lingüistas que creen que el análisis sintáctico debe ser separado del semántico. Este control es simple, cada módulo ejecuta una secuencia. Sin embargo, esto no es suficiente, ya que no siempre es posible completar el análisis sintáctico antes del análisis semántico.

2. **Control Predictivo.** El método *predictivo* o *top-down2* empieza a trabajar asumiendo que un patrón particular debe ser examinado y confirmado en los niveles inferiores. Si el examen falla, se debe regresar y proceder con las otras pruebas disponibles.

3. **Control Jerárquico.** En este tipo de control, los módulos interactúan entre sí. Los módulos funcionan como co-rutinas ejecutadas al mismo tiempo, las cuales pasan mensajes de regreso a los módulos que las llamaron. La desventaja es que se pierde una representación limpia del proceso del NLPS, ya que es difícil analizar la forma en que es operado un determinado ingreso.

4. Modelo Locus. Esta técnica ha sido más exitosa (bajo ciertas circunstancias) que los métodos de control mencionados anteriormente. Consiste en compilar toda la información léxica, sintáctica y semántica en una red, que puede ser similar a un ATN. Esta red es accesada durante el análisis de los elementos de la oración y para construir su representación interna.

Este método es bastante eficiente, ya que mucho del análisis ha sido hecho previamente en el proceso de compilación, pero es algo restrictivo, debido a que ningún cambio puede ser hecho a la gramática o al lexicón, sin recompilar la red.

V. INTERFASES EN LENGUAJE NATURAL PARA ACCESAR BASES DE DATOS

A. Antecedentes

Anteriormente se mencionó que uno de los mayores "cuellos de botella" para el entendimiento de lenguaje natural, lo constituye la adquisición de suficiente conocimiento acerca del dominio en el que se emplee. También se dijo que la tecnología presente está limitada a direccionar dominios muy bien definidos. Pues bien, un área de aplicación que se apega a esta tecnología lo constituye el desarrollo de interfases en lenguaje natural para accesar bases de datos.

A pesar que las bases de datos guardan una gran cantidad de información, ésta es bastante regular y bien definida. Estas características, junto a la utilidad que proporciona el lenguaje natural para accesar información, hacen de estas interfases un área importante de aplicación del procesamiento de lenguaje natural.

Según Obermeier (1990:217), un sistema de consultas para accesar bases de datos por medio de lenguaje natural (**NLQS**, **N**atural **L**anguage **Q**uery **S**ystem) provee fácil acceso a un sistema de bases de datos (**DBMS**, Data **B**ase **M**anagement **S**ystem), ya que las consultas son efectuadas en la forma normal en que se hacen preguntas: lenguaje natural.

Un NLQS traslada las declaraciones efectuadas en algún

idioma como español o inglés, a instrucciones de un determinado lenguaje de manipulación de datos (**DML**, **D**ata **M**anipulation **L**anguage) que utilice el DBMS (ver fig. 5.1).

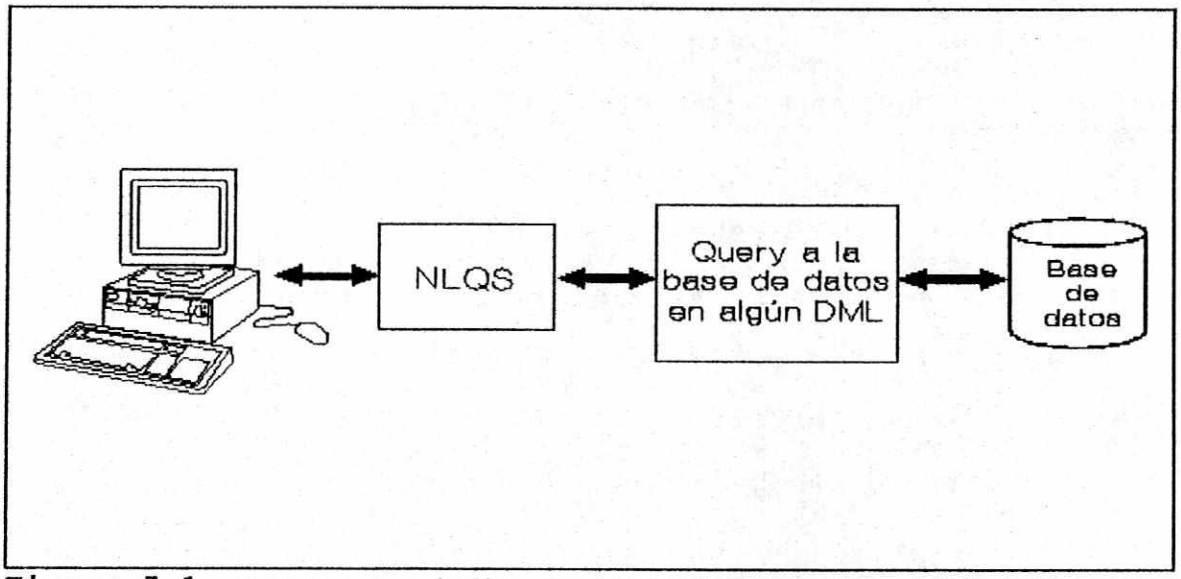


Figura 5.1 **Esquema que indica las funciones básicas efectuadas por un NLQS. Fuente Obermeier (1990:218).**

Si se instala, integra y usa apropiadamente, un NLQS puede resolver los problemas de recuperación de información de un DBMS.

B. Clasificaciones de un NLQS

Como lo indica Obermeier (1990:222), estructuralmente los NLQS se clasifican en tres grupos de acuerdo a su tipo de tecnología: **(1)** sistemas basados en pattern-matching, **(2)** sistemas sintácticos y **(3)** sistemas basados en conocimiento.

Un sistema basado en pattern-matching reduce el lenguaje natural a frases almacenadas. Un NLQS orientado a sintaxis utiliza un análisis gramatical profundo para procesar los ingresos efectuados en lenguaje natural. Un sistema basado en conocimiento define un dominio específico de la base de datos y efectúa inferencias sofisticadas sobre el mismo.

Asimismo Obermeier (1990:222) clasifica, funcionalmente, los NLQS en dos grupos:

1. Sistemas basados en la información contenida en el diccionario perteneciente a la base de datos.
2. Sistemas que poseen su propio conocimiento de la información almacenada en la base de datos.

Los sistemas basados en el diccionario de la base de datos están circunscritos a la información contenida en él, y no poseen la habilidad de distinguir entre consultas lógicas e ilógicas de la información. En cambio, los sistemas basados en conocimiento definido sobre la base de datos pueden efectuar inferencias sobre los datos, otorgando mayor flexibilidad a las consultas. Sin embargo hay dos factores críticos que pueden ser considerados como desventajas: la definición inicial del dominio de trabajo y el mantenimiento del mismo. Ambos sistemas implican un tiempo de desarrollo prolongado, que en algunas situaciones puede ser considerado como costoso e impráctico.

C. Componentes de un NLQS

En Kerschberg (1986:620) se explica que un NLQS es un sistema modular con componentes bien definidos, ya que es necesario separar la visión conceptual que tiene el usuario del dominio de trabajo, de los detalles particulares de la estructura de la base de datos y su sistema de procesamiento. Los resultados del parseo y la interpretación semántica son expresados en términos de la estructura conceptual del usuario, lo que permite modificar la estructura de la base de datos sin tener que cambiar el sistema de entendimiento de lenguaje natural. Esta representación hace posible determinar si los ingresos del usuario están adecuadamente representados antes de accesar la base de datos.

Los componentes de un NLQS son similares a los NLPS, pero a diferencia de este, no efectúan un análisis sintáctico formal y, poseen un dominio de trabajo bien definido. Estos componentes, según Kerschberg (1986:620-621), son:

a. El parser.

b. El interpretador semántico.

c. El módulo de adquisición de conocimiento.

d. La base de conocimiento.

e. El generador.

En la fig. 5.2 podemos observar cada uno de los componentes de un NLQS y la relación funcional entre ellos. Las elipses representan componentes de la base de datos de

conocimiento, mientras que los cuadros representan los distintos módulos funcionales. **MRL** son las iniciales para el lenguaje de representación interna que utiliza el NLQS para mapear los ingresos análizados por el parser y el interpretador semántico. Este será explicado más adelante con mayor detalle.

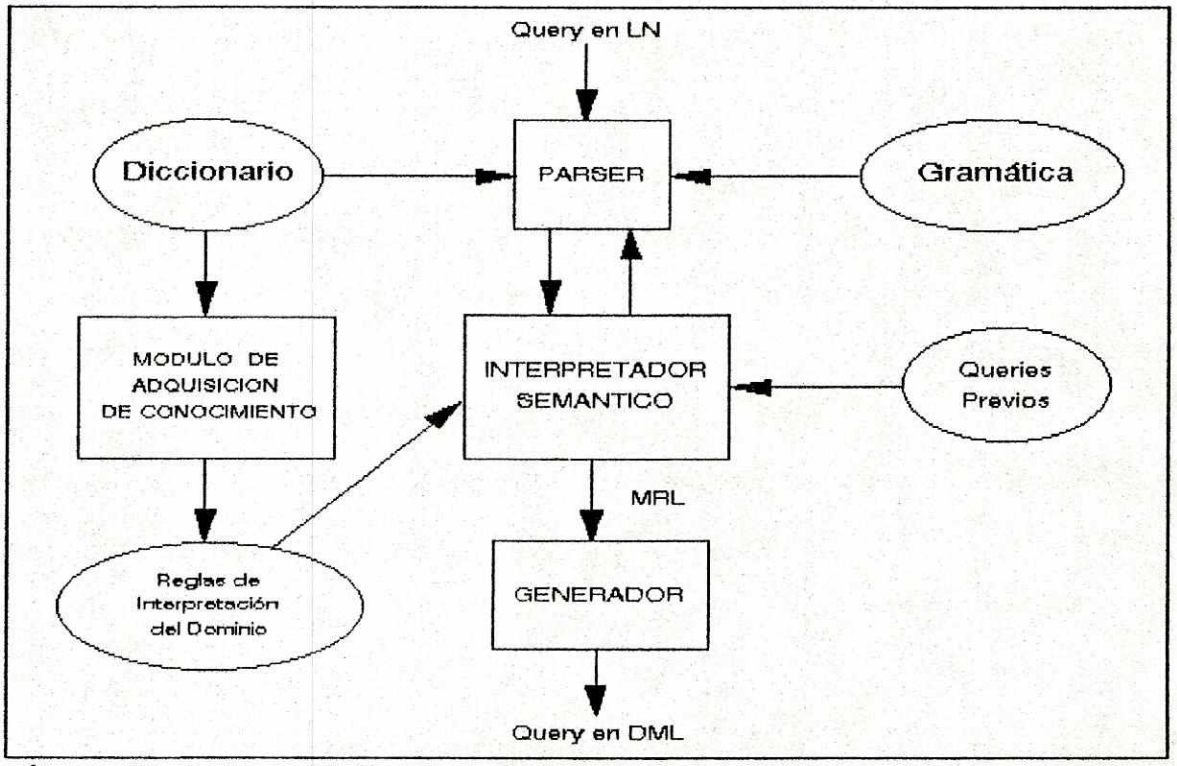


Figura 5.2 Componentes de un NLQS. Fuente Kerschberg (1986:621).

1. **El Parser.** El parser es el encargado de leer e interpretar los distintos ingresos del usuario, ya sean preguntas o declaraciones.

El parser accesa un grupo de **reglas de gramática**, particulares del lenguaje natural a utilizar, y un **diccionario** o **lexicón**, en donde se encuentra definido el vocabulario que utiliza el lenguaje natural.

El parser puede usar algunas de las técnicas de parseo descritas con anterioridad, pero generalmente usan ATNs, ya que constituyen una técnica estándar para el parseo de lenguaje natural; además de permitir mayor flexibilidad en la manipulación de las distintas gramáticas involucradas.

2. **Interpretador semántico.** El interpretador semántico es, en síntesis, un traductor, que se encarga de recibir los datos enviados por el parser y decidir qué acción tomar.

El interpretador semántico accesa un conjunto de **reglas de interpretación del dominio (conocimiento semántico)**, las cuales representan la visión conceptual que el usuario tiene de la base de datos, y la forma en que dicha conceptualización es relacionada con la misma. El conocimiento semántico es usado por el traductor para decidir qué frases u oraciones poseen significado y computar el significado de las mismas.

El traductor, junto con el parser, tienen como objetivo procesar el query o comando en lenguaje natural, sin accesar la base de datos del dominio. Este nivel de entendimiento es definido formalmente en un lenguaje de representación de

significados (**MRL**, **M**eaning **R**epresentation **L**anguage).

El MRL hace posible definir las operaciones del sistema en algún procesador de bases de datos, que provea generadores para las entidades definidas por frases sustantivas (sujetos), y procesos (representados por los predicados) para procesar estas entidades.

3. **Módulo de adquisición de conocimiento.** Este módulo permite al usuario crear entradas al diccionario e ingresar reglas semánticas para extender el dominio.

El proceso de adquisición léxica permite a los usuarios definir las partes del lenguaje y las características sintácticas de las palabras.

El proceso de adquisición de reglas semánticas permite definir el conocimiento semántico de la base de datos, es decir, la vista conceptual que tiene el usuario de los datos.

Mucho del conocimiento necesitado por el procesador de lenguaje natural es específico al lenguaje y no al dominio de trabajo. Esto incluye información sintáctica y semántica. La gramática del lenguaje y las características sintácticas de las palabras son independientes del dominio. Los cuantificadores semánticos, determinantes y otras partes del lenguaje, tienden también a ser independientes del dominio, y son definidos en el procesador de lenguaje natural, y no en el módulo de adquisición de conocimiento.

Como lo indica Kerschberg (1986:624), hay tres tipos de conocimiento semántico dependiente del dominio, que debe ser adquirido:

i. **Modelo del dominio:** son las cosas que pueden ser habladas acerca del dominio, y las relaciones entre las clases semánticas. Una clase semántica es un conjunto de cosas en el dominio.

ii. **Interpretabilidad:** son las posibles construcciones sintácticas que pueden ser hechas, el tipo de cosas que deben rellenar cada vacío en una frase, para que ésta pueda poseer significado en el contexto del dominio.

iii. **Predicación:** es la forma en que los constituyentes de una frase son combinados para poder interpretarla.

4. La Base de Datos de Conocimiento. Kerschberg (1986:624) indica que hay algunos elementos almacenados en la base de datos de conocimiento que contribuyen al análisis sintáctico, semántico y al procesamiento de enunciados o discursos:

a. Un diccionario de palabras que forman parte del lenguaje natural empleado (es ampliamente dependiente del dominio).

b. Una gramática, lo más completa posible, del lenguaje natural utilizado (no cambia de un dominio a otro).

c. El conjunto de reglas semánticas (altamente dependiente del dominio de trabajo).

d. Las representaciones en el MRL, de algunas consultas (queries) previas hechas por el usuario.

5. El Generador. El objetivo de este módulo es la de convertir las representaciones expresadas en el MRL, en operaciones de acceso a la base de datos, utilizando su lenguaje de procesamiento (DML). Este módulo es ad-hoc al DML definida para el tipo de base de datos que se esté empleando, ya sea relacional o no.

Este módulo genera todas las salidas para el usuario, en ciertas circunstancias son respuestas que indican que el ingreso ha sido aceptado y procesado. En otras son respuestas apropiadas que se refieren a la información que ha sido procesada. Este módulo posee un sistema sofisticado de manipulación de errores, así como la capacidad de proveer información adicional sobre los datos, en vez de responder literalmente a las preguntas.

D. Bases de Datos Relacionales y SQL

Date (1987:19) define una *base de datos relacional como un sistema que* organiza los datos en relaciones a través de dominios de entidades. Estas relaciones lo forman un grupo de archivos planos llamados *tablas*, cuyos registros reciben el nombre de *atributos*, los cuales se dividen a su vez en campos o *tuplas* de algún tipo, que constituyen los atributos de las

relaciones.

La teoría sobre bases de datos relacionales fue desarrollada por E. F. Codd en su artículo titulado *A Relational Model of Data for Large Shared Data Banks*, en 1970. La teoría propone recuperar información de las bases de datos, en base a cálculo relacional, el cual consiste en aplicar reglas y operaciones matemáticas a las relaciones de la base de datos.

Hoy día los sistemas de bases de datos relacionales son los sistemas de más amplio uso en el comercio y la industria. Sistemas como el IBM DB2, DEC RDB, Ingres, Oracle e Informix son ejemplos de algunos de ellos. Sin embargo, algo muy importante de mencionar, es que la mayoría de estos sistemas relacionales utilizan un lenguaje común para procesarlas. Este lenguaje es denominado **SQL**.

SQL (**S**tructured **Q**uery **L**anguage), según Date (1987:95), fue definido por Chamberlin y otros en el IBM Research Laboratory en San José, California. Una implementación prototipo fue construida en el mismo laboratorio en 1975, bajo el nombre de "System R". Desde su introducción comercial en 1979, SQL ha sido adoptado por varias compañías como un lenguaje estándar para manejar bases de datos relacionales.

Debido a la importancia que SQL ha adquirido, una gran mayoría de los NLQS actuales lo utilizan como el lenguaje a producir, luego del análisis de los requerimientos escritos en

lenguaje natural.

A pesar de las facilidades provistas y un uso más frecuente de SQL, las bases de datos relacionales siempre están sujetas a discusión, debido a que son construidas para procesamiento eficiente, en vez de un uso efectivo. Los NLQS pueden proveer un acceso más amigable, lo que redunda en una mayor productividad y uso efectivo de estos sistemas.

1. **El lenguaje SQL.** Como lo indica Date (187:95-154), SQL es un lenguaje para definición de datos (**DDL**, Data Definition Language), así como también un lenguaje para la manipulación de datos (**DML**, Data Manipulation Language). SQL provee además un conjunto de instrucciones para definir seguridad y tipos de acceso a los datos.

La definición de datos provee básicamente las siguientes instrucciones:

CREATE TABLE **CREATE INDEX** **CREATE VIEW**

ALTER TABLE

DROP TABLE **DROP INDEX** **DROP VIEW**

En donde CREATE se usa para crear tablas, índices y vistas (tablas virtuales), ALTER para alterar atributos de las columnas de las tablas y DROP, para eliminar tablas, índices y vistas.

SQL soporta los siguientes tipos de datos:

INTEGER números enteros

SMALLINT	números enteros pequeños
FLOAT	números de punto flotante
CHAR (n caracteres)	cadenas ("Strings") de caracteres.
DECIMAL (p, q)	números decimales empaquetados de largo p y con q decimales

La manipulación de datos provee cuatro instrucciones:

SELECT UPDATE DELETE INSERT

INSERT se usa para agregar nuevos registros, UPDATE para modificarlos, y DELETE para eliminarlos. Las tres operaciones son empleadas en una sola tabla.

SELECT es utilizado para recuperación de información, y constituye una de las instrucciones más importante que posee SQL.

La sintaxis general del SELECT es la siguiente:

```
  SELECT [ DISTINCT ] campo(s)
  FROM   tabla(s)
[ WHERE  expresión predicativa
[ GROUP BY campo(s) [ HAVING expresión predicativa ]
[ ORDER BY campo(s) ]
```

En donde las tablas y sus campos fueron definidos utilizando el DDL y las expresiones predicativas son expresiones compuestas por operadores lógicos (**AND**, **OR**, **NOT**, **IN** y **EXIST**) y comparativos (**=**, **<>**, **>**, **<**, **>=**, **<=**, **<>**, **LIKE**).

El SELECT presenta las siguientes características:

a. Recupera información de múltiples y distintas tablas, lo que permite efectuar *una gran variedad de joins* (unión de

una tabla con una o más tablas).

b. Provee algunas funciones básicas incorporadas, que pueden ser utilizadas en expresiones predicativas. Estas funciones son:

COUNT número de valores en una columna

SUM suma de los valores de una columna

AVG promedio de los valores en la columna

MAX el valor máximo de una columna

MIN el valor más pequeño de una columna.

c. Por medio de las cláusulas **GROUP BY** y **HAVING**, así como **ORDER BY**, se puede agrupar y ordenar la información, lo que facilita la creación de reportes.

d. Una importante característica del SELECT de SQL, es que posee un optimizador de código, con lo que el trabajo de efectuar todas las relaciones e interpretar las restricciones impuestas a éstas, decidir si se usa o no un determinado índice, los joins que se deben efectuar, es realizado por el DML, lo que hace sencilla la recuperación de información en bases de datos relacionales.

Como ejemplo, supongamos que tenemos definida una tabla de empleados, compuesta por los campos nombre y salario, si deseamos obtener los nombres y el salario de los empleados que ganan más de 1000.00, la consulta podría se ejecutaría con el siguiente SELECT:

```
SELECT nombre, salario
FROM   empleados
WHERE  salario > 1000.00
```

Como se ve la elaboración de la consulta, no sólo fue sencilla, sino también bastante lógica. El SELECT constituye la instrucción más utilizada del SQL, y debido a la importancia que este lenguaje ha despertado como estándar para el intercambio de datos, sin importar el tipo y manejador de base de datos que se esté utilizando, el SELECT es la clave para el éxito de cualquier DML que desee intercambiar información utilizando SQL.

F. Modelos de procesamiento para NLQS

El procesamiento de lenguaje natural para accesar bases de datos ha sido ampliamente desarrollado en las últimas dos décadas, disponiéndose de algunos modelos, de los cuales dos son los más utilizados:

1. **NLQS basado en Gramática Semántica.** Este modelo consiste de dos partes fundamentales: **(a)** un lexicón y **(b)** una serie de reglas de reescritura.

En este modelo, cuyos datos son mencionados principalmente por Obermeier (1985:227) y Laurel (1990:400), los verbos y los nombres son reemplazados por clases semánticas específicas, las cuales a su vez forman parte de las reglas de reescritura, a las que denominaremos *reglas semánticas*.

Una de las más eficientes implementaciones de este modelo lo constituye el programa **LADDER**, desarrollado en 1978 por Gary G. Hendrix. LADDER fue diseñado como un interfase en lenguaje natural para accesar la base de datos de la Marina de los Estados Unidos de Norteamérica.

La estrategia del modelo de reglas semánticas podemos definirla de la siguiente forma:

a. El usuario ingresa una pregunta u oración en lenguaje natural para accesar la base de datos, a la cual se le busca un patrón en el juego de reglas semánticas almacenadas en el lexicón. El proceso de búsqueda para dicho patrón consiste en localizar en el ingreso las clases semánticas que sirven de relleno a las oraciones básicas que se encuentran definidas como reglas semánticas.

b. Luego de encontrada la regla y su patrón específico, se localiza en la base de conocimiento la representación que poseen las clases semánticas encontradas. Con esta representación se sabe qué contraparte poseen en la base de datos de la cual se está extrayendo información.

c. Identificada la oración y su contraparte en la base de datos a consultar, se obtiene el frame semántico que pertenece a la regla semántica localizada, y se procede a rellenar sus slots. Dicho frame contiene el código para accesar la base de datos.

d. Una vez construido el frame, se procede a procesarlo y el resultado constituye la respuesta a la consulta en lenguaje natural.

Como se ve, los componentes principales de cualquier NLQS tienen su equivalente básico. El parser busca los patrones semánticos para las oraciones ingresadas, el interpretador semántico se encarga de obtener las clases semánticas y accesando la base de conocimiento, obtiene su representación en la base de datos de información, con lo cual el generador procede a rellenar los slots del frame semántico, para luego procesarlo y obtener el resultado.

El módulo de adquisición para este modelo tiene la función de adquirir los nombres y su correspondencia en la base de datos a accesar (adquirir las clases semánticas), y obtener el juego de reglas semánticas que sirven para representar los posibles ingresos del usuario.

Las ventajas y desventajas del modelo, mencionadas por Laurel (400-405), se pueden enumerar de la forma siguiente:

i. El tamaño de las clases semánticas es más pequeño que un equivalente sintáctico consistente de verbos y sustantivos, lo que resulta en una estrategia de parseo más eficiente, ya que se chequean un número menor de posibilidades.

ii. En contradicción con la ventaja anterior, el número de reglas semánticas que tratan de dar un patrón a los

ingresos de usuario, puede crecer y superar las expectativas de cualquier diseñador, debido a que la forma de efectuar una pregunta puede tener un número elevado de variantes, que puede ser de unas cuantas, hasta miles.

iii. Otra desventaja consiste en el traslado de las reglas semánticas de un dominio de aplicación a otro.

iv. El mantenimiento de este modelo es dificultoso, ya que para que sea funcional se necesita un juego de reglas semánticas lo más completo posible. Esto provoca que el tiempo de instalación y el costo sean muy elevados.

v. Una justificación para este modelo es que es el modelo que mantiene una secuencia más exacta entre las preguntas y respuestas, con lo que se puede proveer un mejor diálogo con el usuario, y por consiguiente, se provee de un sistema más cooperativo y operacional (Intellect y HPNL, son ejemplos de ello).

vi. Los frames de código para el acceso a las bases de datos, pueden ser en cualquier DML, lo que significa que el modelo posee una amplia gama de aplicación en distintos modelos de procesamiento de bases de datos.

2. **NQLS basado en menús sensibles al contexto.** En el año de 1983, como lo indica Shneiderman (1990:169), se desarrolló una idea innovativa para mezclar las facilidades

que proveen los interfases por medio de menús y el uso de lenguaje natural. El proyecto denominado **NLMENU** fue desarrollado por Tennant, Ross y Thompson, y ahora el producto resultante es distribuido comercialmente por Texas Instruments bajo el nombre de **Natural Link.**

La idea básica es mostrarle al usuario las frases como una secuencia de menús, los cuales son sensibles al contexto. De esta forma se construyen los queries a través de una serie de menús que presentan los nombres, las distintas operaciones y operadores aplicables, y los distintos criterios de selección. Cada selección efectuada tiene su contraparte en lenguaje natural, el cual el usuario puede observar y así comprobar cómo se construye su consulta.

La estrategia de este modelo podemos enumerarla de la siguiente forma:

a. El usuario, por medio de la interacción con los menús, construye una secuencia de instrucciones en lenguaje natural, las cuales muestran coherencia semántica.

b. Al mismo tiempo que se muestra la secuencia en lenguaje natural, se obtiene la correspondencia del lenguaje natural con la información de la base en el lexicón y se rellenan los slots del frame correspondiente, que contiene el código de acceso a la base de datos consultada.

c. Cuando el usuario termina la interacción, se procede a procesar el código de acceso generado y se obtiene la

respuesta al requerimiento.

Como se nota, este modelo es más sencillo, ya que sus componentes son más simples. El parser muestra los distintos menús que componen una gramática básica, obtiene los requerimientos y se encarga que los menús siguientes muestren sensibilidad al contexto. El interpretador semántico utiliza los requerimientos ingresados para obtener la correspondencia con los datos almacenados en el lexicón. El generador construye el código en DML objeto, lo ejecuta y muestra la respuesta obtenida.

El módulo de adquisición de conocimiento únicamente necesita obtener la representación de los datos con los nombres en lenguaje natural.

Las ventajas y desventajas del modelo son enumeradas por Shneiderman (1990:170) así:

i. Este modelo es más sencillo, y permite mostrarle al usuario todas las posibles variantes, y las distintas funciones que se pueden aplicar.

ii. El proceso de definición es mucho más rápido y menos costoso.

iii. La conducta y comportamiento del modelo es mucho más predecible y comprensible que cualquier otro.

iv. Sin embargo, el modelo limita al usuario a utilizar únicamente lo que el diseñador prepare, lo cual es mucho menos natural para él.

v. El modelo sólo permite el uso de una gramática sencilla, la cual debe ser predefinida, con lo que se elimina la posibilidad de una amplia libertad de formulación de requerimientos.

VI. DISEÑO

Para llevar a cabo el desarrollo del interfase en lenguaje natural para accesar bases de datos relacionales, se definieron las siguientes pasos:

A. Selección de modelo a implementar

Para diseñar el interfase, se analizaron los dos modelos más empleados (reglas semánticas y menús sensibles al contexto), y luego de un riguroso estudio, se eligió el modelo de menús sensibles al contexto.

La elección obedece a las siguientes observaciones:

1. La implementación y puesta en práctica del modelo de menús sensibles al contexto es mucho más sencilla que el modelo semántico, requiere de un diseño y programación precisos que permiten un desarrollo concreto.
2. El modelo de menús requiere de un parser más sencillo; puede ser procesado por ATNs simples y recursión descendente.
3. El modelo de reglas semánticas requiere de un parser que siga un conjunto de reglas que tienden a ser largas y tediosas de definir, presentando un gran número de variantes.
4. La generación de código para el modelo de reglas semánticas es más tedioso de implementar. A menudo gran

parte del código se define junto con las reglas semánticas y poco o nada hace el software para generarlo automáticamente en el proceso de definición.

5. En el modelo de menús sensibles al contexto, el proceso de definición es mucho más sencillo, por lo que su instalación y puesta en funcionamiento es mucho más rápida.
6. El proceso de definición del modelo de menús no requiere de personal altamente calificado para la instalación y puesta en funcionamiento del sistema.

B. Objetivos

El interfase, básicamente deberá cumplir con los siguientes objetivos:

1. Procesar una secuencia de instrucciones ingresadas por el usuario por medio de menús sensibles al contexto.
2. Expresar cada uno de estos requerimientos en lenguaje natural, mediante una gramática predefinida.
3. Traducir los requerimientos ingresados a SQL.
4. Desplegar la información obtenida.
5. Permitir la extracción de más información de la respuesta.

C. Especificación de características del interfase

Las características que tendrá el interfase a diseñar son:

1. **Funcionalidad**. El software del interfase debe operar en microcomputadores tipo IBM PC o compatibles, con un mínimo de 640K de memoria y un disco duro. Dicha interfase debe funcionar en el más sencillo de éstos computadores, y debe tener el mismo comportamiento si se le traslada a otro ambiente similar.

2. **Integridad y seguridad**. El software del interfase debe ser sólido, correr con mínimo de hardware y ante errores demostrar buena recuperación, con lo cual el usuario no se sienta desconcertado.

3. **Flexibilidad**. El interfase debe ser fácil de accesar, así como fácil de operar (uso de funciones y distintas teclas del teclado y un mouse si es posible), con un buen sistema de recuperación de errores, una operación consistente (las mismas teclas y funciones para operar, menús similares, etc.) y buen juego de mensajes que guien al usuario durante la operación.

4. **Estructuramiento**. Para que el usuario pueda indicar sus requerimientos fácilmente y asimismo les pueda dar mantenimiento de igual forma, el interfase debe ser estructurada.

5. **Compactibilidad.** Las especificaciones que se le presentan al usuario deben ser simples y concisas, y a su vez deben ser lo suficientemente poderosas para que se pueda obtener el máximo de información disponible.

D. Diseño del software que opera el interfase

Los programas que operan el interfase son dos:

1. **Programa de especificaciones.** El programa de especificaciones sirve para definir todos los elementos relacionados de la base de datos y su correspondencia en lenguaje natural.

Los módulos básicos que debe incluir son los siguientes:

a. **Módulo de definiciones en el Lexicón:** es el módulo que sirve para definir las tablas de datos y los nombres contenidos en ellas, las unidades de medida (metros, litros, kilogramos) y algunos posibles dominios que puedan poseer los datos.

b. **Módulo de definiciones en el Catálogo:** este módulo opcional, sirve para definir bases de datos, junto con sus tablas y vistas. Es opcional porque si el sistema de archivos no posee un catálogo, este módulo permite su implementación.

c. **Módulo de Consultas:** este módulo no es más que el interfase en lenguaje natural, que se incluye para

facilitar el proceso de definición.

2. **Programa de consultas.** Este programa es el interfase en lenguaje natural para accesar las bases de datos. Dicho interfase debe tener las siguientes características:

a. Incluir una gramática incorporada, basada en lenguaje natural, que sea clara, sencilla y compacta y, a su vez, lo suficiente poderosa para que pueda ser utilizada eficientemente.
b. Un buena interacción con el usuario, así como un buen sistema de mensajes.
c. Un buen manejo de errores.
d. Acceso a los datos a través de SQL que es fácil de utilizar, poderoso y, a su vez, bien optimizado.

Dicho interfase se compondrá de 3 secciones:

a. Sección de menús.
b. Sección para el lenguaje natural.
c. Area para mensajes y teclas definidas.

Dichas secciones se componen de los siguientes elementos funcionales:

a. **Menús sensibles al contexto.** Dichos menús contendrán las acciones, las funciones, predicados y los nombres de los datos a accesar. Sensibilidad al contexto quiere decir que elegida una opción, los siguientes menús y pantallas

deben ser filtros para dicha opción, que además deben mostrar una coherencia semántica comprensible por el usuario.

b. **Uso de botones, funciones y teclas definidas para su operación.** El uso de botones es porque se incluye el uso de un mouse como dispositivo de ingreso opcional, y las funciones y teclas definidas facilitan la operación, aun en el caso de que no se posea el mouse, porque dan respuestas rápidas a las acciones requeridas.

c. **Lineas de mensajes.** Serán las últimas dos de la pantalla (lineas 24 y 25 en un monitor de 25 lineas), en la cual aparecerán mensajes de ayuda, para mantener informado al usuario en todo momento.

d. **Uso de ventanas** para agrupar y leer información.

E. Operación

El interfase en lenguaje natural actuará ante un usuario final que sea casual, de la siguiente forma:

a. Pedir al usuario qué información desea.

b. Solicitar que elija algún dato y repetir el proceso hasta que se desee ejecutar otra acción.

c. Las acciones pueden ser criterio de selección o agrupamiento y ordenamiento de información.

d. En el caso de criterio de selección, el usuario deberá indicar el dato que desea restringir, podrá seleccionar

los distintos comparadores, funciones, predicados y constantes que le ayudarán a definir la selección.

e. El agrupamiento y ordenamiento de datos permitirán al usuario elaborar el formato de sus respuestas.

f. El resultado le será mostrado en una ventana en la cual podrá navegar a través de los datos obtenidos.

g. Al terminar, dicha respuesta podrá ser procesada de nuevo, con lo cual el usuario tendrá capacidad de extraer más información si la necesita.

Cada acción de los siete pasos anteriores, tendrá una correspondencia en lenguaje natural, la cual será mostrada al usuario en una ventana fija, en la que él mismo revisará si los requerimientos especificados son los correctos.

Con estos requerimientos de diseño se procedió a la implementación, de la cual se habla en la siguiente sección.

VIII. IMPLEMENTACION

La implementación del interfase en lenguaje natural para accesar bases de datos relacionales, además de ser fiel a los requerimientos del diseño, introduce algunos criterios de último momento para lograr que el interfase fuese totalmente funcional.

La implementación de desarrolló en las siguientes etapas:

A. Definición del ambiente de trabajo

El ambiente de trabajo se divide en dos:

1. **Requerimientos de hardware.** El diseño requiere el empleo de un computador personal tipo IBM PC sencillo, que como mínimo cuente con estos requerimientos: memoria de 640K, un disco duro de aproximadamente 20 a 30 MB, un monitor monocromático y sistema operativo DOS. A pesar de que el sistema funciona en computadores sencillos, un computador de mayor nivel con procesador Intel 286, 386 o 486 inclusive mejoraría progresivamente el rendimiento del software. En pocas palabras el interfase es un interfase para computadores personales.

2. **Requerimientos de software.** Con los requerimientos de hardware anteriores en mente, el software para el

desarrollo debería ser menos demandante en sus requerimientos de funcionamiento, así como lo suficientemente poderoso para el desarrollo de todo el paquete. Asimismo, el sistema requeriría de facilidades para entrada y salida de información, manejo de ventanas y facilidad para definir menús de opciones, acceso a bases de datos relacionales y, si es posible, funcionamiento en redes locales (LANs).

Para el desarrollo del software era necesario un lenguaje de programación que cumpliera con las siguientes características:

a. Estructurado.

b. Eficiente.

c. Disponibilidad de herramientas de desarrollo (generadores de pantallas, reporteadores, etc.).

d. Acceso a bases de datos a través de SQL.

e. Facilidad para definir y procesar formatos de entrada y salida.

Los lenguajes como **C++** y **Pascal** ofrecen un bonito ambiente de desarrollo, son estructurados y eficientes, pero requieren de un esfuerzo extra en la programación, no poseen acceso directo a bases de datos, carecen de herramientas de desarrollo (generadores de pantallas y código, por ejemplo), y el tiempo de desarrollo resulta exagerado.

Informix 4GL, un lenguaje de cuarta generación que facilita la programación e incluye acceso a bases de datos

relacionales es ideal, pero lamentablemente las versiones disponibles que funcionan en sistemas personales bajo el sistema operativo DOS, no funcionan óptimamente y requieren más recursos de hardware.

Por último se consideró al **dBase IV** como un lenguaje ideal, ya que poseía un lenguaje de cuarta generación atractivo, un manejador de archivos, SQL, facilidades para definir captura de información y funcionamiento en redes locales. Desafortunadamente la versión disponible (1.0) no era lo suficientemente confiable y eficiente, y no fue posible disponer de una versión más actualizada.

Un lenguaje compatible con dBase IV y que posee ventajas para el desarrollo de sistemas que interactúen con el usuario, es **FoxPro/LAN 2.0**. Este lenguaje, además de la compatibilidad con dBase IV, posee un rico set de herramientas de desarrollo, es eficiente, trabaja en redes, tiene rápido acceso a archivos de bases de datos y maneja un subset de SQL, que para propósitos del proyecto, era suficiente.

Como detalle adicional se poseía el software original completo, y considerando lo anterior, se decidió que el desarrollo del software para el proyecto se haría en su totalidad en FoxPro/LAN 2.0.

B. Definición de la gramática en lenguaje natural

Se mencionó en el diseño que la gramática en lenguaje

natural debía ser compacta, sencilla y lo suficientemente poderosa para ser útil.

Tomando en cuenta que el modelo elegido no permite que el usuario ingrese en forma directa sus requerimientos en lenguaje natural, la gramática sería fija e inflexible, pero clara y completa, para que el flujo de los requerimientos den una idea precisa de lo que se está solicitando.

Para definir la gramática en lenguaje natural se presentaron los siguientes problemas:

1. *Las gramáticas en lenguaje natural no son compactas.* Esto quiere decir que al intentar contemplar todas las posibles variantes de una pregunta u oración en lenguaje natural, se generan una cantidad de oraciones, que son imposibles de manejar, así como tan amplias que resultan imprácticas e inútiles.
2. *Las gramáticas en lenguaje natural que son fijas, tienden a ser inflexibles.* Cuando se limita el dominio de una gramática a un juego de oraciones simples, en muchas ocasiones se limita la capacidad y libertad del usuario de poder transmitir sus requirimientos al computador, de manera natural.

Con estos problemas en mente, se desarrolló una gramática que fuera sencilla, compacta y que, a pesar de ser fija, lo suficientemente completa para permitir al usuario expresar sus requerimientos de información de manera natural y libre de

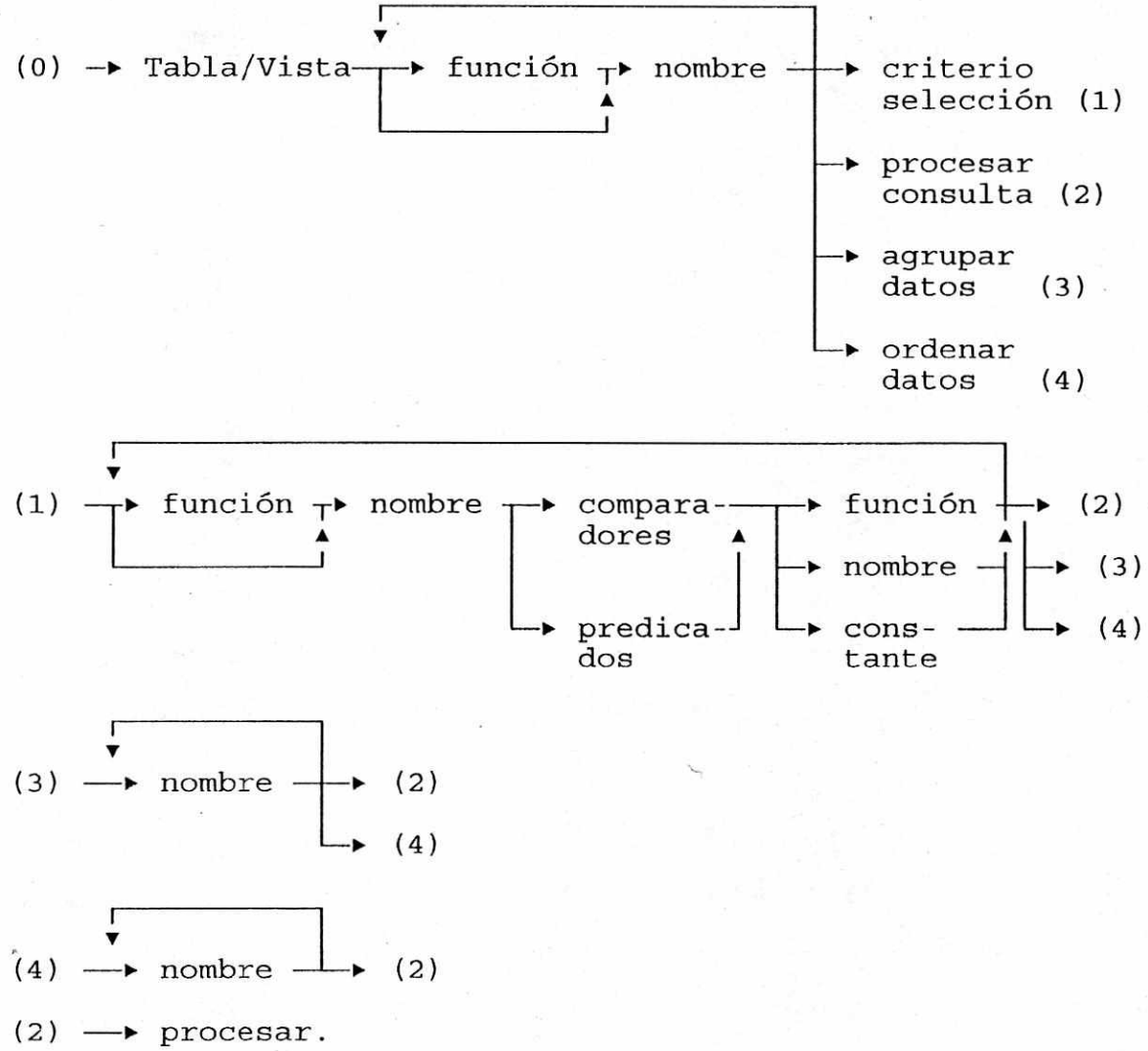
errores.

Nuestra gramática de manera general se compone de 5 secciones:

1. Diálogo para saber de qué archivo o vista se obtienen los datos.
2. Requerimiento de campos para dicho archivo, junto con ciertas operaciones como promedios, sumas y conteo de registros.
3. Criterio de selección (opcional).
4. Agrupamiento de información (opcional).
5. Ordenamiento de información (opcional).

Debido a que el lenguaje objeto a producir es SQL, se trató en lo posible de que cada una de estas secciones sutilmente semejara un parecido con la sintaxis de dicho lenguaje, aunque conservando la línea que posee un lenguaje natural. Un detalle adicional e importante es que se tomó como base al español como lenguaje natural para la definición de la gramática.

1. **Gramática Definida.** La gramática definida es la siguiente:



función ⟶ promedio, valores máximo y mínimo, suma y conteo de datos.

nombres ⟶ determinante (según el género) y un nombre de una columna de datos definida.

predicados ⟶ inclusión, rango y patrón de caracteres.

Dicha gramática tiene su correspondencia en lenguaje natural así:

Muéstreme <función|nombre,*> **de** <tabla|vista> [[**agrupada por** <nombre,*>] [**ordenada por** <nombre,*> <ascendente|descendente> |
que tengan <función|nombre> <comparador|predicado> <función| nombre|constante>.

[**Dicha información deberá estar agrupada por** <nombre,*>| **ordenada por** <nombre,*> <ascendente|descendente]].

C. Lenguaje objeto a producir

El objetivo del interfase es accesar bases de datos relacionales a través de SQL. El parser para la gramática implementada obtiene los requerimientos de los menús que despliegan la información, obtiene su correspondencia en lenguaje natural y al mismo tiempo genera la sintaxis de un Select de SQL, para posteriormente ejecutarla.

Debido a que SQL es un lenguaje extenso, se limitó la sintaxis del Select a lo esencial, pero que a su vez cumpliera con los estándares existentes. Entre estas limitaciones tenemos:

1. Se omitió el operador DISTINCT, ya que dicho operador hace que se pierda naturalidad en las consultas, debido a que es difícil de expresar/3/.
2. No se implementaron sub-queries, ya que, aunque estos son

/3/ Sin embargo este operador puede ser implementado por medio de un switch, el cual puede indicar si se realiza o no dicha operación.

más naturales, sacrifican eficiencia, pero pueden ser, en su defecto, implementados a través de un join común y corriente, que también suena y se escribe lógicamente, tanto en lenguaje natural como en SQL.

3. No se permiten operaciones aritméticas tanto en la sección de requerimientos, como en la del criterio de selección, debido a que incrementan la complejidad de la gramática.

Con el uso de las vistas, como veremos más tarde, algunas limitaciones pueden ser resueltas.

1. **Uso de vistas.** SQL tiene un mecanismo para proveer tanto integridad, restricción de dominios y seguridad en la información. Como lo indica Date (1987:173), este mecanismo es denominado *vista* es de uso común y nos facilita la consulta de información.

Las vistas tienen la característica de ser tablas virtuales, ya que son consultas, y no existen hasta que se les procesa.

El uso de la vista en el interfase es importante debido a tres razones básicas:

a. El uso de lenguaje natural para accesar datos, tiende a efectuar procesamiento basado en una sola tabla de datos, debido a que el lenguaje natural extiende la noción abstracta del usuario, que no sabe (y en muchas ocasiones ni le interesa) si dicha información se encuentra en un

solo archivo o está dispersa en otros.

b. Debido a que las vistas relacionan información de varios archivos, su uso nos permite restringir el dominio de información, lo que permite expresar las muchas y distintas formas lógicas que un usuario o grupo de usuarios tienen respecto de los datos.

c. La eliminación de las restricciones impuestas por la gramática definida para el interfase es otro punto en favor de la utilización de la vista, con lo que se puede lograr un mejor aprovechamiento del Select de SQL. Las restricciones eliminadas se refieren a la relación entre tablas por medio de joins, el uso de operadores aritméticos y el uso de ciertos operadores como DISTINCT.

D. Estructuras de datos implementadas

Estas estructuras son las tablas que almacenan la información del catálogo y el lexicón, que constituyen la base de conocimiento del interfase. A continuación una breve explicación de cada una de ellas:

1. Tablas del catálogo.

Las tablas del catálogo son tres:

a. **Tabla de Bases de Datos:** esta tabla contiene información de las distintas bases de datos que se dispone para trabajar. Una descripción se encuentra en la tabla 1.1.

b. **Tabla de Tablas y Vistas:** esta tabla contiene información de las tablas y vistas contenidas en las bases de datos. Una descripción se encuentra en la tabla 1.2.

c. **Tabla de Columnas:** esta tabla contiene la definición de columnas de las tablas de datos definidas en el catálogo. La descripción de la tabla se encuentra en la tabla 1.3.

Tabla 1.1

Tabla de bases de datos

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Nombre de la Base de Datos	Caracter	10	Guarda el nombre lógico de la base de datos
Path del Dos	Caracter	50	Nombre Directorio de archivos de la base
Fecha del Sistema	Fecha	8	Fecha de creación
Hora del Sistema	Caracter	8	Hora de creación
Usuario	Caracter	8	Usuario creador

Tabla 1.2

Tabla de Tablas y Vistas

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Base Datos	Caracter	10	Nombre de la Base de Datos
Nombre Tabla	Caracter	10	Nombre de la tabla de datos
Tipo Tabla	Caracter	1	Tabla común (**T**) o vista (**V**)
Select SQL	Memo	Indefinido	Código del Select SQL que construye una vista
Fecha del Sistema	Fecha	8	Fecha de Creación
Hora del Sistema	Caracter	8	Hora de Creación
Usuario	Caracter	8	Usuario creador

Tabla 1.3

Tabla de columnas de Tablas y Vistas

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Base de Datos	Caracter	10	Nombre de la base de datos
Tabla	Caracter	10	Tabla que posee la columna
Columna	Caracter	10	Nombre de la columna
Tipo	Caracter	1	Tipo de dato de la columna (**C**haracter, **M**emo, **N**umeric, **D**ate, **L**ogical, **F**loat)
Largo	Entero	3	Posiciones ocupadas
Decimales	Entero	2	Decimales de la columna
Fecha del Sistema	Fecha	8	Fecha de creación
Hora del Sistema	Caracter	8	Hora de creación
Usuario	Caracter	8	Usuario creador

2. **Tablas del Lexicón.** Las tablas del lexicón son cuatro:

a. **Tabla de sinónimos para Tablas en el lexicón:** contiene la definición de las tablas en el lexicón, o sea que contiene los nombres o los "alias" otorgados por el usuario, que se usarán en la generación de lenguaje natural. La descripción de ésta se encuentra en la tabla 2.1.

b. **Tabla de sinónimos de columnas:** esta tabla contiene los sinónimos en lenguaje natural para las columnas, y su definición se encuentra en la tabla 2.2.

c. **Tabla de unidades de medida:** esta tabla contiene el nombre y la abreviatura de las unidades de medida de las columnas, su definición está en la tabla 2.3.

d. **Tabla de dominios o tipos abstractos:** esta tabla contiene los posibles dominios que agrupan columnas de una tabla que posean el mismo tipo de dato (char, numeric, etc.), su definición está en la tabla 2.4.

Tabla 2.1

Tabla de sinónimos de Tablas en el Lexicón

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Base de Datos	Caracter	10	Nombre de la Base de datos
Tabla o Vista	Caracter	10	Tabla o Vista del catálogo
Comentario	Caracter	80	Comentario para la tabla o vista definida
Sinónimo	Caracter	30	Sinónimo en lenguaje natural de la tabla o vista
Género	Caracter	1	Género Masculino o Femenino de la tabla o vista.

Tabla 2.2

Tabla de sinónimos de columnas

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Base de Datos	Caracter	10	Nombre base de datos
Nombre de la Tabla o Vista	Caracter	10	Nombre de tabla o vista que contiene a la columna
Nombre de la Columna	Caracter	10	Nombre de la columna en el catálogo
Comentario	Caracter	80	Comentario de la columna
Sinónimo	Caracter	30	Sinónimo para la columna en lenguaje natural
Género	Caracter	1	Género de la columna
Unidad de Medida	Caracter	10	Unidad de medida para la columna (opcional)
Tipo abstracto	Caracter	10	Dominio al que pertenece la columna (opcional)
Datos default	Memo	Indefinido	Posibles valores que toma la columna (opcional)

Tabla 2.3

Tabla de unidades de medida

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Unidad de medida	Caracter	10	Nombre de la unidad de medida
Abreviatura	Caracter	5	Posible abreviatura (opcional)

Tabla 2.4

Tabla de dominios

Nombre de la Columna	Tipo de Dato	Número de Posiciones	Comentario
Nombre Dominio	Caracter	10	Nombre del dominio
Tipo dBase	Caracter	1	Tipo dBase que representa en dominio

E. Implementación del software necesario

En el diseño se identificaron dos programas básicos: **(1)** el programa de definiciones y **(2)** el interfase en lenguaje natural.

El programa de definiciones nos permite definir la correspondencia que tienen los elementos de la base de datos, en lenguaje natural, y a su vez nos permite restringir dominios de información.

El programa para el interfase es el programa que corre el usuario, es el programa que dados los requerimientos del mismo a través del uso de menús y lenguaje natural, produce un

resultado para una consulta solicitada.

A continuación se explican los distintos módulos involucrados en estos programas y su implementación.

1. **Programa de definiciones.** El programa de definiciones a implementar se compone de 5 módulos principales:

a. **Definiciones en el lexicón:** el módulo de definiciones en el lexicón sirve para relacionar los distintos elementos de las estructuras de la base de datos, y su correspondencia (sinónimos) en lenguaje natural.

Dicho módulo se compone de las siguientes partes:

i. *Definición de nombres para tablas y vistas.* Esta sección sirve para darle nombres en lenguaje natural a las tablas de datos, así como a las distintas vistas definidas. Dichos nombres se representan en el menú de tablas y vistas, que constituye el menú principal del interfase.

Este proceso involucra ingreso, consulta, eliminación y modificación, y obtiene información sobre el nombre de la tabla o vista en el catálogo, el nombre que le corresponde en lenguaje natural, un comentario de información y si dicha tabla o vista tiene género masculino o femenino.

ii. *Definición de nombres para columnas.* Similar a la

definición de nombres para tablas y vistas, sirve para darle nombre en lenguaje natural a las columnas de datos contenidas en tablas y vistas definidas. En este proceso obtiene información sobre la tabla que contiene la columna, el nombre de columna (ambos del catálogo), el nombre en lenguaje natural, un comentario de información, género de la columna, su unidad de medida (opcional), el nombre del dominio al que pertenece (opcional) y los distintos valores constantes que pudiera tener (opcional).

iii. *Definición de unidades de medida.* Permite la definición de unidades de medida para las columnas. Las unidades de medida nos permiten darle más claridad a las oraciones en lenguaje natural, ya que proporcionan más información sobre el nombre o columna de datos que se esté trabajando. Así por ejemplo si estamos trabajando con un nombre denominado edad, la unidad de medida año nos indicará que la edad está medida en años.

iv. *Definición de dominios o tipos abstractos.* Este proceso permite la definición de grupos o dominios de datos, los cuales se utilizan para restringir columnas de información del mismo tipo. Por ejemplo si tenemos dos columnas, ciudad de

nacimiento y ciudad de residencia, ambas de tipo caracter, y deseamos restringirlas a un dominio que agrupe únicamente ciudades, podemos definir el dominio *ciudades*, y en la definición de columnas del lexicón situar ambas en el mismo, con lo que cuando necesitemos tratar dominios de ciudades, podremos trabajar ambas columnas separadas del resto de columnas de tipo caracter.

b. **Definiciones en el catálogo:** el módulo de definiciones en el catálogo no es más que la implementación de un catálogo básico de SQL, que se construyó debido a que la implementación de SQL de FoxPro/LAN 2.0 no incluye un catálogo o diccionario de datos, el cual es útil para las definiciones en el lexicón.

Este proceso define bases de datos, tablas y sus columnas pertenecientes a una base de datos determinada, y la definición de vistas. Todos los procesos poseen módulos de ingreso, modificación, eliminación y consulta.

Una mención especial lo constituye el proceso de definiciones de vistas, que permiten la definición de grupos de información especificados en cláusulas select de SQL, el cual facilita el proceso y seguridad de la información.

c. **Interfase para consultas en lenguaje natural:** el interfase para consultas en lenguaje natural es el segundo

programa (el interfase en sí) que se incluye en el programa de definiciones para ayudar a dar una idea clara de lo que se espera que vea el usuario, con lo cual el proceso de diseño y adaptación de la información de una base de datos determinada puede ser monitoreado, hasta que satisfaga todos los requerimientos.

Más adelante se especificarán cada una de las partes del interfase.

d. **Opciones para definiciones y operaciones del interfase:** el módulo de opciones permite abrir bases de datos, darle mantenimiento a los archivos del sistema y, seleccionar los modos para uso en red y observar código generado en SQL, respectivamente.

e. **Módulo de Ayuda:** el módulo de ayuda da información de todos los procesos involucrados en el sistema. Este módulo incluye un índice para buscar información determinada y provee una explicación del tópico seleccionado.

No está más decir que el programa de definiciones está diseñado para usuarios específicos, que estén bien capacitados y conozcan la estructura de las bases de datos a implementar. Sin embargo el programa es sencillo y facilita la definición (que no deja de ser tediosa en cualquier sistema de lenguaje natural) de información en lenguaje natural.

2. **Programa del interfase en lenguaje natural.** Este programa es el interfase en lenguaje natural con la cual los usuarios interactúan para obtener información de las base de datos. Para su análisis primero haremos una breve descripción de las partes que la integran, describiremos el algoritmo para procesarlas y los distintos detalles y características especiales que distinguen al interfase. Este programa es basado en los módulos mostrados en la fig. 5.2 y definidos en el capítulo V.

a. **Partes que integran el interfase en lenguaje natural:** el programa para manejar el interfase en lenguaje natural consiste de cuatro secciones importantes:

i. **Definiciones:** el proceso de definiciones inicializa el sistema desde las variables hasta los distintos menús involucrados que leen los distintos requerimientos y que se controlan en la gramática involucrada.

Este proceso se compone de las siguientes partes:

1. Un setup, en donde se dan los controles iniciales del programa.
2. Definición de variables globales.
3. Definición de ventanas.
4. Definición de menús.

La explicación de cada una de estas partes es obvia.

ii. **El Parser:** es el proceso que maneja la gramática definida, la cual tiene incorporada. Utiliza ATNs simples

y recursión descendente como técnicas de procesamiento/4/.

El parser se encarga de obtener los requerimientos seleccionados en los menús, generar y mostrar su correspondencia en lenguaje natural, construir el select SQL y chequear errores.

El parser posee las siguientes partes:

1. Un proceso de control, el cual se encarga de ver qué parte de la gramática continúa y qué acción se toma según una determinada variable de control. Cada una de estas partes a ejecutarse constituyen distintos procesos, que son los nodos del ATN. La recursión descendente actúa cuando uno de estos procesos llama a otro y que también llama a éste, o cuando un proceso se llama a sí mismo. Este proceso sigue el algoritmo:

```
procedure control
  parameter ncontrol
  do case
    case ncontrol = 1
       do <proceso 1>
            ...
  endcase
return
```

2. Procesos, que son pantallas que obtienen los requerimientos del usuario y, según estos requerimientos llaman a procesos definidos o

---

/4/ Estas técnicas fueron explicadas en el capítulo III.

validaciones que: **(a)** despliegan lenguaje natural, **(b)** procesan datos u opciones ingresadas en las pantallas y, **(c)** determinan qué control se activa, con lo que se determina qué acción continua. Asimismo, con estos controles, un mismo proceso puede comportarse de distinta forma, como por ejemplo si se está trabajando con calificadores y valores de constantes, según el calificador puede o no mostrarse menús de otros calificadores o menús de constantes posibles.

3. Un generador de código en SQL, que se encuentra incorporado en cada uno de los procesos, y se encarga de producir, según el estado del ATN en que se encuentre, las distintas partes del Select SQL.

iii. **Proceso del Select SQL generado:** el Select SQL generado es el resultado de la interacción del usuario con los menús y la correspondencia de la información en lenguaje natural. El proceso de este Select es hecho por esta sección que indica los distintos errores ocurridos (acceso indebido a las tablas, información no encontrada para el requerimiento) si los hubiere, y deposita el resultado en una tabla temporal, la cual es denominada *tabla resultado*.

iv. **Resultado y manejo de respuestas:** la *tabla resultado* es desplegada en una ventana de datos, en la cual el usuario

puede navegar para observar los datos obtenidos. Cuando el usuario finalice, dicha respuesta es colocada como una tabla más del menú principal, permitiéndosele al mismo granular aún más la información.

b. **Algoritmo para procesamiento del interfase:** la interfase procesa el siguiente algoritmo:

1. Obtener los requerimientos de los usuarios a partir de los distintos menús desplegados por el parser según la gramática incorporada.
2. Generar código en lenguaje natural, el cual es mostrado al usuario, al mismo tiempo que el código en SQL.
3. Procesar el código en SQL y generar el resultado.
4. Mostrar el resultado y colocar la respuesta en el menú principal.
5. Se regresa al paso **a.** hasta que el usuario termine la sesión con el interfase.

c. **Características especiales del interfase:** las características especiales del interfase pueden ser enumeradas de la siguiente forma:

i. **Pantallas:** las pantallas que integran el interfase se componen de 4 secciones importantes:

1. La ventana de menús de información, de acciones y predicados.
2. La ventana de botones para procesamiento, que son

tres: (1) botón de regreso, que regresa a la pantalla anterior; (2) el botón de ayuda, para obtener ayuda en la pantalla que se esté procesando; y (3) el botón de cancelación de operación. Estos botones se encuentran representados por teclas correspondientes en el teclado.

3. Ventana de código en lenguaje natural, que muestra el estado de los requerimientos ingresados en los menús, en lenguaje natural.
4. Lineas de mensajes, que son las últimas dos de la pantalla, y nos sirven para escribir mensajes de información para el usuario.

ii. **Manejo de la gramática:** la gramática en lenguaje natural posee detalles como los siguientes:

1. Las fechas son indicadas mostrando en forma escrita el día, el nombre del mes en español y el año.
2. Los nombres según el género se les antepone el determinante el (en caso de género masculino) y el determinante la (en caso de género femenino).
3. Las unidades de medida de cada nombre ayudan a clarificar el contexto y dominio de información que se está trabajando.
4. El manejo de plurales, que se detectan para los nombres de las vistas y las unidades de medida.

En los apéndices A y B se ejemplifica el proceso de cada

uno de estos programas, así como algunas de las características mencionadas.

VIII. CONCLUSIONES

1. Un sistema de procesamiento de lenguaje natural debe ser capaz de: **(a)** aceptar ingresos en lenguaje natural, **(b)** almacenar conocimiento relacionado a algún dominio de aplicación, **(c)** trazar inferencias y responder a preguntas sobre dicho conocimiento, y **(d)** generar respuestas apropiadas.
2. Un sistema de procesamiento de lenguaje natural se compone de cinco partes esenciales: **(1)** el parser, **(2)** el generador, **(3)** el traductor, **(4)** la base de conocimiento y **(5)** el lexicón.
3. Un sistema de consultas para accesar bases de datos utilizando lenguaje natural (NLQS), puede ser más fácil de manejar, ya que las consultas son efectuadas en la forma normal en que se hacen preguntas: lenguaje natural.
4. Los NLQS se clasifican en tres grupos: **(1)** sistemas basados en pattern-matching, **(2)** sistemas sintácticos y **(3)** sistemas basados en conocimiento.
5. Los modelos de procesamiento para NLQS más utilizados son: modelo basado en una gramática semántica y el modelo basado en menús sensibles al contexto.
6. El modelo de NLQS basado en una gramática semántica mantiene una excelente secuencia entre preguntas y

respuestas, siendo un sistema más cooperativo y operacional.

7. El NLQS basado en una gramática semántica, necesita un juego de reglas semánticas muy completo, lo que provoca que su instalación y mantenimientos sean muy costosos.
8. El modelo de NLQS basado en menús sensibles al contexto es más sencillo, ya que muestra todas las posibles variantes de los datos; además su instalación y mantenimiento son menos costosos.
9. El NLQS basado en menús, limita al usuario a utilizar únicamente lo que el diseñador prepare, ya que elimina la posibilidad de una amplia formulación de requerimientos.
10. Un NLQS si se instala, integra y usa apropiadamente, puede resolver los problemas de recuperación de información en bases de datos relacionales.

IX. BIBLIOGRAFIA

Advanced Topics dBase IV. Torrance, California,
1988 Ashton-Tate. 239 pp.

Aho, Alfred V.; Ravi Sethi y Jeffrey D. Ullman.
1987 Compilers. Principles, Techniques, and Tools.
Addison-Wesley Publishing Company, Inc. 796 pp.

Allen, James. Natural Language Understanding. Menlo
1988 Park, California, Benjamin/Cummings Publishing
Company, Inc. 579 pp.

Date, C.J. An Introduction to Database Systems. 4ta.
1987 ed. Addison-Wesley Publishing Company, Inc. 2
v.

FoxPro Commands & Functions. Perrysburg, Ohio, Fox
1991 Software, Inc. 1,077 pp.

FoxPro Developer's Guide. Perrysburg, Ohio, Fox
1991 Software, Inc. 662 pp.

FoxPro Getting Started. Perrysburg, Ohio, Fox
1991 Software, Inc. 310 pp.

FoxPro Installation Guide. Perrysbug, Ohio, Fox
1991 Software, Inc. 20 pp.

FoxPro Interface Guide. Perrysburg, Ohio, Fox
1991 Software, Inc. 514 pp.

FoxPro Master Index. Perrysburg, Ohio, Fox Software,
1991 Inc. 44 pp.

FoxPro Quick Reference. Perrysburg, Ohio, Fox
1991 Software, Inc. 76 pp.

Gotlieb, C.C.; Leo R. Gotlieb. Data Types and
1978 Structures. Englewood Cliffs, N.J., Prentice-
Hall, Inc. 440 pp.

Harris, Mary Dee. Introduction to Natural Language
1985 Processing. Reston, Virginia, Reston Publishing
Company, Inc. 368 pp.

Kerschberg, Larry. Expert Database Systems,
1986 Proceedings from the First International Workshop. Menlo Park, California, Benjamin/Cummings Publishing Company, Inc. 698 pp.

Laurel, Brenda. The Art of Human Computer Interface
1990 Design. Addison-Wesley Publishing Company, Inc. 523 pp.

Luger, George F.; William A. Stubblefield. Artificial
1989 Intelligence and the Design of Expert Systems. Menlo Park, California, Benjamin/Cummings Publishing Company, Inc. 659 pp.

Obermeier, Klaus K. "Natural-Language Processing".
1987 Byte, The Small Systems Journal; 12 (14): 225-232.

_____; "Natural Selection". Byte; 15 (8): 217-222.
1990

Shneiderman, Ben. Designing the User Interface.
1987 Addison-Wesley Publishing Company, Inc. 448 pp.

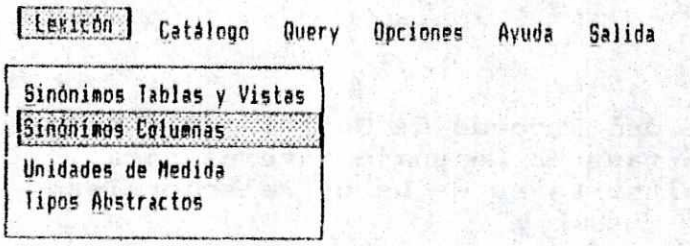
Tenembaum, Aaron M.; Moshe J. Augenstein. Data
1981 Structures Using Pascal. Englewood Cliffs, N.J., Prentice-Hall, Inc. 545 pp.

Wirth, Niklaus. Algoritms+Data Structures=Programs.
1986 Englewood Cliffs, N.J., Prentice-Hall, Inc. 366 pp.

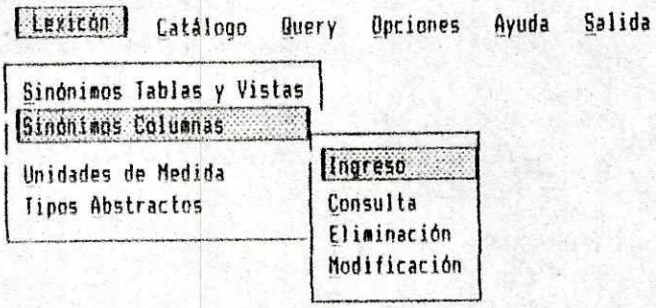
APENDICE A

Ejemplo del Proceso de Definiciones del Interfase en Lenguaje Natural para Consultar Bases de Datos Relacionales

En las páginas de este apéndice se muestra un ejemplo del proceso de definiciones, el cual se utiliza para definir los distintos elementos de una base de datos y su correspondencia en lenguaje natural.

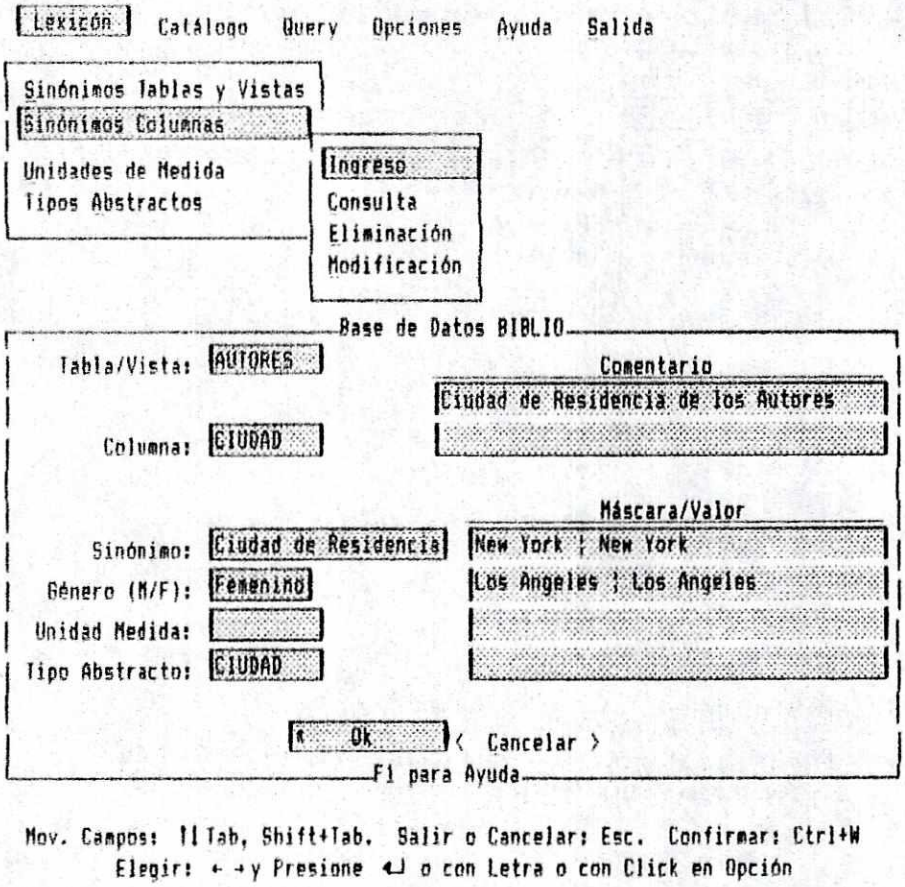


Pantalla 1: Para este ejemplo seleccionaremos el ingreso de sinónimos para columnas de tablas y vistas, con lo cual ejemplificaremos el proceso de definiciones para el interfase en lenguaje natural para consultar bases de datos relacionales.



Pantalla 2: Elegida la opción de sinónimos de columnas, aparece el menú de opciones que incluye ingreso, consulta, eliminación y modificación de sinónimos, respectivamente.

Para este ejemplo elegimos la opción de ingreso.



Pantalla 3: Este es el proceso de ingreso de sinónimos para columnas, en el cual se ingresan los siguientes datos:

(a) Se ingresa la tabla o vista a la cual pertenece la columna, así como el nombre que la columna posee en el catálogo para dicha tabla.

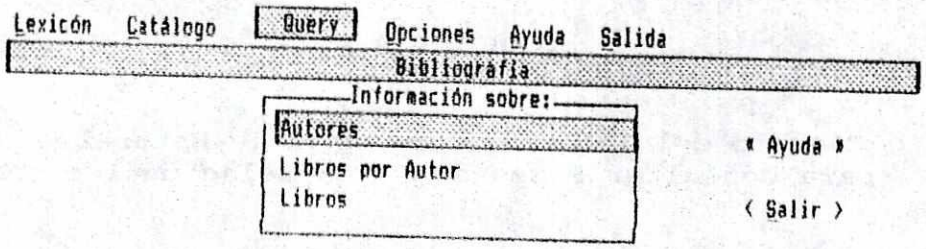
(b) Se ingresa el comentario que aparece en la línea de mensajes del interfase, el nombre o sinónimo que toma en lenguaje natural, el género masculino o femenino de la columna, su unidad de medida (opcional), el dominio de datos al que pertenece (opcional), y los distintos valores que puede tener la columna (opcional).

La pantalla mostrada explica por sí misma el proceso.

APENDICE B

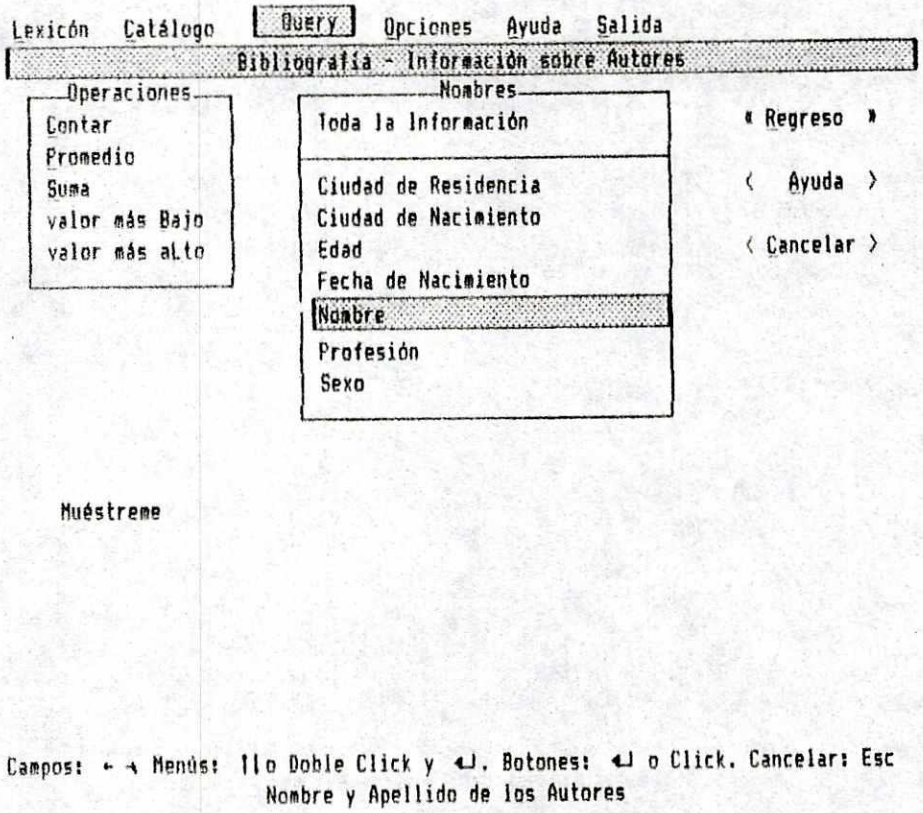
Ejemplo del Interfase en Lenguaje Natural para Consultar Bases de Datos Relacionales

En las páginas de este apéndice, se encuentra una muestra de corrida del proceso del interfase. El ejemplo completo nos da una idea de las capacidades y características que posee el interfase en lenguaje natural para consultar base de datos relacionales.



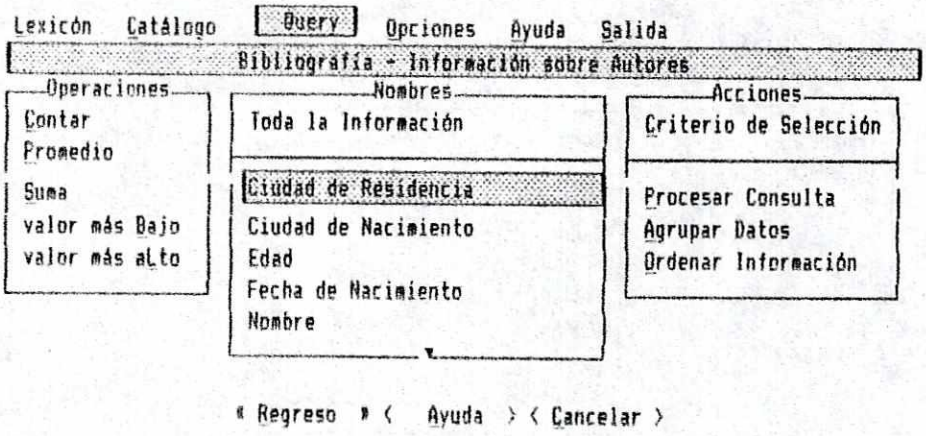
Pantalla 1: Esta es la pantalla principal del interfase, en ella se le presentan al usuario las distintas tablas y vistas de las cuales puede extraer información. Podemos apreciar los mensajes de información y los botones, que en este caso, sirven para obtener ayuda y para abandonar el interfase respectivamente.

Para este ejemplo seleccionamos, como se indica, información sobre Autores y continuamos.



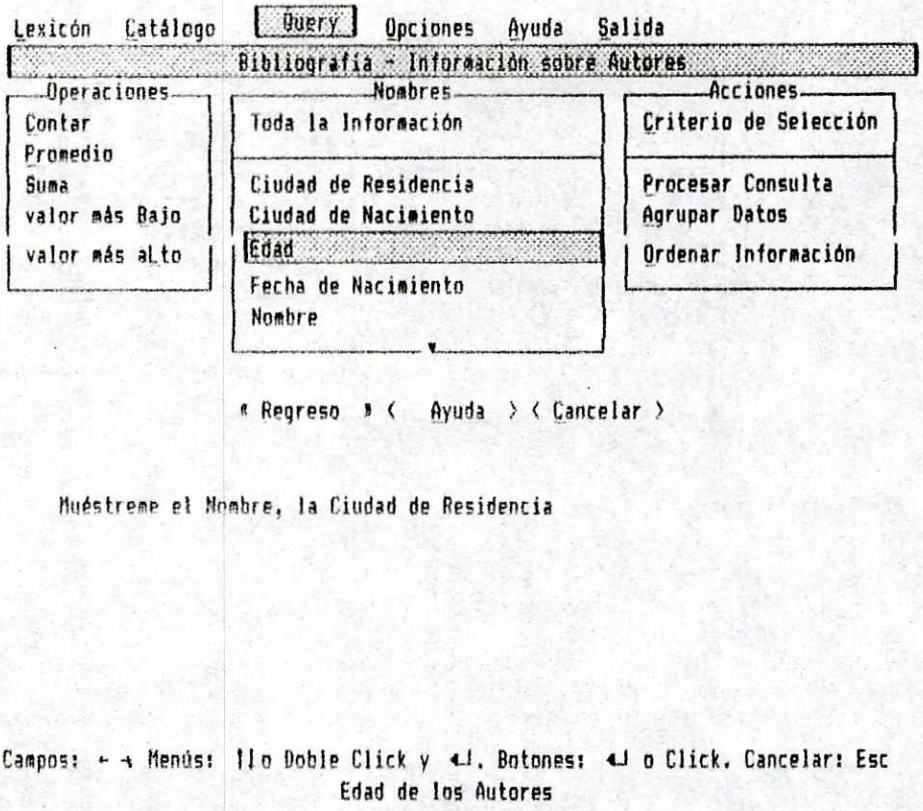
Pantalla 2: Luego de seleccionar información sobre Autores, la siguiente pantalla que aparece es esta, y en ella se indica con qué tabla o vista se está trabajando y se piden los datos que se desean obtener de la misma. También aparecen los distintas funciones que se le pueden aplicar a los datos, los nombres que existen en la tabla o vista seleccionada; se muestra el primer código en lenguaje natural (Muéstreme) que precede al dato a obtener. Los botones indican regreso a la pantalla anterior, ayuda y abortar la sesión con el interfase, respectivamente.

Continuando con el ejemplo, seleccionamos Nombre como columna de la cual deseamos obtener información.

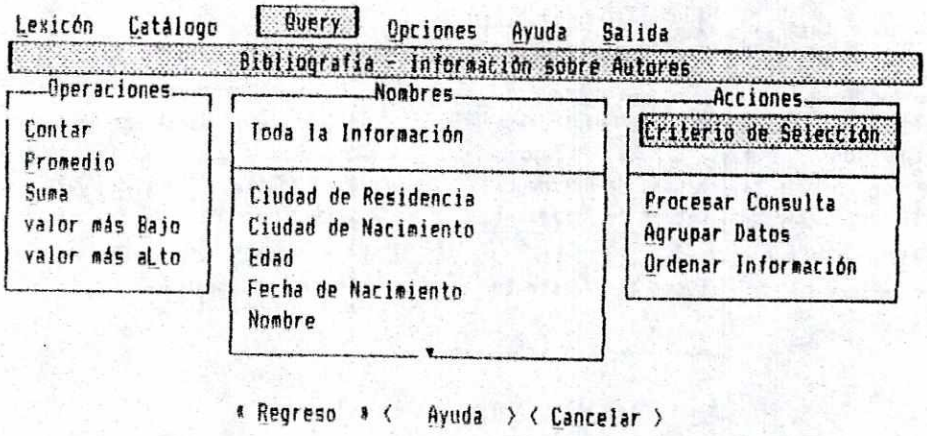


Pantalla 3: Luego de seleccionar nombre, aparece esta pantalla la cual difiere de la pantalla 2 en que se presenta un menú de acciones, en el cual podemos solicitar criterio de selección o procesar, agrupar y ordenar información. El cláusula Toda la Información, nos permite obtener todos los datos que se encuentran en la tabla o vista seleccionada.

Para continuar con nuestro ejemplo pediremos información sobre la ciudad de residencia.



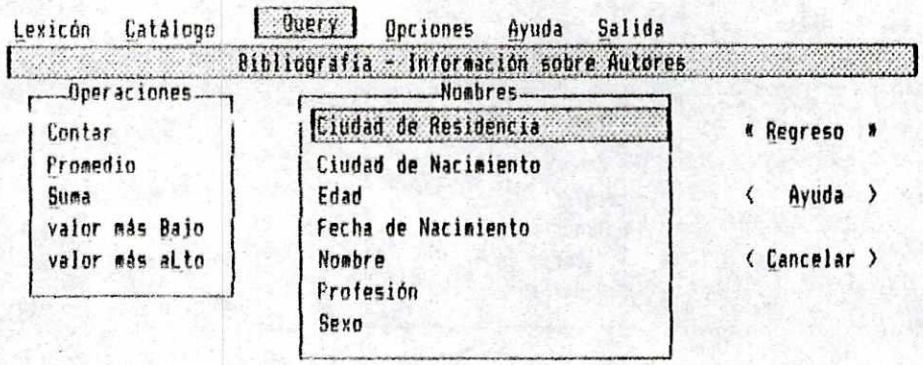
Pantalla 4: En esta pantalla continuamos solicitando información sobre los datos de autores. Seleccionamos la edad como siguiente dato.



Muéstreme el Nombre, la Ciudad de Residencia, la Edad

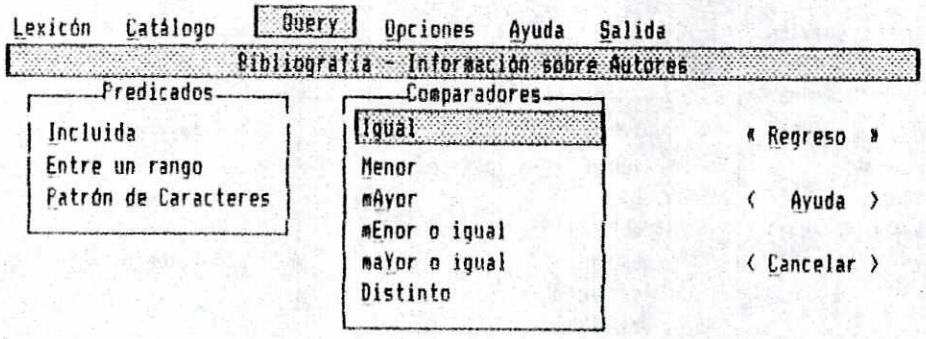
Campos: ← → Menús: ↑↓ o Doble Click y ↵, Botones: ↵ o Click, Cancelar: Esc

Pantalla 5: Completado nuestro requerimiento de información, procedemos a indicar nuestro criterio de selección. Nótese el código en lenguaje natural que se ha generado hasta el momento.



Pantalla 6: Esta pantalla de criterio de selección nos sirve para restringir nuestra información. Es similar a la pantalla 2, pero con la excepción que los datos y operaciones seleccionados filtran los datos solicitados con anterioridad.

Para el ejemplo restringiremos información sobre la ciudad de residencia.

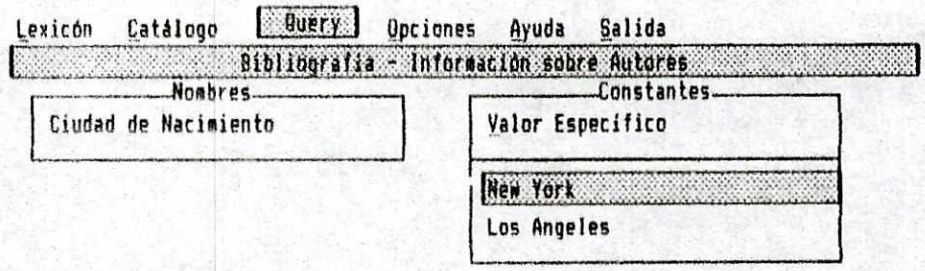


Muéstreme la Profesión, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia

Campos: ← →, Cancelar: Esc
Elija con ↑↓ o Letra y Presion ↵ o con Doble Click en Opción

Pantalla 7: Luego de seleccionar ciudad de residencia como columna sobre la cual restringiremos los datos, nos aparece la pantalla de comparadores y predicados, en la cual podemos indicar que deseamos comparar nuestra columna con otro dato, o que deseamos aplicarle alguna operación de inclusión, restricción o búsqueda en algún patrón de caracteres.

Para nuestro ejemplo seleccionamos igual como el comparador a usar. Nótese el código en lenguaje natural generado hasta el momento.

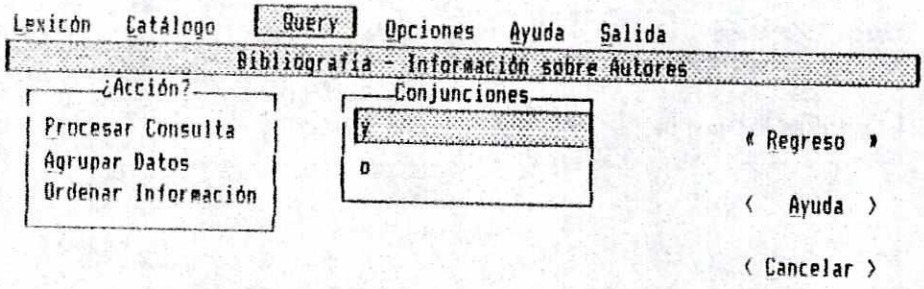


Pantalla 8: Al seleccionar igual como el comparador para la columna ciudad de residencia, nos aparece esta pantalla, en la cual hay dos menús:

(1) El menú de nombres, en el cual se muestran las posibles columnas que se pueden comparar con la ciudad de residencia; aquí se muestra únicamente la ciudad de nacimiento, debido a que pertenece al mismo dominio que la ciudad de residencia.

(2) Menú de constantes, en el que aparecen los posibles valores predefinidos que puede tener la columna, o se puede especificar otro valor si este no aparece en la lista.

Para continuar elegimos la constante New York, como el valor de nuestra restricción.

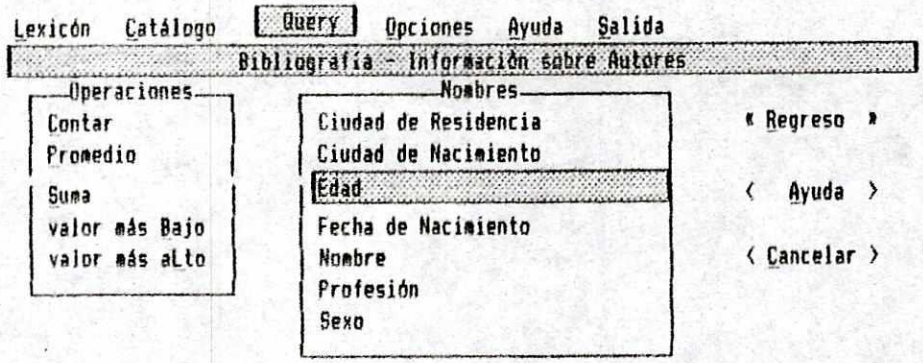


Muéstreme la Profesión, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York

Campos: ← → Cancelar: Esc
Elija con ↑↓ y Presione ↵ o con Doble Click en Opción

Pantalla 9: El código en lenguaje natural nos da una idea clara de nuestro requerimiento hasta el momento, y en esta pantalla podemos proceder con nuestra consulta o continuar con otro tipo de operación. Las conjunciones *y* y *o*, nos sirven para enlazar otros criterios de selección y son la base para construir más oraciones en lenguaje natural.

Para completar el ejemplo aún nos hace falta otro criterio de selección, por lo que utilizaremos la conjunción *y* para enlazarlo.

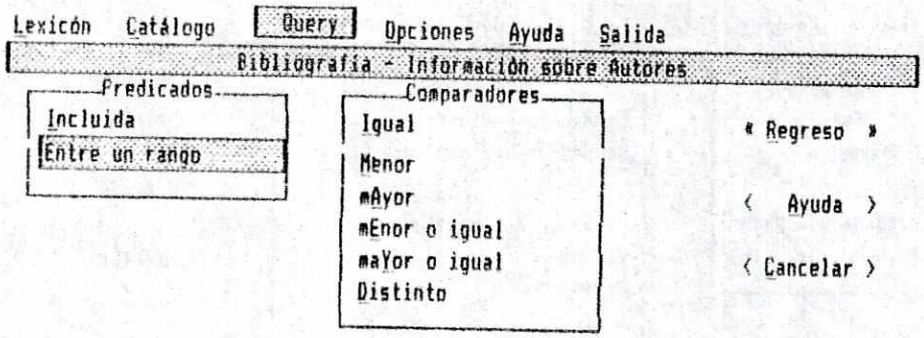


Muéstreme la Profesión, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York, y

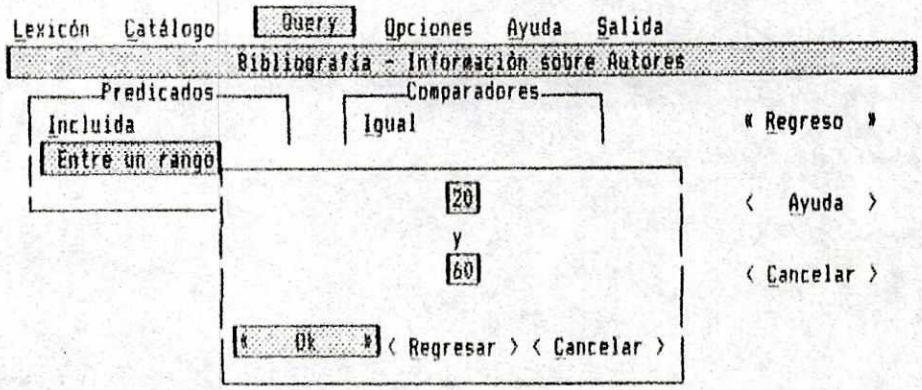
Campos: ← → Menús: ↑↓ o Doble Click y ↵. Botones: ↵ o Click. Cancelar: Esc
Edad de los Autores

Pantalla 10: Al seleccionar una conjunción, la pantalla que aparece de nuevo es la pantalla de criterio de selección, ya que eso indica que se pide una nueva columna sobre la cual se aplica otro criterio.

El siguiente requerimiento para nuestro criterio de selección será el restringir los datos por edad, por lo que seleccionamos la columna edad.



Pantalla 11: Regresamos a la pantalla de comparadores y predicados. Para este ejemplo vamos a utilizar un predicado como operación a ser aplicada a la columna seleccionada. Elegimos el predicado entre un rango, el cual sirve para restringir datos entre dos valores, los cuales son inclusive, es decir, que se toman en cuenta en la comparación.

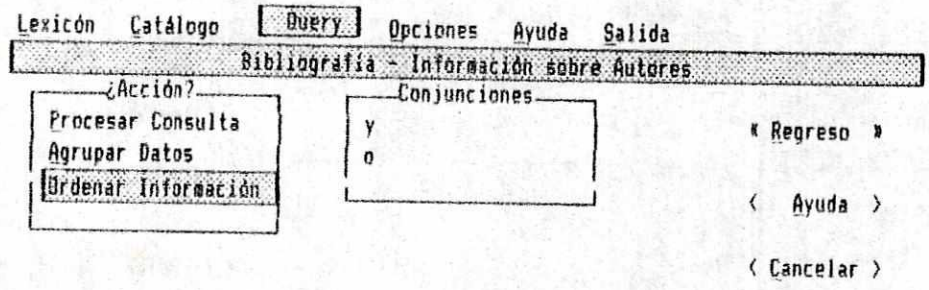


Muéstreme el Nombre, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York, y la Edad entre

Campos: ↑↓ Cancelar: Esc
Elija con ← →y Presione ↵ o con Letra o con Click en Opción

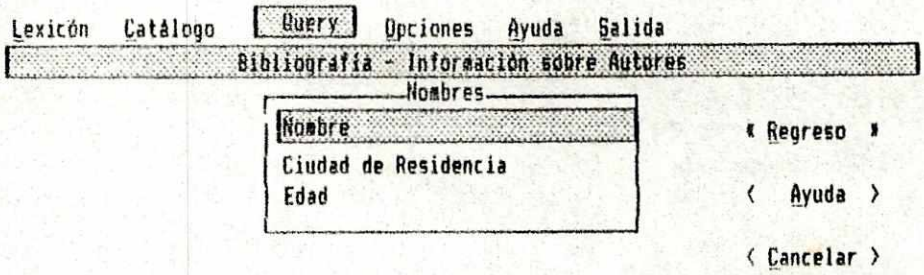
Pantalla 12: Al elegir el predicado de rangos, se nos presenta esta pantalla, en la cual se deben indicar los valores extremos del rango solicitado. El botón ok sirve para confirmar el ingreso.

Para nuestro ejemplo restringiremos la edad entre 20 y 60.



Pantalla 13: Al indicar el rango, aparece de nuevo la pantalla de conjunciones y de acciones. Nótese que el código en lenguaje natural indica que la edad está restringida en años, lo cual ejemplifica el uso de las unidades de medida, ya que la columna edad se encuentra definida con esta unidad.

Para continuar, ordenaremos nuestra salida, por lo que elegimos la opción de ordenar información.



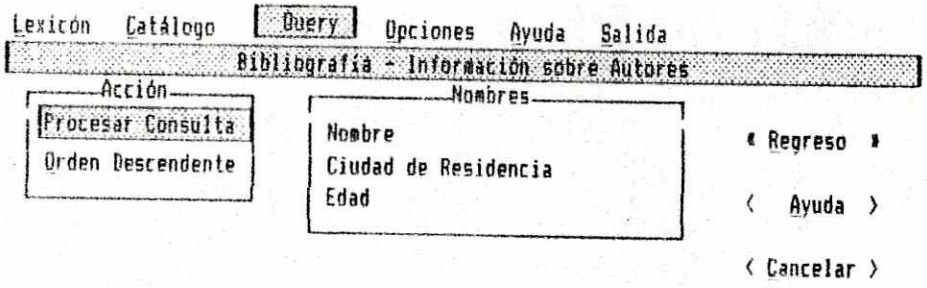
Muéstreme el Nombre, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York y la Edad entre 20 y 60 años.

Dicha información deberá estar ordenada por

Campos: ← → Menús: ↑↓ o Doble Click y ↵. Botones: ↵ o Click. Cancelar: Esc
Nombre y Apellido de los Autores

Pantalla 14: Al seleccionar la opción de ordenamiento, nos aparece esta pantalla, en la cual se presentan las posibles columnas por las que se puede ordenar la información. Nótese que aparecen únicamente los datos de los que se desea obtener información.

Para nuestro ejemplo, ordenaremos la información por nombre del autor.



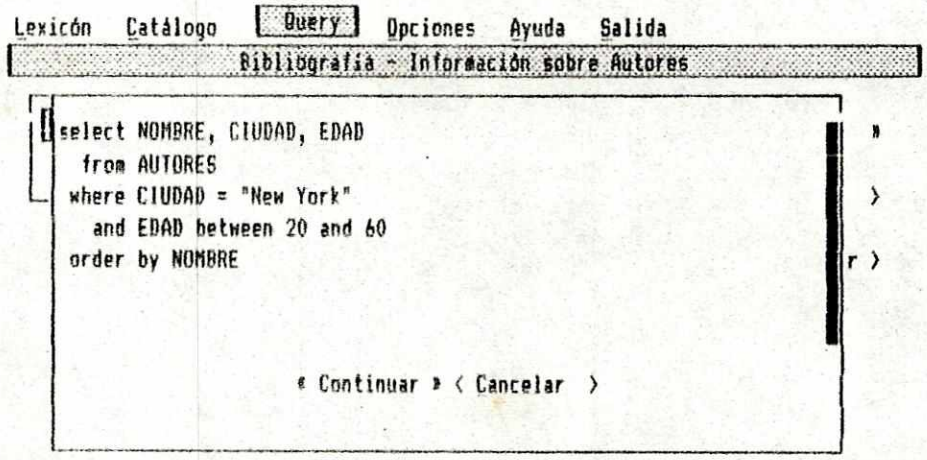
Muéstrene el Nombre, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York y la Edad entre 20 y 60 años.

Dicha información deberá estar ordenada por el Nombre

Campos: ← → Menús: ↑↓ o Doble Click y ↵. Botones: ↵ o Click. Cancelar: Esc

Pantalla 15: Completado nuestro requerimiento de consulta, podemos observar como se encuentra construído el mismo en lenguaje natural, lo cual nos da una idea exacta de lo que queremos obtener.

Seleccionamos procesar consulta para obtener la respuesta a la consulta solicitada.

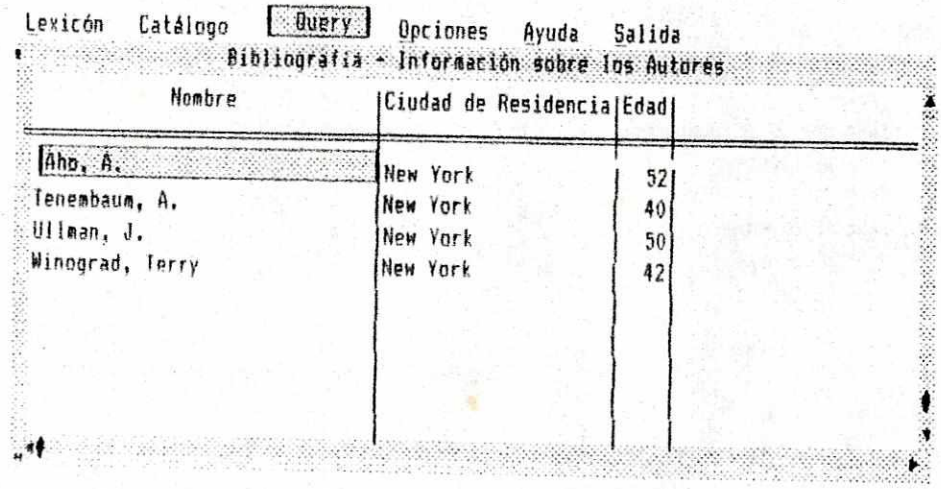


Muéstreme el Nombre, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York y la Edad entre 20 y 60 años.

Dicha información deberá estar ordenada por el Nombre.

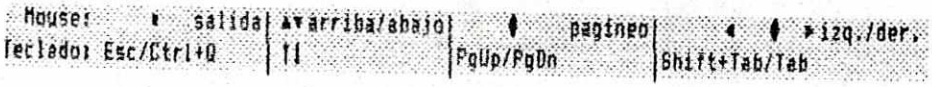
Mov. Campos: Tab, Shift+Tab. Salir o Cancelar: Esc
Consulte con ↑↓ o con Click en ▲ ◆ ▼

Pantalla 16: Para efectos de nuestro ejemplo, se muestra en esta pantalla el código generado en SQL, que es la contraparte de nuestro requerimiento de consulta en lenguaje natural.



Muéstreme el Nombre, la Ciudad de Residencia y la Edad de los Autores que tengan la Ciudad de Residencia igual a New York y la Edad entre 20 y 60 años.

Dicha información deberá estar ordenada por el Nombre.



Pantalla 17: En esta pantalla se muestra el resultado de la consulta solicitada usando el interfase en lenguaje natural para accesar bases de datos relacionales.

Dicha respuesta pasa a ser una de las vista disponibles en el menú principal, por lo que puede ser utilizada para obtener más información, con lo que se constituye en un subconjunto de datos del primer resultado.