UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGON

INTERFAZ EN LENGUAJE NATURAL PARA CONSULTA DE BASE DE DATOS.

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

PRESENTA:

CARLOS MARIO MARTINEZ MASCARUA

MEXICO, D.F. SEPTIEMBRE DE 1986.

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

I N D I C E

R E S U M E N

La interacción en lenguaje natural entre el ser humano y la computadora es un área de investigación del campo de la Inteligencia Artificial. Este trabajo enfrenta el problema de la comprensión automática del lenguaje natural con el fin de lograr que el usuario consulte la información contenida en una base de datos mediante preguntas en español.

En esta tesis se muestra la manera en que se realiza el análisis sintáctico en las interfaces para consulta de bases de datos en lenguaje natural, y se desarrolla la etapa de análisis denominada **analisis semántico**, de tal forma que permita que el sistema completo de comprensión de lenguaje sea lo más independiente posible de la base de datos con que opere. La teoria desarrollada para tales fines es expuesta, y posteriormente se muestra el desempeño de un prototipo implantado para su validación.

1 INTRODUCCION.

El impacto de la computación en nuestra sociedad es comparable con el que en ella tuvo la revolución industrial. Dia a dia, el procesamiento electrónico de datos influye en más y más actividades de nuestra vida. Nuestros datos personales forman registros en sistemas de información que imprimen nuestro recibo telefónico; nuestro salario es depositado automáticamente en nuestra cuenta bancaria; nuestra póliza de seguros se mantiene al dia automáticamente, etc.

Todos los sectores de la sociedad dependen cada vez más de la computadora. Compañias aéreas, bancos y hospitales, además de todo el sector público, son usuarios de grandes sistemas computarizados, por no mencionar los sectores militares de los paises industrializados, que son el motor principal del vertiginoso desarrollo de la electrónica y las ciencias computacionales.

La computadora incursiona ahora en los sectores productivos, en donde sirve para controlar producción, distribución e inventarios de bienes; también se utiliza la computadora para racionalizar el diseño y la ingeniería de nuevos productos.

La lista de las actividades en las que se emplea la computadora como herramienta es muy larga para exponerla aqui. Sabemos que la tendencia en su utilización se incrementa, y no está lejano el dia en que todos dispondremos, en nuestra propia casa, de una computadora conectada a una red internacional con la que podremos consultar, cuando lo deseemos, desde el pronóstico del tiempo hasta el estado del mercado de cambios en Nueva York; desde la programación televisiva local hasta el horario de un concierto en Viena, que podremos ver en vivo por televisión o retransmitido a la hora que lo deseemos. Todo este desarrollo será posible con la ayuda de sistemas diseñados para manejar enormes cantidades de información.

En la actualidad, la administración de información es un problema que se puede resolver mediante técnicas de base de datos. Sin embargo, el usuario final que desee consultar esta información tiene que adaptarse a serias limitaciones: debe conocer cuál es la estructura de sus archivos y debe dominar la sintaxis de un lenguaje formal de consulta.

> "Uno de los mayores obstáculos para la aceptación universal de las bases de datos es la resistencia del usuario promedio al relativamente molesto método de acceso, o por lo menos, a la rigidez de la interfaz entre el usuario y la información almacenada... Un slogan sobre el uso comercial de la Inteligencia Artificial dice que debemos hacer que la máquina sepa más sobre el usuario para que el usuario necesite saber menos sobre la máquina" [PYL85].

Es por lo anterior que normalmente se emplea una persona que se encarga del manejo de la base de datos -generalmente un programador-. Los problemas no se hacen esperar, como lo ejemplifica Hendrix en "Natural Language processing: the field in perspective" [HEN81a]:

> "Supongamos que un ejecutivo (usuario final) desea saber con urgencia cuantos productos "X" se han vendido en el mes. Se dirige al lugar donde debería estar el programador y resulta que éste salió a tomar un café o se encuentra demasiado ocupado para atenderlo. Finalmente el programador atiende su pregunta, y cuando se obtiene la respuesta, resulta que: el ejecutivo ya consiguió la información de otra fuente, o bien el programador no entendió bien la pregunta del ejecutivo, y presenta información inservible, o la información ya no es importante, pues ya es tarde para emplearla."

Los altos costos de manejo de información se deben en gran parte a que resulta necesario emplear personal especializado para que maneje los programas.

Una interfaz hombre-máquina es un programa diseñado para mejorar la comunicación entre el usuario y la computadora, y de lo expuesto anteriormente, resulta lógico pensar que los sistemas que cuenten con interfaces hombre-máquina de fácil utilización serán los más competitivos en el futuro inmediato [SAN82].

Se llega entonces a la conclusión de que las interfaces hombre-máquina deben ser diseñadas con un modelo de comunicación persona-persona, ya que es el tipo de interacción en que todos somos expertos.

Existen dos tendencias básicas en el diseño de interfaces hombre-máquina, como se denominarán los sistemas de comunicación entre el hombre y la computadora:

- El enfoque tradicional, que descansa sobre técnicas tales como menús de opciones, lenguajes de comandos, etc.

- El enfoque lingüistico -actualmente en investigación y desarrollo-, que emplea algún lenguaje natural (español, inglés, etc.) [RIC84].

Las interfaces en lenguaje natural se han desarrollado con el fin de reducir al máximo los requerimientos impuestos al usuario, cuando éste trata de comunicarse con la computadora. Todos dominamos el lenguaje natural; es nuestra lengua materna y, por lo tanto, es fácil de emplear y expresivamente muy poderoso. Hendrix lo resalta con la siguiente frase:

> "El lenguaje natural es el medio de comunicación del carnicero, el panadero y el cerero; del poeta y el amante; del político y el religioso; del padre y el hijo."[HEN81a].

En un estudio de evaluación del lenguaje natural en el mejoramiento de las interacciones hombre-máquina, se llega a la conclusión de que es recomendable usar, si no la totalidad del lenguaje natural, si un subconjunto de él en el diseño de interfaces eficientes [SAN82]; de hecho, Sanford propone que cuando menos se usen frases declarativas, en vez de imperativas. En este mismo estudio se afirma que el usuario llega a desarrollar "neurosis de terminal" cuando emplea un lenguaje formal, lo que bloquea su proceso de adaptación. Incluso algunas de las personas que iban a trabajar con la computadora en este estudio dimitieron, comentando escandalosamente que la "estúpida computadora" no cooperaba.

Es entonces cuando salta a la vista la necesidad de que las computadoras se adapten a los usuarios más que los usuarios se adapten a las computadoras.

Por otro lado, una condición necesaria para lograr generalización en el uso de las computadoras es que éstas sean de fácil uso, inclusive para personas que no saben ni desean saber nada de electrónica, lenguajes de programación y sistemas de comunicaciones.

Lograr que una computadora entienda el lenguaje natural es por ende de gran importancia en el desarrollo de interfaces agradables al usuario final. Si se logra que el usuario final pueda inquirir a la computadora lo más naturalmente posible, la información obtenida será más oportuna y precisa, redundando en su utilidad. Esto propiciará además la aceptación total de los sistemas de bases de datos y, con ellos, los medios automáticos de manejo de información.

En este trabajo se emplea el enfoque linguistico en el desarrollo de una interfaz, utilizando el español como medio de comunicación entre el usuario y la computadora. En particular, se concetra en el problema de consulta de bases de datos, entendiendo una base de datos como un sistema un sistema cuyo propósito general es el de registrar y mantener información [DAT82].

1.1 COMPRENSION DEL LENGUAJE NATURAL.

Uno de los mayores logros de la humanidad ha sido el poder comunicarse en un lenguaje natural. Muchas de las tareas cotidianas del ser humano no podrán ser realizadas por la computadora a menos que tenga la capacidad de usar el lenguaje natural.

Un niño de tres años puede hablar y entender el lenguaje sin tener aún la capacidad de jugar una partida reglamentaria de ajedrez. Por otro lado, aunque se ha logrado que las computadoras dominen la mecánica del ajedrez, no se ha podido escribir un sólo programa que pueda manejar fluidamente el lenguaje natural.

Los intentos más serios para lograr que las computadoras conversen en algún lenguaje se han encontrado con grandes dificultades, y los mejores prototipos de laboratorio son todavía un pálido reflejo de la habilidad lingüística de un niño normal.

El entendimiento del lenguaje natural es una tarea difícil. Se requieren conocimientos lingüísticos del idioma en particular, así como conocimientos sobre el mundo que se relacionan con el tema que se discute.

La mayor parte de la comunicación en lenguaje natural se realiza en forma de comunicación hablada; la escritura es un invento relativamente reciente y mucho menos empleado. El campo de la comprensión automatizada del lenguaje natural se divide en esas dos ramas:

- **Entendimiento del texto escrito,** que requiere conocimiento léxico, sintáctico y semántico del lenguaje -el texto-, así como información sobre el mundo -el contexto-.

- **Entendimiento del lenguaje hablado, que requiere toda la información citada arriba, más conocimientos adicionales sobre fonología, y mayor cantidad de información para resolver ambigüedades propias del lenguaje hablado.**

El presente trabajo se relaciona con la comprensión del lenguaje natural escrito.

Comprender algo significa poder transformarlo de una representación a otra, donde esta segunda ha sido elegida para que corresponda con un conjunto de acciones realizables por el sistema y donde el mapeo se ha diseñado de tal manera que para cada evento se realice una acción apropiada.

El lenguaje se debe entonces apreciar como un par (representación-original, representación-final), junto con unas reglas de correspondencia entre los elementos de una representación a otra.

Comprender un lenguaje es una noción un tanto difícil de definir. Esto se debe a que, en el caso de la comprensión del lenguaje natural, el contexto juega un papel primordial. El contexto es todo aquel conjunto de conocimientos que definen la situación en que se desarrolla una conversación (texto).

La comprensión, por lo tanto, no se puede definir en términos absolutos. Es necesario tener en mente que el entendimiento de un lenguaje natural se realiza en un momento determinado y con respecto a una tarea en particular. Por ejemplo, si se desarrolla una interfaz de lenguaje natural para mover el brazo mecánico de un robot, el sistema sólo requerirá comprender aquellas oraciones que se relacionen con el objetivo de dicho sistema:

- Mueve el brazo a la izquierda.
- Toma el cubo amarillo.
- etc.

y dicho sistema no tendrá necesidad de entender oraciones como: "¿Cuál es la capital de Honduras?".

El problema que se enfrentará es el de diseñar una interfaz para consultar bases de datos. Es por eso que el área de comprensión del lenguaje natural empleada en esta tesis estará definida sólo por aquéllas oraciones que se relacionan con peticiones de información.

1.2 HISTORIA DEL PROCESAMIENTO DEL LENGUAJE NATURAL.

El estudio del lenguaje natural mediante computadoras

se inició desde poco después de la aparición de éstas. Este estudio recibe el nombre de linguistica computacional, y se inició desde los años 40. En 1949, Warren Weaver propuso que las computadoras podrían ser empleadas para resolver problemas "mundiales" de traducción [WEA55].

Los primeros intentos para procesar el lenguaje natural en una computadora se realizaron precisamente en el campo de la traducción automática: una computadora debia simular el trabajo de un traductor. Las primeras ideas a ese respecto fueron tan sencillas como buscar cada palabra en un diccionario bilingüe para luego reordenarlas de acuerdo a la sintaxis del lenguaje de salida.

Evidentemente, estos métodos empezaron a sacar a la luz problemas ocultos hasta ese momento, tanto para solucionar las equivalencias de palabras, como para arreglarlas al producir una oración en el lenguaje de salida. Resultó obvio que habia que involucrar más elementos para resolver el problema.

Fué entonces cuando surgió la idea de que el **entendimiento** era necesario para manejar el lenguaje. Se propuso una representación intermedia entre el lenguaje original y el lenguaje objetivo. Dicha representación pretendia ser un lenguaje universal llamado **Machinese**, según la propuesta de Warren Weaver en una conferencia realizada en 1952.

Si una máquina lograba "entender el significado de una oración" era lógico que pudiera entonces parafrasearla, responder preguntas sobre ella e incluso traducirla a otro lenguaje. Sin embargo, la naturaleza del entendimiento es en si un problema complejo. En los 60's surgieron nuevos enfoques en el procesamiento del lenguanje natural e incluso de toda el área de la Inteligencia Artificial, influidos por el desarrollo de lenguajes de programación de alto nivel, el procesamiento de listas, la gran expansión del poder de las computadoras y su capacidad de memoria, y los hallazgos de Chomsky en teoria ligüistica [CHO69].

A continuación se presenta un conjunto de sistemas desarrollados a partir de esta época. En esta compilación se pueden apreciar claramente las tendencias seguidas por los investigadores del campo: inicialmente estaban interesados en el desarrollo de representaciones gramaticales (hasta el sistema LUNAR de Woods), y en una segunda etapa (a partir del sistema SHRDLU de Winograd) se interesan por la evolución de modelos más poderosos para representar el conocimiento.

Fué durante este tiempo que Joseph Wiezenbaum desarrolló su sistema ELIZA, que fué un ejemplo claro del empleo de una técnica llamada **Apareamiento de Patrones**. ELIZA es un sistema que **aparenta** la comprensión del lenguaje natural mediante especificaciones incompletas

de la estructura de una oración, llamadas patrones. Este programa asume el papel de un terapeuta Rogeriano o "no directivo" en su diálogo con el usuario (su paciente).

Daniel Bobrow escribió en 1968 el programa STUDENT como su proyecto de investigación doctoral en el "Masachussets Institute of Technology" (MIT). STUDENT puede (también mediante el apareamiento de patrones) leer y resolver problemas algebráicos de secundaria como el siguiente:

> "Si el número de clientes de Tomás es dos veces el 20 por ciento del número de anuncios que entrega y el número de anuncios que entrega es 45, cuál es el número de clientes de Tomás ?"

SIR fué escrito por Bertram Raphael en 1968, como parte de sus investigaciones en el MIT. SIR es una máquina prototipo de entendimiento, ya que puede acumular hechos y hacer deducciones sobre ellos para luego contestar preguntas. También emplea apareamiento de patrones en su análisis.

Fué en esta época cuando Robert Lindsay desarrolló su programa SAD-SAM en el "Carnegie Institute of Technology". SAD-SAM fué desarrollado siguiendo los trabajos de Chomsky sobre **gramáticas libres de contexto**. El programa acepta oraciones en inglés que hablan sobre relaciones de parentesco, crea una base de datos y responde preguntas sobre los hechos que ha almacenado.

Otra de las tendencias en boga era utilizar reglas de tipo estructuralista en el análisis sintáctico. Tal es el caso del programa llamado BASEBALL escrito por Bert Green y sus colegas en los Laboratorios Lincoln. BASEBALL es un programa de recuperación de información, ya que su base de datos -que contiene información sobre todos los juegos de la Liga Americana durante un año- no se modifica.

En 1972 William Woods diseñó LUNAR, para ayudar a los geólogos a accesar, comparar y evaluar datos de análisis químicos sobre la composición de roca y tierra lunar obtenidas en la misión Apollo-11. El formalismo gramatical que emplea son las **redes de transición aumentada** (ATN's, por sus siglas en inglés), presentadas por el mismo Woods en 1970 [WOO70]. Este es de los formalismos más empleados en la actualidad. Este es el punto cumbre en el desarrollo de formalismos de representación gramatical. El sistema PARSIFAL, desarrollado por Marcus en 1985 [MAR85] propone algunas mejoras a las redes de transición que no han sido del todo aceptadas.

SHRDLU es un sistema desarrollado por Terry Winograd en MIT alrededor de 1970. SHRDLU fué uno de los primeros sistemas que manejaban contexto y direcciona un dominio de mayor complejidad lógica que la que puede ser manejada por un sistema de base de datos. Según la clasificación de Hendrix, SHRDLU es un sistema que trabaja con micromundos

dinámicos [HEN81a]. Su formalismo gramatical fué desarrollado por el mismo Winograd, y se llama PROGRAMMAR. Además, emplea muchas representaciones de conocimiento, donde almacena datos sobre el estado de su micromundo [WIN72]. A partir de este momento, la investigación en el campo se enfoca al desarrollo de representaciones del conocimiento.

En la Universidad de las Américas (Puebla) se desarrolló en el presente año un sistema semejante al de Winograd, sólo que el micromundo es en dos dimensiones y el formalismo gramatical es también diferente (Modelo de Marcus) [MAR85].

Roger Schanck y sus estudiantes en el laboratorio de IA en Stanford, desarrollaron en 1975 un programa llamado MARGIE [SCH75]. Schanck y sus colaboradores intentaban realizar un modelo intuitivo del proceso del entendimiento del lenguaje natural. Para representar el significado de frases y oraciones, utilizaron la teoría de **Dependencia Conceptual** que Schanck había desarrollado en 1973, y extendido en 1975 [SCH75, RIC84].

Otra de las tendencias importantes en los sistemas desarrollados en los últimos años es la de emplear **marcos** para representar libretos [MIN75]; los libretos son situaciones típicas que se emplean para resolver referencias contextuales y completar información faltante.

Dentro de esta corriente, Schanck y Abelson desarrollaron un sistema llamado SAM (siglas que en inglés significan "mecanismo aplicador de libretos").

Otro de los ejemplos de este tipo de sistemas es el PLANES, que David Waltz diseñó para realizar consultas en lenguaje natural a una base de datos que contiene información sobre la operación de los aviones de una compañía [WAL78].

GUS es un sistema que, aplicando el uso de marcos para representar libretos, cae dentro de esta corriente. Desarrollado por Daniel Bobrow [BOB77], GUS ("Genial Understanding System") es un programa que ayuda a hacer reservaciones de boletos de aviación a los supuestos clientes de la compañía. El sistema debe guiar la conversación con el fin de extraer la información necesaria para apartar un boleto para un viaje redondo, partiendo de un punto prefijado.

1.3 TRABAJOS ACTUALES.

El campo donde ha habido mayor número de desarrollos prácticos en el entendimiento del lenguaje natural, es el de interfaces para consulta de base de datos.

La mayoría de este tipo de interfaces se han desarrollado en inglés, aunque ya existen algunas en italiano [LES81], e incluso multilingües.

A continuación se enlistan algunos de los trabajos que actualmente se están empleando a nivel comercial en en área de interfaces de consulta para bases de datos, y algunas de las razones de su éxito.

- LADDER, desarrollado por Hendrix en el "SRI International". Fue diseñada para realizar consultas a una base de datos distribuida. Algunas de sus características importantes son su capacidad para corregir posibles errores de ortografía y para realizar razonamientos elipticos.

- El sistema RENDESVOUZ versión 1, desarrollado en el Centro de Investigaciónes de la IBM en San José.

- El sistema ROBOT, desarrollado en el "Dartmouth College". ROBOT es comercializado por "Artificial Intelligence Corporation" como una interfaz para varios sistema de base de datos.

- INTELLECT, que es una nueva etapa de ROBOT, está siendo comercializado por IBM con mucho éxito.

- ENGLISH y Français, interfaces para la base de datos RAMIS II.

- THEMUS, interfaz para el sistema de base de datos Oracle, con capacidad de aprendizaje.

- TEAM, cuyo objeto es ser una interfaz transportable de fácil implementación en nuevos dominios.

Algunas razones para que estos sistemas tengan este gran éxito en su desarrollo son las siguientes:

- Se manejan dominios limitados del discurso en vez de tratar de entender grandes áreas del lenguaje natural.

- Se interactúa más con el usuario. Los sistemas antiguos pretendían ser completamente automáticos.

- Se han desarrollado métodos más eficientes para procesar las gramáticas y su semántica.

- Se pone mayor énfasis en las técnicas para resolver ambigüedades y pronombres.

- Las computadoras actuales trabajan a mayor velocidad y tienen mayor capacidad de almacenamiento de datos.

En este trabajo se aplicará el enfoque lingüístico en el desarrollo de una interfaz, utilizando el español como medio de comunicación entre el usuario y la computadora. Se utiliza la comprensión del lenguaje natural escrito debido a que de ésta manera se eliminan algunas fuentes de ambigüedad propias del proceso del habla.

Se verá cuál es el estado del arte de estas técnicas, para después identificar sus limitaciones cuando se aplican a la consulta de bases de datos, y se propondrán modificaciones a estas técnicas que harán que un sistema de consulta de bases de datos en lenguaje natural pueda ser generalizado.

1.4 ESTRUCTURA DE LA TESIS.

En el capitulo 2 de este trabajo, se plantea la problemática que involucra el desarrollo de interfaces de lenguaje natural en general. Su objetivo es proporcionar una perspectiva clara del campo en que se ubica este trabajo dentro del área de procesamiento de lenguaje natural de la Inteligencia Artificial. Para ello, se presenta una nueva perspectiva del campo tomando en cuenta la amplitud de la covertura lingüistica de los programas desarrollados. Posteriormente se enfoca la atención en el problema especifico de que se ocupa este trabajo: el desarrollo de un analizador semántico que proporcione a la interfaz la posibilidad de ser independiente de la estructura de la base de datos con que opere, para finalmente plantear un método de solución, delimitando el alcance, restricciones y caracteristicas del trabajo desarrollado.

Una vez hecho lo anterior, el capitulo 3 se ocupa de la descripción de la estructura del lenguaje natural; para ello, se presenta primeramente una perspectiva de la evolución de la ciencia que se ocupa del estudio del mismo: la lingüistica. Esto tiene como fin ubicar al lector en el contexto actual de esta ciencia, para poder entrar de lleno a la descripción de reglas que describen el lenguaje, a la vez que pueden ser empleadas para analizarlo. Finalmente se expone la descripción del método más empleado para la

representación de gramáticas en los sistemas de comprensión del lenguaje: las redes de transición aumentada. Esta descripción permite comprender con claridad la manera como se realiza el análisis sintáctico en los sistemas de procesamiento del lenguaje natural. De hecho, como se verá más tarde, el analizador sintáctico es una de las etapas de análisis del lenguaje más evolucionadas hasta la fecha, y su salida constituirá la entrada del analizador semántico que constituye el objetivo de esta tesis.

Una vez que se ha definido la etapa de análisis que precede al analizador semántico, se describirá la etapa que le sigue (su salida); éste es el tema del capítulo 4. La salida del analizador semántico está definida dentro del álgebra relacional, un lenguaje de manipulación de datos desarrollado especificamente para bases de datos con estructura relacional. Para poder hablar de esto, en un principio la atención se enfoca en los sistemas de bases de datos; se habla someramente sobre las principales estructuras de bases de datos, para posteriormente describir en detalle la estructura relacional. Finalmente se explican las operaciones del álgebra relacional que empleadas como salida del analizador semántico desarrollado.

Hasta este punto se habrá definido tanto la entrada como la salida del proceso de análisis semántico. El capítulo 5, que es el punto focal de esta disertación, está

estructurado como un conjunto de secciones autocontenidas a través de las cuales se expone la teoría desarrollada, y en base a la cual se construyó el prototipo de analizador semántico objeto de este trabajo. El esquema de una base de datos será considerado como una estructura reticular por la cual se pueden realizar viajes para formular la expresión de salida. Primeramente se explican algunos de los conceptos básicos del modelo conceptual empleado para después exponer gradualmente la teoría desarrollada. Finalmente se desgloza el proceso de análisis semántico y se aplica a algunos ejemplos.

El capítulo 6 contiene las conclusiones a que se llegó al final de este trabajo, así como la identificación de trabajo futuro.

A continuación se incluyen las referencias empleadas en esta tesis. La forma en que se codifican es la siguiente: [LLLnnl], donde "LLL" son los tres primeros caracteres del apellido del primer autor del trabajo, "nn" son los dos últimos dígitos del año de publicación, y "l" es una letra minúscula consecutiva empleada en caso de que los datos anteriores se repitan en más de un trabajo.

En seguida se presenta una bibliografía anotada, con la misma codificación que las referencias, que puede servir como base para orientar a quien desee incursionar en el campo del procesamiento del lenguaje natural en general, así

como en el área particular de consultas de bases de datos en lenguaje natural.

Algunas de las afirmaciones formuladas en esta tesis sobre el lenguaje se basan en datos obtenidos en una encuesta realizada al inicio de este trabajo. El análisis de los resultados, así como la justificación de la encuesta se presentan en el apéndice A, el que será referenciado en su oportunidad. Si bien la encuesta no tiene un volumen que le dé solidez estadística, las hipótesis planteadas en base a la misma no carecen de validez, debido a que la intuición las apoya; no quiero decir con esto que no se procedió científicamente: como se verá más tarde, una de las más sólidas teorías lingüísticas actuales se basa en las intuiciones del hablante sobre su lengua nativa.

Finalmente, en el apéndice B se muestran algunos ejemplos del desempeño del programa.

2.1 PROPOSITO.

El objetivo de este capítulo es plantear la problemática que envuelve el desarrollo de sistemas de comprensión de lenguaje natural. Para ello, se presentan dos enfoques básicos para su tratamiento: la comprensión del lenguaje orientada a la aplicación (limitada) y la comprensión amplia del lenguaje natural. Se presenta el modelo típico del proceso de comprensión del lenguaje natural en ambos enfoques, y se circunscribe el tema de esta tesis al enfoque limitado de comprensión del lenguaje natural.

En seguida se presenta la problemática específica que aqueja los sistemas de esta clase cuando su aplicación es la consulta de bases de datos.

Por último se propone una forma de solución modificando el modelo actual de reconocimiento de lenguaje natural, y se delimita el alcance del presente trabajo en el marco de la solución propuesta.

2.2 EL PROBLEMA DE LA COMPRENSION DEL LENGUAJE.

Existen procesos inteligentes que son fáciles de explicar: por ejemplo, cuando un dentista decide extraer una muela es debido a que ha llegado a la conclusión de que

es necesario hacerlo. Este "llegar a una conclusión" es un proceso de razonamiento lógico, en el que se observan las condiciones del paciente de manera objetiva y se les aplican una serie de "reglas" (que se han aprendido en la escuela o con la práctica) que al ser evaluadas en conjunto llevan a la conclusión correcta (en el mejor de los casos). En gran medida, se puede discutir lo que se ha hecho, decidir qué hacer después y aún enseñar sus conocimientos a otra persona. Para llegar a ser dentista es necesario estudiar una carrera en que se aprenden, entre otras cosas, las reglas que rigen ése tipo de razonamiento. Debido a que el proceso aplicado en la obtención de un diagnóstico está regido por una serie de reglas bien definidas, resulta entonces clara la razón por la cual es factible programar computadoras para hacerlas "expertas" en el diagnóstico de enfermedades.

Por otra parte, existen procesos mentales en que todos somos expertos y que, sin embargo, no logramos explicar de la misma forma; tal es el caso de los actos cotidianos de hablar y entender el lenguaje, ya que estos procesos no están regidos por el mismo tipo de razonamiento que el expuesto anteriormente; los hacemos todo el tiempo sin esfuerzo consciente, como respirar o caminar; pensamos explícitamente sobre su comprensión sólo cuando el proceso de comunicación falla de alguna manera.

impresión de que la comunicación lingüística es un proceso sencillo que no requiere muchos conocimientos. Cuando tratamos de aprender una lengua, sabemos que es necesario familiarizarse con alguna información nueva, incluyendo el significado de las palabras y la estructura de las oraciones. Todo parece ser muy sencillo: lo único que hacemos es hablar y escuchar y de hecho no sabemos hasta qué grado estamos utilizando también nuestros conocimientos sobre la comunicación y la lengua materna.

Sin embargo, cuando intentamos que una computadora entienda el lenguaje y produzca oraciones en lenguaje natural se presenta la necesidad de representar explícitamente todas las fuentes de conocimiento inconsciente. Hendrix identifica las fuentes principales de conocimiento necesarias para la comprensión del lenguaje natural como las siguientes [HEN81]:

- Conocimiento léxico, que se ocupa de las palabras de manera individual.

- Conocimiento sintáctico, que trata del agrupamiento de las palabras en frases con significado.

- Semántica composicional, que indica cómo componer el significado literal de unidades sintácticas a partir de la semántica de sus elementos.

- Conocimiento del discurso, que se ocupa de la forma en que se obtienen claves del contexto presente para ayudar en la interpretación de una oración.

- Conocimiento del mundo, que trata de la manera en que está configurado y de las restricciones físicas propias de las configuraciones posibles.

- Conocimiento de los estados mentales, que se relaciona con la comprensión del conocimiento y las metas de otros participantes en el diálogo.

Cuando hablamos o escuchamos a alguien, manejamos por lo menos todas estas fuentes de conocimiento. Normalmente se emplean todavia mas fuentes de conocimiento, que nos ayudan a llevar a cabo una comunicación fluida con nuestros semejantes.

Lo anterior da una idea clara de lo difícil que resulta modelar un comportamiento humano tan cotidiano, y a la vez tan complejo, como es el entendimiento del lenguaje natural.

2.3 TENDENCIAS BASICAS DE COMPRENSION DEL LENGUAJE.

Resulta entonces que el problema de la comprensión automatizada del lenguaje natural es muy extenso: es necesario tener una gran cantidad de conocimientos representados en la computadora de manera accesible,

flexible y además económica. En lo concerniente a técnicas de representación del conocimiento ha habido grandes avances; sin embargo, como se verá más tarde, queda mucho camino por recorrer [BAR81].

Existen dos tendencias básicas en el diseño de sistemas de procesamiento del lenguaje natural, tomando en cuenta la amplitud de su covertura lingüística:

- **Comprensión amplia del lenguaje natural.** En este enfoque se pretende entender el lenguaje natural como un todo, con el fin de elaborar programas que se asemejen al ser humano en su capacidad para manipular y comprender el lenguaje. El análisis es auxiliado por bases de conocimientos que ayudan a esclarecer las ambigüedades del lenguaje mediante un conocimiento claro del contexto.

- **Comprensión del lenguaje natural orientada a la aplicación.** Dentro de este campo, se desarrollan programas que entienden subconjuntos del lenguaje natural en contextos restringidos, con el fin de que funcionen como interfaces de uso cotidiano entre el usuario y un programa cualquiera (programa objetivo).

El primer enfoque ha impulsado el desarrollo de teorías del conocimiento y modelos de representación del mundo. De él se han derivado los conceptos que actualmente son la base de la Inteligencia Artificial. Es además el camino seguido por aquéllos que intentan desarrollar teorías nuevas sobre el lenguaje.

El segundo, es un camino práctico que emplea la subdivisión del lenguaje natural para lograr definir áreas útiles en aplicaciones concretas (controlar un brazo mecánico o consultar una base de datos, por ejemplo).

En el campo del entendimiento "puro" del lenguaje natural se han empleado teorías gramaticales completas, como la gramática transformacional (que más adelante se detalla). Los modelos de representación son de dependecia conceptual, marcos o redes semánticas, por ejemplo [WIN84, BAR81, FAR86]. Las herramientas empleadas para la representación gramatical son las redes de transición aumentadas (explicado en el capítulo 3 de esta tesis) o el modelo de Marcus [MAR85].

Por otro lado, en el entendimiento "restringido" del lenguaje natural se emplean principalemente gramáticas semánticas; prácticamente no se emplean representaciones del conocimiento, y las gramáticas se representan en árboles de transición -un subconjunto de las redes de transición-.

Si se pretende entender el lenguaje natural en toda su extensión, es necesario emplear un enfoque sensitivo al contexto; el discurso desplaza a la oración en su papel como elemento básico de comunicación [VAN80]. En cambio, se tiene una intención tan concreta como la de consultar bases de datos en lenguaje natural, no se necesitan modelos tan complejos de comunicación y se podrá emplear la oración como unidad básica de comunicación: el contexto no tiene gran importancia.

El problema que se trata este trabajo es el de diseñar una interfaz para consultar una base de datos en lenguaje natural. Debido a que esta tarea cae dentro de la comprensión limitada del lenguaje natural, este enfoque será empleado como el apropiado para el fin mencionado. Sin embargo, a continuación se contempla por separado cada uno de los dos enfoques, con el fin de tener una visión más clara del trabajo en esta área.

2.3.1 SISTEMAS DE COMPRENSION AMPLIA DEL LENGUAJE -

2.3.1.1 ESTRUCTURA. -

Un programa de comprensión amplia del lenguaje necesita varios componentes que corresponden a los diferentes niveles a los que se analiza el lenguaje.

El primer nivel de análisis es el morfológico. En este nivel, el programa aplica reglas que descomponen la palabra en su raíz o forma básica y sus inflexiones. Estas reglas corresponden en gran parte a las que se enseñan a los niños, quienes aprenden, por ejemplo, que la raíz de "jugando" es "jugar" mientras que la raíz de "nadando" es "nadar". Las palabras a las que no se pueden aplicar estas reglas se manejan en una lista de excepciones.

Un segundo nivel de análisis lo constituye la **fase léxica,** en la que a cada una de las palabras raíces obtenidas se le asigna el conjunto de categorías léxicas a las cuales pertenecen. Cuando una raíz tiene más de una categoría léxica surge la **ambigüedad léxica,** la que se resuelve ayudándose de las inflexiones que la acompañan y de referencias contextuales.

La salida del análisis léxico sirve como entrada a una tercera etapa: el **análisis sintáctico.** En esta fase se aplican reglas gramaticales que determinan la estructura de la oración. Al diseñar un analizador sintáctico surgen dos problemas diferentes:

El primero es la especificación de un grupo preciso de reglas -una gramática- que determinan el conjunto de oraciones válidas en el lenguaje. Durante los últimos 30 años se ha realizado mucho trabajo en lingüística teórica dirigido al diseño de sistemas lingüísticos formales

(construcciones en que las reglas sintácticas de un lenguaje se establecen de manera tan precisa que una computadora pueda emplearlas para analizarlo). Los primeros intentos valiosos en este sentido fueron las gramáticas generativas transformacionales inventadas por Noam Chomsky del "Masachusets Institute of Technology", que especifican la sintaxis de un lenguaje mediante un conjunto de reglas cuya aplicación mecánica genera todas las estructuras permisibles. Estos formalismos han sido desarrollados por lingüistas para estudiar el lenguaje utilizando la computadora sólo como herramienta; existen otros formalismos más novedosos diseñados especificamente para lograr que una computadora entienda el lenguaje.

El segundo problema es el de realizar el análisis en sí, pues no siempre se puede saber, al encontrar una parte de la oración, qué papel desempeña en ella. Los analizadores sintácticos aplican varias estrategias para explorar las diversas formas en que se acoplan las frases. Algunos emplean el retroceso ("backtracking") para explorar alternativas cuando falla una posibilidad determinada; otros usan procesamiento en paralelo, para seguir simultáneamente un número de alternativas; otros más exploran el resto de la oración buscando información que les ayude a determinar el camino a seguir ("wait-and-see"), mientras que otros exploran la cadena de entrada por adelantado con el fin de resolver conflictos en el momento en que se presenten

("look-ahead"). Para profundizar en el tema se recomienda recurrir a [LEW78, AHO77, WIN83].

Cuando se han empleado formalismos gramaticales desarrollados con el fin de lograr que las computadoras comprendan el lenguaje, se tienen grandes ventajas, ya que los procedimientos de análisis son prácticamente una consecuencia lógica de dichos formalismos. Tal es el caso de las "redes de transición aumentada" [WOO70] que expresan la estructura de las oraciones y frases como una secuencia explícita de transiciones seguidas por la máquina entre los estados de un autómata finito. Existen infinidad de formalismos de este tipo, como la "gramática de casos", la "gramática de funciones léxicas" o la "gramática de estructura de frases" [WIN84].

Aunque ninguna gramática formal maneja exitosamente todos los problemas gramaticales que se presentan en los lenguajes naturales, los analizadores sintácticos existentes pueden manejar hasta un 100 por ciento de todas las oraciones [SOW86]. Lo anterior se logra, por ejemplo, en un analizador sintáctico del inglés llamado PLNLP ("Programming Language for Natural Language Processing") mediante el empleo de una técnica llamada "análisis sintáctico ajustado" ("fitted parsing"), que se emplea para procesar expresiones idiomáticas que tendrían significados distintos a los normales, y algunas otras irregularidades

del lenguaje natural.

En la etapa de análisis sintáctico surge la ambigüedad estructural. Este tipo de ambigüedad se presenta cuando una sola oración puede ser representada en más de una estructura sintáctica. Normalmente, esta ambigüedad se puede resolver en función del contexto.

La salida del analizador sintáctico se convierte en la entrada del cuarto componente de un programa de comprensión del lenguaje: **el analizador semántico.** En esta etapa se traduce la forma sintáctica de una oración en una forma "lógica". El objetivo de esta etapa es convertir las expresiones lingüísticas en una estructura que permita que la computadora aplique procedimientos de raciocinio y realice inferencias. Hay teorías que compiten acerca de cuál es la representación más apropiada, siendo las metas en este campo la eficacia y la eficiencia.

Una buena parte de la investigación en análisis semántico se ha dedicado al diseño de "lenguajes de representación" para proveer una forma eficiente y eficaz de codificar el significado. La dificultad principal estriba en el problema de definir la naturaleza del sentido común del ser humano, ya que la mayoría de lo que una persona sabe no puede ser expresado en reglas lógicas absolutas; normalmente se basa en "espectativas comunes". Si uno pregunta "¿Hay tierra en el jardín?" la respuesta es casi siempre "si". Sin

embargo este "si" no puede ser una inferencia lógica, pues algunos jardines son hidropónicos y en ellos las plantas crecen en agua. Una persona tiende a confiar en sus espectativas comunes a menos que haya excepciones relevantes. Sin embargo, se ha realizado muy poco progreso en torno a la formalización de la "relevancia" y la forma en que ésta influye en el cúmulo de espectativas empleadas para analizar las expresiones lingüísticas.

La etapa final de un sistema de comprensión amplia del lenguaje natural es el **análisis pragmático**: el análisis del contexto. Todas las oraciones están envueltas en un conjunto de situaciones: vienen de un comunicador particular, en un momento específico y se refieren -por lo menos implícitamente- a un área particular del entendimiento. Parte de esta "envoltura" es fácil de obtener: el pronombre "yo" se refiere al comunicador; el adverbio "ahora" se refiere al momento en que se produjo la oración.

Las estructuras de representación son completadas entonces en el análisis pragmático, para convertirse en la entrada de la última etapa del proceso: **la etapa de razonamiento**. Es aquí donde se trata de "manejar" el significado de las oraciones de entrada con el fin de contestar preguntas, proporcionar información y otras operaciones que indican el grado de "asimilación" de la

información, en base a la forma empleada para representarla.

La figura 2.1 ilustra la arquitectura típica de esta clase de sistemas.

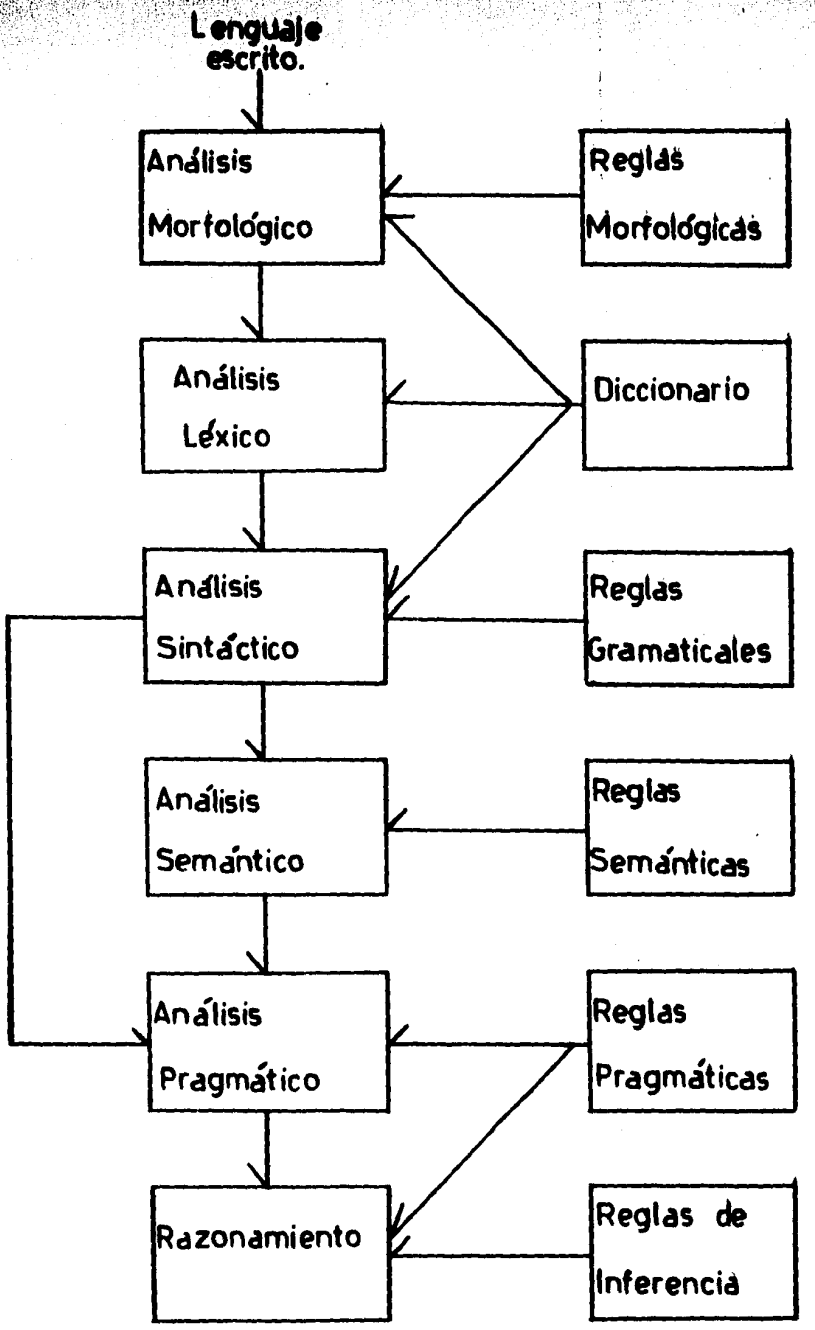


FIGURA 2.1.- Arquitectura de un sistema de comprensión amplia del lenguaje natural.

2.3.1.2 PROBLEMATICA. -

Los problemas de los sistemas de comprensión amplia del lenguaje natural se presentan a partir del análisis semántico. Desde hace algún tiempo se ha discutido sobre la forma de representación de la "lógica" de la oración de entrada. A este respecto existen dos tendencias básicas: los partidarios de la lógica de primer orden y aquellos que pretenden diseñar un nivel superior de lógica (dependencia conceptual, por ejemplo).

Cuando se emplea lógica de primer orden se tiene la gran ventaja de que este formalismo está fuertemente apoyado e instrumentado. La lógica se ha desarrollado por siglos, y ha habido tiempo suficiente para el desarrollo de teorías e instrumentaciones muy poderosas. Sin embargo, debido a su simpleza, el problema principal de esta técnica es su incapacidad para representar relaciones comúnmente empleadas en lenguaje natural.

Por otro lado, si se emplean formalismos lógicos más novedosos y elevados se elimina el problema de la expresividad (se pueden diseñar con el poder expresivo que uno desee). A pesar de esta gran ventaja, el problema de estos modelos es que no existe teoría que los apoye, y tampoco tienen una instrumentación tan rigurosa como los de la lógica formal.

Siguiendo el proceso de comprensión del lenguaje, se encuentra el análisis pragmático, en que existen partes del contexto que son muy complicadas: el pronombre "nosotros" es un ejemplo. "Nosotros" puede referirse al comunicador y al receptor o al comunicador junto con alguna tercera persona. No está explícito de quienes se trata y de hecho esto es una fuente común de confusión cuando las personas conversan.

Hay otros tipos de referencia contextual que no son señalados por palabras conflictivas como "nosotros". La frase nominal "aquéllos ojos verdes" tiene un significado dependiente del contexto, ya que puede haber sólo una instancia de ojos verdes o puede haber más de una. La oración presupone un conjunto de conocimientos de los cuales se pueden identificar los ojos a los que se refiere el locutor.

Un enfoque para solucionar estos problemas ha sido codificar conocimientos del mundo de tal manera que el programa pueda usarlos para hacer inferencias. Esto incluye el conocimiento de libretos (situaciones estereotipadas), metas y estrategias. Este enfoque presenta las mismas dificultades que las representaciones del conocimiento en la formalización del sentido común que determina cuáles libretos, metas y estrategias son relevantes y cómo interactúan. Es por eso que los programas de comprensión

amplia del lenguaje natural escritos hasta ahora trabajan sólamente en ambientes altamente artificiales y limitados.

Distinguir los usos literales del lenguaje de aquéllos metafóricos o poéticos no es un problema de fácil solución. Si asi fuese, los programas de computadora podrían enfrentar sólo el uso literal del lenguaje para estar libres de dilemas contextuales. Sin embargo, la metáfora y el "significado poético" no están limitados a la literatura. El lenguaje cotidiano está lleno de metáforas inconscientes. Un ejemplo es la oración: "Perdí dos horas tratando de aclarar mis ideas". Prácticamente todas las palabras tienen significados abiertos variando desde los literales hasta los claramente metafóricos.

Las limitaciones en la formalización del significado contextual imposibilitan en el presente -y probablemente para siempre- el diseño de programas de computadora que siquiera se asemejen al entendimiento humano del lenguaje.

2.3.2 SISTEMAS DE COMPRENSION LIMITADA DEL LENGUAJE -

Los únicos programas de comprensión de lenguaje natural que se usan prácticamente en la actualidad, se restringen al entendimiento limitado del lenguaje. Una aplicación típica son las interfaces que permiten al usuario pedir información haciendo preguntas en lenguaje natural; el programa responde entonces con oraciones en lenguaje natural o con un

desplegado de datos. El proceso es el siguiente:

Una persona que desea accesar información almacenada en la computadora teclea oraciones en lenguaje natural, que la computadora interpreta como peticiones de información ("queries"). El rango del cuestionamiento se circunscribe al rango de los datos a partir de los cuales se van a formular las respuestas; de esta forma se puede dar a las palabras significados precisos, eliminando gran parte de las fuentes de ambigüedad que aquejan a los sistemas de entendimiento amplio del lenguaje. En una base de datos de automóviles, por ejemplo, la palabra "obscuro" puede ser definida como los colores "negro" y "azul marino" y nada más que eso. El significado contextual está ahí, pero está predeterminado por el constructor del sistema, y se espera que el usuario lo aprenda.

2.3.2.1 ESTRUCTURA. -

La estructura de estos sistemas es muy similar a la de los sistemas de comprensión amplia. De hecho, los hallazgos realizados en el intento por comprender la totalidad del lenguaje han sido de gran utilidad para el desarrollo de programas que entienden subconjuntos del mismo.

El **análisis morfológico** de éste tipo de sistemas es omitido en algunas ocasiones, ya que la representación de la gramática es tan específica que tiene ya implícitos los diferentes morfemas de las palabras. En el caso en que se incluya, su función y principios de operación son los mismos que en los sistemas de comprensión amplia del lenguaje.

El **análisis léxico** sirve los mismos propósitos que en los sistemas de comprensión amplia del lenguaje natural. Debido a que las aplicaciones de estos sistemas son tan específicas, se eliminan prácticamente las fuentes de ambigüedad léxica. Esto se logra mediante procesos que sustituyen las palabras que podrían resultar anbigéas por palabras con significados precisos.

En el **análisis sintáctico** se resuelven los problemas que se plantean en la comprensión amplia del lenguaje natural de la siguiente forma:

La especificación de la gramática es más sencilla, puesto que el conjunto de oraciones posibles es más reducido y de estructura más simple. Esto se debe a que no se va a manejar la totalidad de un lenguaje, sino sólo un subconjunto de él. La representación sintáctica contiene en algunos casos información semántica. Esta es la tendencia seguida por muchos de los sistemas de este tipo, y una representación sintáctica con tales características recibe el nombre de "gramática semántica".

Los mecanismos empleados para realizar el análisis sintáctico utilizan las mismas estrategias que los sistemas de comprensión amplia. Obviamente se reducen las posibles fuentes de ambigüedad estructural. Además, la cantidad de información necesaria para especificar la gramática es mucho menor en estos sistemas.

En caso de que surja alguna ambigüedad estructural, existen ya modelos preestablecidos de interacción aclarativa con el usuario [COD74]. Por otro lado, se han desarrollado métodos para realizar satisfactoriamente el análisis de oraciones incompletas y aún de aquellas que no sean estrictamente gramaticales.

La salida del análisis sintáctico en los sistemas de comprensión limitada del lenguaje natural tiene diferentes formas. Algunas son muy semejantes a las de los sistemas de comprensión amplia del lenguaje natural, en cuanto que se expresan en términos estrictamente lingüísticos, tales como "sujeto" y "predicado". Por otro lado, existen formalismos gramaticales que al ser empleados en el análisis sintáctico generan una salida que contiene ya información sobre la semántica implícita en la oración de entrada.

La salida del análisis sintáctico se convierte entonces en la entrada del siguiente componente del sistema: **el analizador semántico.**

Cuando se habla de análisis semántico, es necesario recordar que se trata de interfaces que permiten que el usuario se comunique en lenguaje natural con un programa (manejador de base de datos, por ejemplo). Este es el programa objetivo de la interfaz.

El analizador semántico se encarga entonces de traducir la estructura entregada por el análisis sintáctico a una expresión que evoca el conjunto de acciones realizables por el programa objetivo. Esta es la etapa final de entendimiento de esta clase de sistemas. La oración ha sido traducida a un conjunto de acciones que el programa objetivo puede realizar.

En algunas ocasiones, el analizador semántico se encuentra mezclado con el analizador sintáctico. Esto significa que el análisis sintáctico y el semántico se realizan en una sola etapa, que se llama análisis sintáctico-semántico.

En la figura 2.2 se esquematiza un sistema típico de comprensión limitada del lenguaje natural. Las fuentes de información necesarias para cada etapa son completamente dependientes del contexto, por lo que el bloque denominado "Contexto" las incluye a todas.

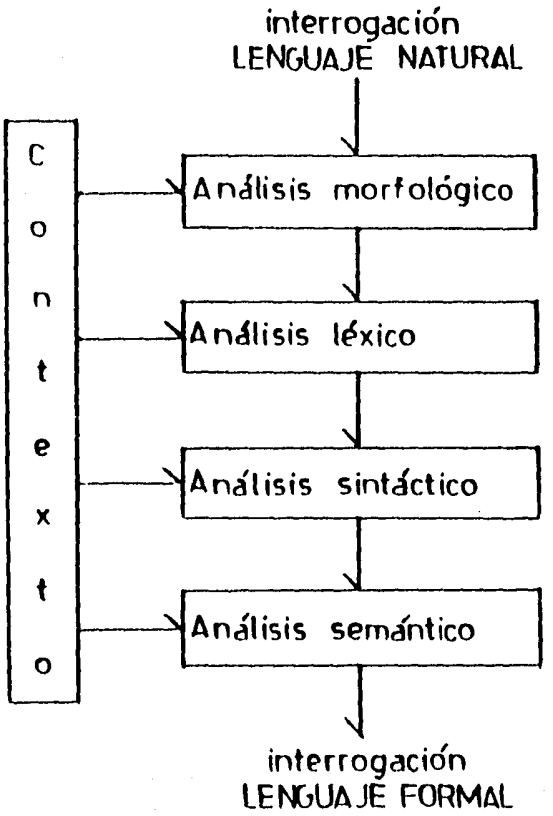


FIGURA 2.2.- Arquitectura típica de un sistema de comprensión limitada del lenguaje.

2.3.2.2 PROBLEMATICA. -

Debido a que el entendimiento en los sistemas de comprensión limitada del lenguaje natural es un proceso de traducción de éste a diferentes representaciones, una oración se entiende sólo con respecto a un conjunto de acciones. Esto es muy importante, y es uno de los problemas principales en la construcción de los sistemas de comprensión limitada de lenguaje natural. Cada conjunto de acciones diferente requiere un proceso diferente de comprensión.

2.3.2.2.1 LA INDEPENDENCIA DEL DOMINIO. -

Cuando estos sistemas son trasladados a un dominio (área de conocimientos) diferente, es necesario definir una gramática completamente nueva, con el fin de que el sistema pueda operar convenientemente. Esta definición debe ser hecha por personas que conocen a fondo el funcionamiento del sistema y tienen además fundamentos de lingüística, debido a que la especificación gramatical usada en ellos es muy restrictiva. Además, muchas veces contiene información sobre el dominio a discutir, lo que provoca que los analizadores sintácticos tengan una covertura lingüística irregular, y en ocasiones sea necesario aumentar la gramática para poder aceptar nuevas oraciones. De hecho, se han diseñado sistemas que inician un diálogo con el usuario

encaminado a aumentar las reglas de la gramática en cuanto no pueden realizar el análisis de una oración; sin embargo, el usuario tiene entonces que conocer cómo funciona el sistema y estar familiarizado con términos lingüísticos.

Las gramáticas semánticas -que unifican el análisis sintáctico y el semántico- contienen gran cantidad de información sobre la estructura conceptual del dominio de la aplicación. Por ejemplo: en vez de usar categorías generales tales como "sustantivo" y "frase verbal", las gramáticas semánticas pueden tener categorías tales como "especificación de color" o "lugar de fabricación". Dentro de los sistemas que emplean gramática semántica en su análisis sintáctico se encuentra el LADDER de Hendrix [HEN81]. El sistema LADDER fué desarrollado mediante otro programa (el LIFER), con el cual se redefine totalmente la gramática cada vez que se cambia el dominio del discurso. Esto provoca que los costos que resultan al trasladar este sistema sean muy altos, teniendo en cuenta el alto grado de especialización requerida para realizar esta tarea. De hecho, Hendrix propuso que se creara una nueva rama de la ingeniería ("Human Engineering") para formar profesionales encargados de acoplar sistemas como el LADDER a nuevos dominios de aplicación [HEN81].

Se han realizado esfuerzos importantes con el fin de lograr que un sistema de comprensión limitada del lenguaje natural sea independiente del dominio, y éste es uno de los retos principales para los diseñadores de interfaces de esta clase.

Existen investigadores convencidos de que la solución propuesta por Hendrix no es la única salida. Si bien es cierto que construir un sistema totalmente independiente del dominio es una tarea difícil, se plantea la posibilidad de que las personas que se encarguen de trasladar la interfaz no necesariamente sean expertos en lingüística. Esto implica que gran parte del conocimiento lingüístico necesario será parte de la interfaz. A continuación se analiza un sistema que utiliza este enfoque en el desarrollo del sistema TEAM [MAR85]:

El sistema TEAM es una interfaz para consultar bases de datos en lenguaje natural, y se basa en la afirmación de que la información necesaria para realizar el análisis sintáctico se puede dividir en dos partes principales:

- Información dependiente del dominio, que está principalmente constituida por elementos léxicos tales como sustantivos, verbos y adjetivos propios de cada área de conocimiento.

- **Información independiente del dominio, constituida por elementos léxicos tales como pronombres, conjunciones y artículos, cuyo significado es invariable; asimismo, toda la estructura sintáctica debe ser independiente del dominio.**

Empleando esta división de la información lingüística se han desarrollado sistemas altamente independientes del dominio, en los que el diseñador sólo provee aquella información que depende del dominio.

Los proyectos que (como TEAM) pretenden resolver el problema de la independencia del dominio en la consulta de bases de datos, toman la base de datos como un modelo del mundo que contiene objetos y relaciones. Las palabras de la oración de entrada expresan conceptos que están de alguna manera representados en la base de datos en forma de valores o relaciones. Entonces una palabra como "población" puede ser asociada con un conjunto de pares de cifras de población y nombres de paises, y una pregunta como "¿Qué población tiene España?" debe ser traducida a una expresión que se refiera a este conjunto. Un problema importante es decidir qué partes de la base de datos corresponden a qué partes de la oración. Este problema se resuelve en el análisis sintáctico mediante la introducción de nombres de roles que relacionan complementos de verbos, sustantivos y adjetivos con objetos de la base de datos.

2.3.2.2.2 LA INDEPENDENCIA DE LA ESTRUCTURA. -

Como se mencionó con anterioridad, una aplicación tipica de los sistemas de comprensión limitada del lenguaje es el desarrollo de interfaces para consultar bases de datos en lenguaje natural.

Además de tener problemas de independencia del dominio, existe otro problema que aqueja a las interfaces de lenguaje natural para consultar bases de datos: la independencia de la manera en que se estructura la información.

La independencia del dominio del discurso ha sido resuelta de manera satisfactoria en proyectos como el TEAM. Sin embargo, existe la posibilidad de que la estructura de la base de datos sea variable, aún cuando el dominio del discurso sea el mismo. Supongase que se tiene una base de datos que contenga información sobre una escuela: profesores, alumnos, materias, etc. Los profesores pueden estar directamente relacionados con los alumnos, en diferentes archivos; o pueden estar mezclados en el mismo archivo; o pueden estar ligados a través de otro archivo, llamado grupo. La interfaz debe tener la posibilidad de adaptarse a las diferentes situaciones con un mínimo de esfuerzo por parte del usuario.

2.4 SOLUCION PROPUESTA.

El desarrollo de una interfaz para consultar bases de datos en lenguaje natural es entonces un problema que cae dentro del área de comprensión limitada del lenguaje.

En la sección 2.3.2 se presentaron las técnicas empleadas en esta clase de sistemas, así como los problemas relacionados con la independencia tanto del dominio como de la estructura de la base de datos.

A continuación se propone un proceso de reconocimiento del lenguaje natural que presenta algunas variantes con respecto al proceso tradicional, encaminadas a solucionar esos problemas.

2.4.1 MODELO DEL SISTEMA PROPUESTO. -

En la figura 2.3 se presenta el modelo del proceso para el sistema propuesto. En este modelo se puede apreciar que las etapas de análisis son las mismas que se emplean en los sistemas de comprensión limitada del lenguaje natural. La diferencia reside en las fuentes de información que emplea cada una de las etapas.

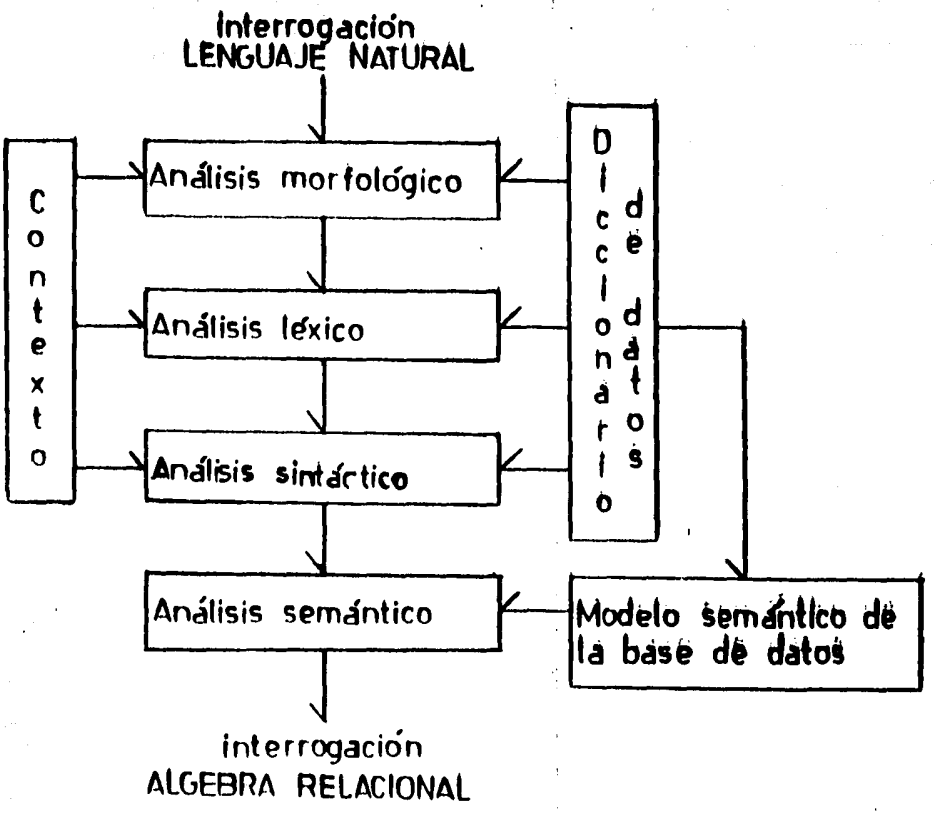


FIGURA 2.3.- Modelo del sistema propuesto.

La fuente de información denominada CONTEXTO, contiene toda la información independiente del dominio, tal como los articulos, las preposiciones y conjunciones, además de la estructura de la sintaxis a emplear en el análisis. Esta fuente contiene además alguna información dependiente del dominio, como son los sustantivos, adjetivos y verbos

propios de la especialidad. Esta información es obtenida mediante diálogos con el usuario o bien con el diseñador de la base de datos. El contexto abarca además información sobre los roles que relacionan complementos de verbos, sustantivos y adjetivos con partes de la base de datos, y que se obtienen en una interacción (en que no se pide información lingüística) con el usuario.

La fuente de información denominada CONTEXTO es empleada en las etapas de análisis morfológico, léxico y sintáctico.

Adicionalmente, estas etapas utilizan información residente en un diccionario de datos [GON85, GON85a, CUR81].

De esta manera, se logra independencia del dominio de aplicación de la base de datos, ya que el contexto es independiente y la parte dependiente se mantiene en un módulo que es interpretado para cada aplicación particular.

A partir del diccionario de datos, y mediante una interacción con el diseñador del modelo de la base de datos, se obtiene la tercera fuente de información: EL MODELO SEMANTICO DE LA BASE DE DATOS. En él se encuentra codificada la información sobre la semántica que implica cada relación o conjunto de relaciones en el esquema de datos.

El MODELO SEMANTICO es empleado como base para realizar el análisis semántico de la oración, con el fin de que a su salida se obtenga una formulación en Algebra Relacional (figura 2.4) que define el conjunto de datos que contestan a la pregunta original.

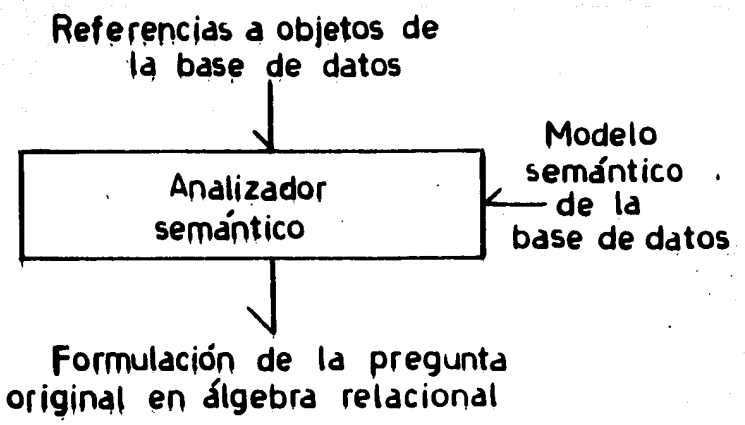


FIGURA 2.4.- Analizador semántico del modelo propuesto.

De esta manera se logra independencia de la estructura de la base de datos, ya que el modelo semántico se genera a partir del esquema particular de cada aplicación.

2.4.2 ALCANCE, LIMITACIONES Y CARACTERISTICAS. -

Este trabajo se concentrará en el desarrollo de un analizador semántico que opere independientemente de la estructura de la base de datos.

Se puede apreciar en la figura 2.3 que el análisis semántico requiere:

1. La salida del analizador sintáctico: referencias a partes de la base de datos y relaciones semánticas entre las partes aludidas de la base de datos.

2. El modelo semántico de la base de datos.

Es factible obtener la salida del analizador sintáctico utilizando técnicas existentes. El modelo semántico de la base de datos se obtiene, como se dijo anteriormente, a partir del esquema de datos (almacenado en el diccionario de datos) con la ayuda de una interacción con el diseñador de la base de datos, en que se definen aquellas relaciones semánticas que no se encuentran explícitas en el esquema de datos.

Un esquema de datos es una estructura que contiene información sobre las relaciones existentes entre los objetos representados en la base de datos, así como de la forma en que se representan.

El alcance de este trabajo es la realización de un programa que, independientemente de la estructura de la base de datos, sea capaz de encontrar y expresar qué operaciones se deben realizar sobre ella con el fin de hallar una información determinada. Para ello se considera el esquema de datos como una gráfica en que cada uno de los nodos representa un elemento de la base de datos, y los arcos que los unen representarán las relaciones existentes entre ellos (cada arco representará sólo una relación).

El esquema de datos empleado se basará en los conceptos usados en la definición del modelo ELKA ("Entity, Link, Key, Attribute) formalizado por Guillermo Rodríguez en [ROD81].

Se eligió el modelo conceptual ELKA debido a que, a pesar de su sencillez y reducido número de conceptos, tiene una capacidad expresiva equiparable a la de otros modelos conceptuales, tales como el E-R ("Entity-Relationship"), que resultan considerablemente más complicados (ver [GON85, BAT84a, CHA79, WIE84]).

La salida del analizador semántico será expresada en Algebra Relacional, y se empleará una base de datos de estructura Relacional.

La razón por la cual se eligió el Algebra Relacional como medio para expresar la salida es que es considerada "relacionalmente completa", en el sentido de que sus

operaciones le proporcionan por lo menos la capacidad de recuperación de información del Cálculo Relacional, como lo demuestra Codd en "Relational Completeness of Database Sublanguages" [COD72]. La sintaxis empleada es la utilizada en el lenguaje ASTRID [GRA84], debido a que su estructura infija facilita su comprensión.

El hecho de haber elegido una estructura de datos relacional para la base de datos está sustentado por la sencillez en que se pueden representar en ella las relaciones entre los elementos representados en la base de datos. Sus características principales, en contraposición a otras estructuras, las expresa Date [DAT82] como sigue:

1. Cada "archivo" contiene sólo un tipo de registro.

2. Cada ocurrencia de un registro dentro de un "archivo" dado tiene el mismo número de campos.

3. Cada ocurrencia de un registro tiene un identificador único.

4. En un "archivo", las ocurrencias de los registros tienen un orden no especificado, o bien están ordenadas de acuerdo a valores contenidos en esas ocurrencias.

Otra de las razones por las cuales se eligió la estructura Relacional para la base de datos es el gran auge que tiene sobre las otras estructuras de datos (Jerárquica y de Red), debido a que es de fácil comprensión y, sobre todo, a que forma un sistema matemáticamente cerrado.

Si se concibe entonces que el análisis semántico tendrá como entrada un conjunto de referencias a partes del esquema de datos (nodos), el analizador semántico se encargará de "viajar" por la estructura del esquema de tal manera que la travesía por cada arco (relación) implicará operaciones bien definidas que al ser ejecutadas sobre la base de datos definirán el subconjunto de la misma que conteste (dentro de las limitaciones del Algebra Relacional) la pregunta formulada por el usuario.

3 ESTRUCTURA DEL LENGUAJE NATURAL.

3.1 INTRODUCCION.

Este capítulo tiene como objetivo dar una idea clara de la forma en que se realiza el análisis sintáctico en un sistema de comprensión del lenguaje natural.

Primeramente se bosqueja la historia de la ciencia que se dedica al estudio del lenguaje: la Lingüística. En el bosquejo se emplea una visión particular de la evolución científica: los paradigmas de Kuhn.

En seguida se trata de manera somera la forma en que se estructura la oración simple del español, que es la estructura principal del lenguaje. Esto tiene como propósito sentar las bases necesarias para la explicación que se da a continuación sobre uno de los formalismos más empleados en la actualidad para representar las estructuras gramaticales en la computadora: las redes de transición aumentadas.

Finalmente se explican las redes de transición aumentadas, con el fin de aclarar la forma en que se realiza el análisis sintáctico en la mayoría de los sistemas de comprensión del lenguaje.

3.2 EVOLUCION DE LA LINGÜISTICA.

En esta sección se esbozará la evolución de la ciencia lingüística en base a los conceptos desarrollados por Kuhn. Se explicarán algunos de sus conceptos básicos necesarios para después poder presentar una historia breve de la evolución de la ciencia que se ocupa del estudio del lenguaje: la Lingüística.

3.2.1 LOS PARADIGMAS DE KUHN. -

Existe una visión popular de la ciencia en la que se aprecia su evolución como un progreso lineal. La naturaleza se aprecia en esta visión como un conjunto de fenómenos observables, y los científicos se dedican a construir teorías que explican sus regularidades y predicen lo que sucederá. Estas teorías mejoran con el tiempo, explicando más fenómenos, haciendo predicciones más exactas y volviéndose más claras.

Thomas Kuhn presentó una visión alternativa en su libro "The Structure of Scientific Revolutions" [KUH70], basada en la noción de revoluciones recurrentes que cambian totalmente los fundamentos sobre los que se apoya la ciencia. Afirma que la naturaleza no presenta al científico paquetitos de información con una etiqueta diciendo "explícame"; el científico se enfrenta a un mundo complejo e interconectado, y una parte principal en la definición de

cada ciencia se ocupa de la selección de las preguntas a realizar sobre el fenómeno y la clase de respuestas que se considerarán aceptables.

Kuhn describe un ciclo que incluye periodos de ciencia "normal", separados por revoluciones. Durante un período de ciencia "normal", existe un acuerdo general sobre las preguntas manejadas y las respuestas buscadas. Los fundamentos de la ciencia se dan por establecidos y las suposiciones básicas son muy poco cuestionadas. Durante este periodo, hay gran cantidad de trabajo por hacer en los detalles especificos de las teorías, y el científico progresa de acuerdo con el punto de vista general. Se dice que la ciencia está operando dentro de un **paradigma** particular cuando se encuentra en ese estado. Este período se conoce también como un período de ciencia "normal".

Posteriormente, cuando se agota la capacidad expresiva de un paradigma, los científicos vuelven sus ojos hacia aquellos "pequeños detalles" que el paradigma no puede explicar, con la idea de extenderlo. Una vez que se demuestra la imposibilidad de este enfoque, se inicia la instrumentación de nuevos paradigmas, cuya tendencia es la de mejorar el paradigma existente, o inclusive sustituirlo. Surgen nuevas teorías más completas que la anterior, y se realizan acaloradas discusiones entre los científicos conservadores y los revolucionarios. Finalmente, sólo uno

de la gran cantidad de paradigmas propuestos es aceptado como indiscutible, y se acepta mundialmente como el nuevo paradigma que regirá la ciencia en cuestión. Se regresa entonces a un nuevo período de ciencia "normal" y el ciclo se reinicia.

3.2.2 PANORAMA HISTORICO DE LA LINGUISTICA. -

El estudio de la Lingüística puede considerarse tan antiguo como la lengua misma, y la ciencia Lingüística actual puede encontrar sus orígenes en tiempos tan remotos como en los estudios de la gramática del sánscrito de hace dos mil años. Al analizar esta historia, se encuentran algunos puntos de evolución en que se cambia el enfoque del estudio y en que los lingüistas se sentían seguros de haber arribado a los temas "reales" del lenguaje. Observando el proceso de la ciencia en general, se puede llegar a algunas conclusiones útiles sobre la manera como se ha estudiado el lenguaje.

Una manera de apreciar los cambios en una ciencia es ver la secuencia de metáforas en que se basa. Un científico en busca de un nuevo paradigma está fuertemente influído por otras ciencias que tienen un desarrollo exitoso en ese momento particular. Consciente o inconscientemente, la ciencia en apogeo se aprecia como un modelo, y existe una imposición metafórica de sus ideas sobre qué preguntas

formular y qué tipo de respuestas son aceptables. La Lingüística y las otras ciencias humanísticas han sido siempre muy receptivas a este tipo de influencia, principalmente proveniente de las ciencias exactas.

La historia de la lingüística presentada aquí está estructurada como una serie de metáforas. Estas no son exactamente lo mismo que los paradigmas de Kuhn, ya que su teoría involucra la organización social de la ciencia más que sus fundamentos conceptuales. La conexión, sin embargo, puede resultar útil cuando se trata de entender por qué los paradigmas tienen tal fuerza unificadora entre la gente que trabaja en ellos.

La primera metáfora no forma parte de un período de tiempo determinado, sino que representa un enfoque regulativo que ha permanecido entre nosotros desde hace siglos y que continúa dominando el entendimiento popular actual del lenguaje. El resto de las metáforas entran en una secuencia que abarca todo el siglo pasado y lo que va del presente, cubriendo las direcciones principales del pensamiento lingüístico de ese tiempo. Por supuesto, tal simplificación no hace justicia ni con mucho a la realidad sobre cómo se originaron realmente las ideas, sino que sólamente da una noción sobre la forma en que surgieron los diferentes paradigmas.

3.2.2.1 GRAMATICA PRESCRIPTIVA: LA LINGÜISTICA COMO LEY. -

El lenguaje es una forma del comportamiento social del ser humano, y como tal ha sido objeto de legislación y control a través de la historia. Esta metáfora es la perspectiva predominante sobre la estructura del lenguaje en nuestra sociedad. En las escuelas se enseña la gramática no como una forma de entender la estructura de un fenómeno, sino como un conjunto de reglas que los alumnos deben seguir para expresarse con propiedad para poder conseguir un lugar propio dentro de la sociedad.

El objetivo principal de la gramática prescriptiva es la "corrección" o la "pureza" del lenguaje. Tal como sucede con cualquier estructura legal, aquellos que viven dentro de ella están de acuerdo en la forma correcta de hacer las cosas, y desaprueban lo incorrecto. El trabajo del lingüista es entonces vigilar (como un policía) el cumplimiento cabal de las reglas del "buen decir".

Todas las teorías lingüísticas actuales rechazan esta metáfora. Un lenguaje es un fenómeno natural sujeto a cambios naturales con el tiempo; se define por lo que la gente dice en la realidad, no por lo que debería decir. El trabajo de un lingüista es el de tratar de entender la estructura del lenguaje, y la manera en que la alcanzó. De ninguna manera debe tratar de impedir su evolución, con el fin de que las convenciones de un grupo social se impongan

en el resto de la sociedad.

Existen, sin embargo, algunas razones por las cuales conviene enseñar una gramática "correcta" a las personas, como lo es el hecho de permitirles entrar en la estructura social (de otra manera serían rechazados por ella). Además, el trabajo de instituciones como la Real Academia de la Lengua no puede ser calificado tan a la ligera debido a que, si bien pretende unificar el uso de la lengua de tal manera que nos podamos comunicar, también permite que las palabras que logran tener un uso suficientemente frecuente, sean adoptadas por la lengua (el español en este caso); de esta forma se permite que la lengua evolucione con el uso.

3.2.2.2 LINGUISTICA COMPARATIVA: LA LINGUISTICA COMO BIOLOGIA. -

En el siglo XIX emergió un paradigma en el campo de la Lingüística denominado "filología comparativa". Su ocupación principal era descubrir las relaciones existentes entre los diferentes lenguajes y la forma en que evolucionaron hasta ser lo que son. La teoría de la evolución de Darwin estaba provocando cambios drásticos en la forma en que la visión del mundo prevaleciente en la época.

Los lingüistas comenzaron a poner mayor atención a las similitudes entre los lenguajes, particularmente en el vocabulario y en los patrones sonoros. Notaron que podían construir una especie de árbol genealógico de lenguas mediante la comparación de sus estructuras, postulando estructuras ancestrales en lenguajes que ya no existían, tal como los biólogos desarrollaban taxonomías de organismos que podían explicar las características que encontraban.

La mayoría del éxito obtenido en este paradigma fué parecido al de la historia natural. Tal como en los estudios contemporáneos de la evolución biológica, se podían desarrollar sólamente algunos principios que explicaban vagamente por qué las cosas habían evolucionado de la forma en que lo hicieron. Había que realizar la gigantesca tarea de clasificar y estructurar los datos existentes para formar un árbol filogenético perfecto.

Se postularon lenguajes como el "Proto-indo-europeo", como ancestros comunes desaparecidos hace mucho tiempo. Conforme se progresaba en este estudio, un lingüista podía encontrar un reto en un lenguaje (o palabra, o sonido específicos) que no había sido ubicado todavía en la taxonomía, y quedaba satisfecho si su estructura resultaba ser derivable de lenguajes ancestrales que ya habían sido postulados, o si los ancestros necesarios para explicarla resultaban poder ser empleados para explicar otros eslabones

perdidos.

Este tipo de actitud para solucionar enigmas es común a todos los periodos de ciencia "normal", y su éxito provee la motivación de los científicos para seguir trabajando dentro de un paradigma particular.

La sensación de que las piezas se acomodan de manera natural les proporciona la certeza de que la ciencia está en el camino correcto, y de que se están realizando progresos importantes.

Conforme los filólogos empezaron a agotar el conjunto de lenguas conocidas, se avocaron al estudio de lenguas más remotas reportadas por los antropólogos, pero su entusiasmo decayó cuando lo que habia que realizar era sólo una tediosa tarea de clasificación.

3.2.2.3 LINGUISTICA ESTRUCTURAL: LA LINGUISTICA COMO QUIMICA. -

La revolución que dejó a un lado la filología comparativa como el centro de los estudios lingüisticos se basó en un cambio de enfoque de un grupo de lenguas a una sóla lengua. Los problemas concernientes a los cambios del lenguaje se dejaron a un lado, y los lingüistas empezaron a preocuparse por describir las regularidades que encontraron en las expresiones de un lenguaje individual.

Un lingüista suizo llamado Ferdinand deSaussure fué uno de los primeros devotos del enfoque estructural, pero el mayor desarrollo del mismo se llevó a cabo en los Estados Unidos, empezando con la publicación "Language" de Leonard Bloomfield en 1933 [BL033]. Este paradigma mantuvo su predominancia en los 50's y todavía es seguido en muchos departamentos universitarios de lingüística.

La lingüística estructural fué influida fuertemente por los principios del conductismo: un paradigma que dominó la psicología americana durante el mismo período. Los conductistas sentían que la mayoría del trabajo previo en psicología no era científico, por que describía el comportamiento humano en función de procesos mentales que no se podían observar por experiencia científica objetiva; afirmaban que una verdadera ciencia psicológica se debe basar estrictamente en observaciones de la conducta. En la lingüística, ésto implicaba que los objetos apropiados de estudio eran cuerpos observables de comportamiento lingüístico. El material de trabajo apropiado para un lingüista era entonces un corpus recolectado de expresiones lingüísticas naturales.

Un químico realiza experimentos para determinar el conjunto de moléculas que forman una sustancia compleja, y luego analiza esas moléculas en función de sus elementos básicos. El gran éxito de la química consistió en encontrar

un grupo reducido de elementos primitivos cuyas combinaciones podían producir el gran número de sustancias diferentes existentes en la naturaleza. El lenguaje, con sus oraciones hechas de palabras, formadas por sonidos, fué sometido al mismo tipo de análisis.

La analogía con la química es muy cercana a la forma en que se organizan los sonidos en las palabras. Cada lenguaje tiene un pequeño grupo de categorías sonoras diferentes (entre 20 y 50), llamadas fonemas, y éstos se combinan en el habla con el fin de formar palabras.

En el estudio de la sintaxis -la forma en que se combinan las palabras en expresiones gramaticales- se usaron los mismos métodos, pero con menos éxito. El conjunto de elementos de significado (morfemas) de un lenguaje es mucho mayor y está menos estructurado que el de los fonemas. Los lingüistas eran tan incapaces de catalogar las diferentes clases de estructuras sintácticas de un lenguaje como los químicos de hacer poco más que catalogar la multitud de compuestos con estructura molecular grande. Muchos lenguajes diferentes fueron descritos en función de las estructuras que aparecen en ellos y se hizo énfasis en tratar de analizar un rango tan grande como fuera posible de lenguajes diferentes en el mundo.

3.2.2.4 LINGUISTICA GENERATIVA: LA LINGUISTICA COMO MATEMATICA. -

Un nuevo paradigma ha tenido preponderancia en los últimos veinte años debido en gran parte al trabajo de Noam Chomsky y sus estudiantes, dando comienzo con su libro "Syntactic Structures" publicado en 1957 [CHO69]. Este paradigma refuta la metodología empírica de la lingüística estructural y propone un cambio de enfoque: del análisis de la conducta observable, a las intuiciones de los hablantes natos acerca de su lengua.

Chomsky alegaba que el análisis estructural de textos o colecciones de expresiones no podía captar la creatividad escencial del lenguaje humano. Las expresiones no aparecen simplemente en la naturaleza, sino como resultado de capacidades mentales del hablante. El dominio apropiado del estudio, de acuerdo con el paradigma generativo, no es el de las oraciones en sí, sino el de la facultad subyacente que nos permite crearlas y entenderlas.

El problema central es caracterizar la gramática de un lenguaje, que ahora es el conocimiento tácito que emplea el comunicador. Esto requiere una explicación de las intuiciones de la gente sobre si una secuencia de palabras (o sonidos) es o no una oración válida en el lenguaje, y sobre las relaciones estructurales (tales como la paráfrasis) entre oraciones diferentes. Además, en vez de

seguir procedimientos rigurosos para analizar un lenguaje, un lingüista que trabaja dentro de este paradigma emplea intuiciones basadas en sus habilidades lingüísticas natas.

Chomsky distinguió entre "performance" (el proceso que realmente determina qué dirá un hablante o cómo se entenderá una expresión en un contexto particular) y "competence" (una caracterización abstracta del conocimiento lingüístico del hablante).

El concepto de "competence" se asemeja en gran medida con la noción de una comprobación matemática. Podemos pensar que la matemática es un lenguaje de fórmulas, y que el trabajo de un matemático es explicar qué combinaciones de símbolos representan proposiciones verdaderas, dado un conjunto de axiomas y reglas de inferencia. La expresión "(x + 1)^2 = x^2 + 2x + 1" es una oración verdadera del álgebra ordinaria, mientras que otra secuencia hecha con los mismos símbolos, tal como "(x + 2)^2 = x^2 + 4x + 2" no lo es. La Matemática no es el estudio de cómo llegan las personas a inventar esas expresiones o qué pasa en sus mentes cuando las leen o tratan de comprobarlas; su meta es producir un conjunto de reglas y mecanismos formales que determinan de manera precisa cuáles son verdaderas. La medida del éxito de una formalización de reglas y operaciones para un área de la Matemática estriba en su elegancia y economía.

La lingüística generativa aprecia el lenguaje como un objeto matemático y construye teorías que se parecen mucho a los conjuntos de axiomas y reglas de inferencia de la Matemática. Una oración es gramatical si hay alguna derivación que demuestra que su estructura está de acuerdo con el conjunto de reglas, tal como una comprobación la veracidad de una oración matemática. Una analogía útil es recordar que la Matemática formal provee una forma precisa de reconocer una comprobación válida, sin proveer la manera de describir cómo la genera un matemático.

3.2.2.5 EL PARADIGMA COMPUTACIONAL.-

La computadora comparte con la mente humana su capacidad de manipular símbolos y llevar a cabo procesos complejos que incluyen hacer desiciones en base a conocimientos almacenados. A diferencia de la mente humana, los procesos computacionales están completamente abiertos a su inspección y estudio, y podemos experimentar construyendo programas y bases de conocimiento a nuestro antojo. Los conceptos teóricos de algoritmo y estructuras de datos pueden formar las bases para construir modelos computacionales precisos de los procesos mentales. Podemos tratar de explicar las regularidades entre estructuras lingüísticas como consecuencia de las operaciones subyacentes.

En la actualidad, la computadora se emplea de dos maneras en relación con el lenguaje: como una herramienta de análisis y como una máquina que puede simular el entendimiento del lenguaje.

Cuando se dice que la computadora es empleada como una herramienta para el estudio del lenguaje, el campo denominado **lingüística computacional**, del que Chomsky es uno de los pioneros, es el referido.

Por otro lado, al apreciar una computadora como una **máquina que puede simular el entendimiento del lenguaje, se habla del área de la Inteligencia Artificial denominada procesamiento del lenguaje natural**; dentro de esta área, se desarrollan los programas mencionados en el capítulo anterior como sistemas de comprensión amplia y sistemas de comprensión limitada del lenguaje natural.

El campo de acción de este trabajo es el segundo, ya que el primero es ocupación principalmente de los lingüistas. A continuación se da una breve explicación de algunas de las estructuras del lenguaje, para continuar con la explicación de uno de los modelos de representación gramatical de mayor uso en la actualidad: las redes de transición aumentada.

de derivación. De hecho, la descripción de la gramática del español se parece mucho a una gramática libre de contexto; sin embargo, no hay que olvidar que el lenguaje natural no es un lenguaje descriptible por estos medios, ya que es necesario emplear extensiones que permitan que la descripción contemple las relaciones contextuales necesarias en todo lenguaje natural.

Una regla de derivación se escribe de la siguiente manera:

A -> B

Y se lee de la siguiente forma:

"B es una derivación de A", o bien "B se deriva de A".

Lo anterior significa que B es una forma de "extender" o "reescribir" A, por lo que se llaman también reglas de reescritura.

Cada una de las reglas que describen a la oración simple del español podrá ser enunciada explicando cada una de las derivaciones implicadas por ellas, y de esta manera se logrará entender la forma como se estructura la oración simple a partir de partículas (palabras) incluídas en grupos funcionales que todos conocemos (adjetivo, sustantivo, artículo, etc.).

La primera regla es la que define la oración, y es la siguiente:

1. ORACION -> SNOM SVERB

Donde:

SNOM es un sintagma (grupo) nominal.

SVERB es un sintagma verbal.

Como se puede apreciar, aquí todavía no se distinguen los dos papeles de los constituyentes de la oración: sujeto y predicado. Esto se debe a que a este nivel no se puede definir el papel del grupo nominal que antecede al verbo. Puede ser, por ejemplo, que se esté analizando una oración en voz activa, como la siguiente:

"Pedro corrió las cortinas."

y en tal caso el sujeto sería el sintagma nominal que precede al verbo. Sin embargo, puede darse el caso de que se trate de una oración en voz pasiva, como:

"Las cortinas fueron corridas por Pedro."

y en este caso el sujeto es el introducido por la preposición "por", y el sintagama nominal que antecede al verbo viene siendo el objeto direto (el que recibe la acción). Es por ello que a este nivel es imposible predecir el papel que juega el sintagma nominal en la oración.

La segunda regla de derivación se encarga de describir el sintagma verbal de la siguiente forma:

2. SVERB -> (AUX) VERBAL (CIRC)

Donde:

AUX es un auxiliar.

VERBAL es la frase verbal en sí.

CIRC es un complemento circunstancial.

El hecho de que un símbolo esté encerrado entre paréntesis indica que dicho símbolo puede o no ser considerado en la definición. Concretamente, en el sintagma verbal puede o no existir un grupo auxiliar, y lo mismo sucede con el complemento circunstancial. En otras palabras, los paréntesis indican opcionalidad.

El sintagma verbal describe -en voz activa- lo que en la gramática estructural se denomina predicado. Está compuesto por un número arbitrario de auxiliares seguidos de una frase verbal "VERBAL".

La frase verbal puede estar seguida por uno o más complementos circunstanciales "CIRC", que como su nombre lo indica, expresan las circunstancias de modo, tiempo y lugar en que se realiza la acción verbal.

La siguiente regla define los auxiliares empleados en la frase verbal:

3. AUX -> verbo modal (AUX)

Donde:

verbo modal es un verbo como ser, estar o haber.

Los verbos modales son los siguientes:

1. **HABER**, antepuesto al participio, forma los tiempos compuestos de la voz activa de todos los verbos; juntos constituyen un verbo (en perífrasis). Ejemplo: "Yo he comprado un coche".

2. **SER Y ESTAR**, antepuestos al participio de un verbo, forman los tiempos (todos compuestos) de la voz pasiva perifrástica. Ejemplo: "Todo fué grabado perfectamente" o "Las cortinas fueron corridas por Pedro".

La siguiente regla define la **frase verbal** empleada en la regla 2 (VERBAL):

4. VERBAL -> verbo (SNOM) |
cop PREDNOM |
VERBAL (GPREP) (GPREP)

Donde:

verbo Es un verbo cualquiera.

cop Es un verbo copulativo ("ser" por ejemplo).

GPREP Es un sintagma nominal antecedido por una preposición.

En esta regla aparece un nuevo símbolo: la barra vertical "|" indica opcionalidad, o sea que puede tomarse la primera derivación, o la segunda, o la tercera, etc.

Esta regla trata los componenetes verbales, que pueden ser cualquiera de los siguientes:

- Un OBJETO DIRECTO, que es un sintagma nominal que indica el objeto que recibe la acción. Está expresado en la primera parte de la regla.
- Un PREDICADO NOMINAL, que es un sintagma nominal o un adjetivo que sigue a un verbo copulativo, refiriéndose al sujeto. Concuerda con el sujeto (en género y en número) y con el verbo (en número y persona). Está expresado en la segunda opción de la regla, y su regla

de derivación es la siguiente:

5. PREDNOM -> adj | SNOM | CIRC

- **Un OBJETO DE INTERES, que incluye los objetos indirectos y los objetos dativos de interés se encuentra ligado al verbo generalmente mediante la preposición "a": "Le di una rosa a Lulú".**

  Esta construcción está expresada por el primer GPREP de la tercera opción de la regla 4.

- **Un COMPLEMENTO VERBAL, que se encuentra en estrecha relación con el verbo principal, como en: "Sueño con mi tesis".**

  Esta estructura está representada por el segundo GPREP de la tercera opción de la regla 4.

La tercera parte de la regla 2 indica otro de los complementos más usuales del verbo: los **complementos circunstanciales**. Estos complementos tienen la siguiente regla de derivación:

6. CIRC -> (adv) adv CIRC |
GPREP (CIRC) |
SNOM (CIRC)

Donde:

adv Es un adverbio.

Los circunstanciales están normalmente compuestos por adverbios de tiempo, modo o lugar, como se expresa en la primera opción de la regla.

Otra forma de calificar la acción verbal es mediante la inclusión de grupos preposicionales, como se expresa en la segunda opción de la regla.

Los sintagmas nominales incluídos en la tercera parte de la regla, se indican para los casos particualres en que la acción verbal sea calificada por un grupo nominal.

La recursividad de la regla refleja la particularidad de que un verbo puede estar calificado por un número arbitrario de circunstanciales.

En seguida se tratan los **sintagmas nominales**, que se encuentran en la mayoría de las reglas. La regla que indica su derivación es la siguiente:

7. SNOM -> (DET) sust

Donde:

DET	**Es un determinante del sustantivo.**
sust	**Es el sustantivo principal del sintagma.**

En esta regla se refleja la característica de los grupos nominales de estar compuestos por un sustantivo principal calificado por un determinante.

Finalmente, las reglas que definen las derivaciones posibles del **determinante** son las siguientes:

8.	DET	->	INDEF (NUM) | DEF (otr-) | DEF (NUM) | calif (otr-)
9.	INDEF	->	art-ind | (otr-)
10.	DEF	->	(FRAC) pos | (FRAC) dem | (FRAC) art-det
11.	NUM	->	card | ord
12.	FRAC	->	tod- | SNOM "de"

Donde:

DET	**Es un determinante.**
INDEF	**Significa indefinido.**
DEF	**Significa definido.**
NUM	**Significa numeral.**
otr-	Significa "otro", "otra", etc.
calif	Significa calificativo.
art-ind	Es un articulo indeterminado.
FRAC	Es un indicador de fracción.
pos	Es un adjetivo posesivo.
dem	Significa adjetivo demostrativo.
art-det	Es un articulo determinado.
card	Significa cardinal.
ord	Significa ordinal.
tod-	Significa "todo", "todos", etc.

El uso de la palabra determinante para designar esta categoría se justifica si se considera que su función es la de determinar al sustantivo que les sigue. La primera de estas reglas define los tipos de determinantes que existen: indefinido, definido y calificativo.

El determinante indefinido tiene dos clases principales: el articulo indefinido (un, una, etc.) y otr- (otro, otra, otras, etc.) como se indica en la regla 9 y puede ser seguido por un numeral. Este tipo de determinante se emplea cuando la determinación del sustantivo no es definida.

Los determinantes definidos sirven para especificar cuál de todos los elementos posiblemente referenciados es el que la oración implica. En la regla 10 se establece que los determinantes definidos pueden ser posesivos, demostrativos o articulos determinados. Pueden ir seguidos de otr- o un

numeral. Además pueden ir precedidos de un elemento que indique una fracción.

Los determinantes calificativos son aquellos que funcionan como intensificadores de adjetivos (mucho, poco, demasiado, ninguno, tal, cierto, etc.) así como indicadores de fracciones (frac).

Los numerales, definidos en la regla 11, pueden ser cardinales (uno, dos, etc.) u ordinales (primero, segundo, etc.).

Finalmente (regla 12) se definen las fracciones (FRAC), que sirven para indicar la cantidad o fracción del grupo indicado por el sustantivo que les sigue a los determinantes.

Como se aclaró en un principio, las reglas explicadas son las reglas de estructura sintagmática de la gramática transformativa del español. Resulta claro que es necesario implicar más elementos contextuales para definir las reglas que rigen la inclusión de una palabra determinada en una estructura definida por las reglas anteriores; estas reglas también son parte de la gramática transformativa, y reciben el nombre de reglas de inserción léxica. En su definición se emplean elementos tales como son las matrices de rasgos, y reglas de subcategorización. Además, con el fin de

permitir la identificación de estructuras ambiguas, existen reglas que permiten verificar la concordancia semántica (semántica transformativa), adicionalmente a las reglas que definen las relaciones sintácticas permisibles, que permiten a la gramática transformativa determinar la concordancia (de género, número y persona) entre los elementos de la oración. En el presente capitulo no se incluyen debido a que se consideró que no son necesarias para los fines de esta exposición. Sin embargo, si se desea profundizar en el tema, se recomienda ampliamente la lectura de "Gramática Transformativa del Español" [HAD71] como la más apropiada.

3.4 REPRESENTACIONES DE INFORMACION GRAMATICAL.

A continuación se explica uno de los modelos más empleados para la representación de la información gramatical. Este formalismo se basó en los conceptos desarrollados en teoría de compiladores; es por ello que el presente apartado se ocupa de algunos métodos de representación gramatical empleados para el análisis sintáctico de lenguajes formales, con el fin de familiarizar al lector con los conceptos manejados en el desarrollo de las redes de transición aumentada.

Al mismo tiempo que emergía la gramática generativa, se realizaron una serie de desarrollos en el campo de la Inteligencia Artificial que seguían un enfoque diferente: en vez de tomar el proceso como iniciado en una derivación abstracta, se encaminaron a desarrollar el proceso de análisis sintáctico, como se entiende en teoría de compiladores. A los sistemas desarrollados para compilar lenguajes formales de programación se les fueron agregando mecanismos cada vez más sofisiticados con el fin de que pudieran manejar sintaxis cada vez más complejas, del tipo de las que describen los lenguajes naturales.

Surgió entonces un gran número de sistemas de esta clase, y el formalismo más claro de los mismos se realizó en función de las redes de transición aumentadas; los descendientes de las redes de transición aumentada son

actualmente los métodos más difundidos de análisis sintáctico en sistemas automatizados de comprensión del lenguaje natural. Estos formalismos sirven además como base para el desarrollo de algunas teorías psicolingüísticas actuales.

A continuación se describe un formalismo empleado para el reconocimiento de cadenas de entrada (cuerdas de símbolos), con el fin de introducir gradualmente los conceptos básicos de las redes de transición aumentadas.

3.4.1 LOS AUTOMATAS DE ESTADOS FINITOS. -

Un autómata de estados finitos es un modelo de reconocimiento de estructuras consistente en:

- Un conjunto de estados, donde uno es un estado distinguible llamado estado inicial, y un grupo de ellos se denominan estados finales o de aceptación.

- Un conjunto de arcos (segmentos dirigidos) que unen los estados entre sí. Los arcos representan "transiciones" del autómata -cambios de un estado inicial a otro estado cualquiera-.

En la siguiente figura se muestra un autómata de estados finitos cuyo alfabeto está constituido por ceros y unos. La cadena que reconoce el autómata es la formada por

un número impar cualquiera de unos:

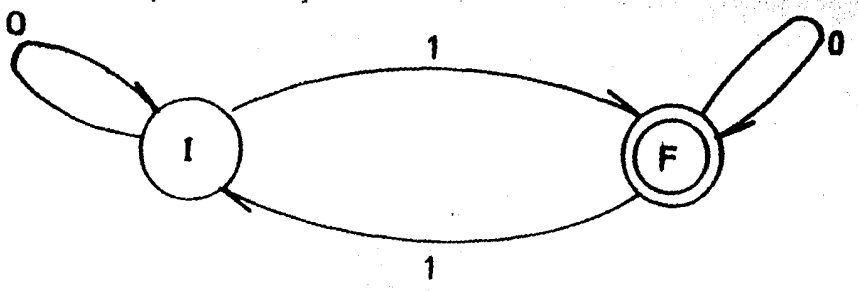


FIGURA 3.1.- Un autómata de estados finitos.

El estado inicial está representado por el estado I y el estado final es el estado F. Como se puede observar en la figura, los arcos tienen "etiquetas". para poder realizar la transición indicada por un arco, la entrada actual del autómata tienen que ser igual al símbolo que lo etiqueta. Cuando se ha agotado la entrada y el autómata se encuentra en un estado final, se ha reconocido la validez de la cadenaa de entrada; ésta es la razón por la que el estado final también se llama estado de aceptación.

Veamos el siguiente autómata de estados finitos:

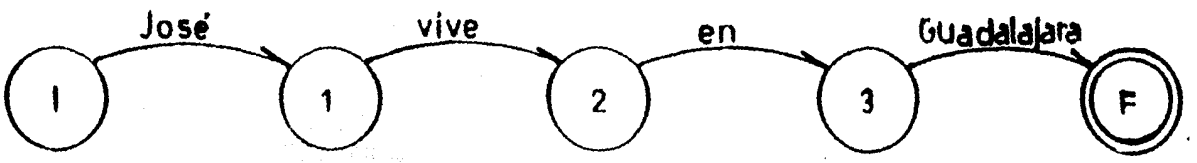


FIGURA 3.2.- Autómata que reconoce sólo una cadena de entrada.

El autómata anterior sólo reconoce la oración "José vive en Guadalajara". Es evidente que este tipo de reconocedores resulta muy fácil de construir cuando el repertorio de frases que vamos a manejar es muy reducido. Cuenta además con la ventaja de que resulta muy sencillo conocer el alcance del autómata: todo lo que ha de hacerse es seguir los arcos que unen el estado inicial con los estados de aceptación.

Un autómata de estados finitos sólo puede tener símbolos terminales en sus arcos, lo que lo hace incapaz de manejar problemas comunes en los lenguajes naturales, como son la recursividad, o el manejo de sus regularidades.

Se llega entonces a la conclusión de que un autómata de estados finitos no cuenta con las características necesarias para manejar un lenguaje natural.

En el intento por aumentar su poder expresivo, surgió un modelo que supera algunas de sus limitaciones. A continuación lo exponemos.

3.4.2 LAS REDES DE TRANSICION RECURSIVA. -

Una red de transición recursiva contiene los mismos elementos de un autómata de estados finitos: un conjunto de estados, uno de los cuales es el estado inicial, y un grupo de ellos, que se conocen como estados de aceptación; también

entre los estados del modelo. La gran diferencia estriba en los simbolos que se pueden emplear para etiquetar un arco: además de poder tener elementos terminales como etiquetas de los arcos, el modelo incorpora la posibilidad de emplear símbolos no terminales como etiquetas. Lo anterior permite que un arco tenga como etiqueta el nombre de un estado que será el estado inicial de otra subred, provocando el siguiente efecto al realizarse la transición:

1. El nombre del estado al final del arco es almacenado en la parte superior de una pila (stack).

2. El control se transfiere (sin avanzar la entrada) al estado cuyo nombre etiqueta el arco.

3. Cuando se alcanza un estado final, se saca el contenido del tope de la pila.

4. El control se transfiere al estado cuyo nombre es el elemento sacado de la pila.

Es por lo anterior que un estada de aceptación se alcanza cuando, estando en el nodo de aceptación, se intenta sacar el tope de la pila y ésta se encuentra vacía, habiendo agotado además la cadena de entrada.

Los nombres de los estados que se usan como etiquetas para los arcos corresponden normalmente a nombres de construcciones que puedan ser encontradas como "frases" de la expresión de entrada. La transición representada por un arco equivale entonces al reconocimiento de está frase en particular.

Este modelo tiene su semejante en compiladores, y recibe el nombre de autómata "Push-down", debido a sus características recursivas. Precisamente debido a que es recursivo permite capturar las relaciones de subordinación y coordinación de oraciones. Existen, además, regularidades del lenguaje que pueden ser representadas por subredes, eliminando la necesidad de tener varias copias exactas en diferentes lugares.

Se puede, entonces, emplear redes principales y subredes; un ejemplo podría ser aquél en que la red principal reconozca la oración, en términos de sintagma nominal y sintagma verbal, como se aprecia en la siguiente figura:

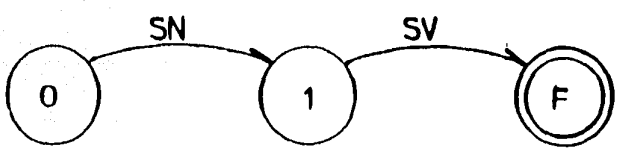


FIGURA 3.3.- Red de transición recursiva que reconoce una oración.

La anterior sería la red principal. Las subredes estarían compuestas de la siguiente manera:

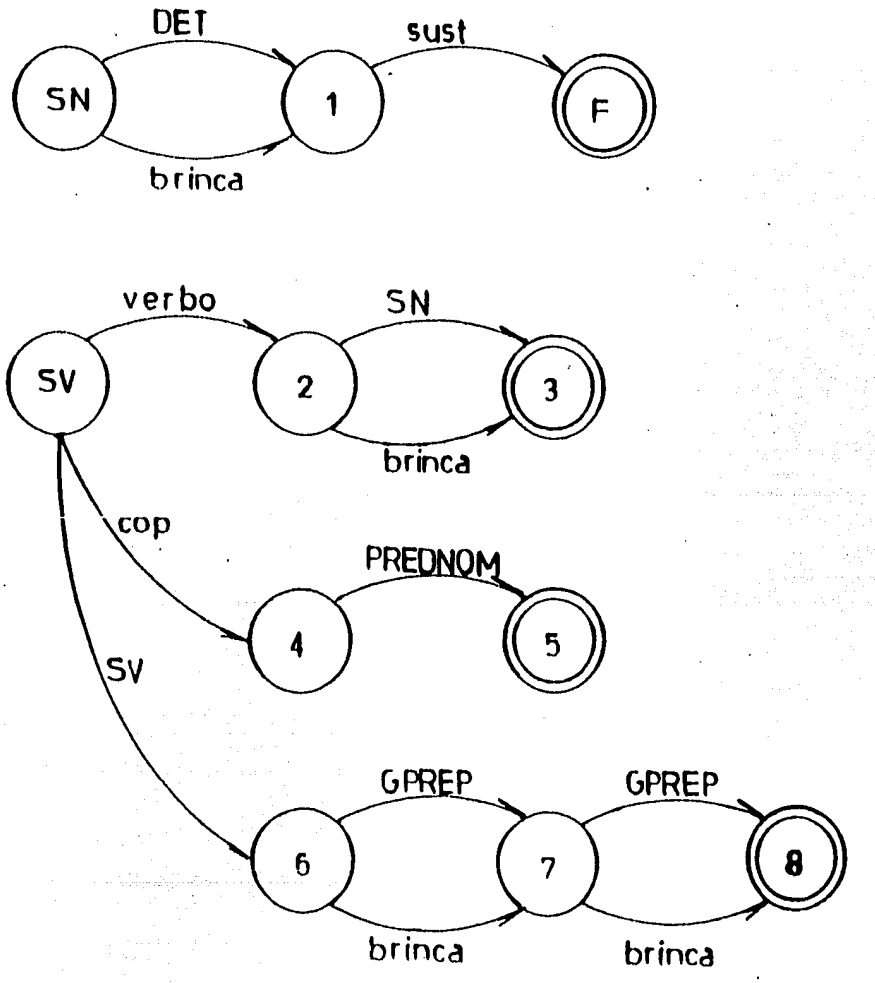


FIGURA 3.4.- Subredes para reconocer una oración.

Se puede apreciar un nuevo tipo de arco: el arco etiquetado por "BRINCA". Este arco no requiere que se realice ninguna acción para recorrerlo; no se debe avanzar en la cadena de entrada, y se emplea para denotar

opcionalidad.

Del conjunto de redes expresado arriba se pueden deducir las siguientes reglas para la gramática que acepta:

```
ORACION     ->    SN SV
SN          ->    (DET) sust
SV          ->    verbo (SN) |
                  cop PREDNOM |
                  VERBAL (GPREP) (GPREP)
```

Los arcos etiquetados con palabras en minúsculas indican símbolos terminales de la gramática, mientras que los etiquetados por mayúsculas indican símbolos no terminales, que señalan la existencia de un estado inicial llamado de esa manera (en la gramática sería la derivación en la cual el símbolo no terminal se encuentra a la izquierda del signo de derivación "->"). Para profundizar en los conceptos de definición de la gramática se recomienda recurrir a [LEW78, AHO77].

La gramática anterior recibe el nombre de gramática libre de contexto, debido a que es incapaz de capturar la influencia que ejerce la presencia de unos u otros componentes contextuales de la entrada sobre la parte de la entrada que se está analizando. Esta gramática será incapaz, por ejemplo, de verificar la concordancia en género y número que debe existir entre un sustantivo y el adjetivo

que lo califica, provocando la aceptación de las siguientes oraciones como válidas:

"La mujer altos se desmayó." ó

"José Luis es muy trabajadores."

en vez de:

"La mujer alta se desmayó." y

"José Luis es trabajador."

Otra de las características que definen a una gramática libre de contexto es que los símbolos terminales a la izquierda de las reglas de derivación se encuentran completamente despejados.

Debido a que las gramáticas libres de contexto -y por lo tanto las redes de transición recursiva- son incapaces de representar relaciones elementales entre el contexto y la cadena de entrada, surgió la necesidad de realizar nuevamente algunas extensiones a este modelo, lo que dió origen a las redes de transición aumentadas (ATN's, siglas de "Augmented Transition Networks").

3.4.3 LAS REDES DE TRANSICION AUMENTADAS. -

Debido a que resulta necesario aumentar el modelo de las redes de transición recursiva, surgió la idea de realizar pruebas arbitrarias en algunos arcos de la red, condicionando entonces la transición a ciertas restricciones planteadas por el diseñador. Además se pensó que era necesario realizar un conjunto de operaciones sobre la cadena de entrada, en vez de sólo cambiar de estado al recorrer un arco.

Realizar un conjunto de operaciones para reorganizar los elementos de la cadena de entrada es un mecanismo muy útil para lograr encontrar las relaciones entre los diferentes elementos de una oración. Asimismo, como se verá más tarde, permite mantener un conjunto de registros que servirán para construir un arbol sintáctico que explicite la relación entre una oracion afirmativa y su correspondiente forma negada o interrogativa, dándole al modelo el poder expresivo de una gramática transformacional.

Lo anterior implica que las redes de transición aumentada son básicamente unas redes de transición recursiva con los siguientes extensiones:

- Definición de condiciones arbitrarias, que deben ser satisfechas para poder seguir el arco que las contenga.

- **Implantación de conjuntos de acciones de construcción de estructuras, que se ejecutan en el momento en que se sigue el arco asociado.**

Según Woods:

> "Las características principales que una gramática transformacional agrega a las gramáticas libres de contexto, son la capacidad de mover fragmentos de la estructura de la oración (para que sus posiciones en la estructura de salida sean diferentes a las que ocupan en la estructura superficial), copiar y borrar fragmentos de estructura, y realizar acciones sobre partes que generalmente dependen del contexto en que ocurren." [W0070]

Una red de transicion aumentada construye una descripción estructural parcial de una oración conforme pasa de un estado a otro por la red. Las piezas de la descripción parcial se almacenan en registros que pueden contener cualquier árbol o lista de árboles que son automáticamente almacenados ("pushed down") en el momento en que se realiza el llamado de una aplicación recursiva de la red de transición, y reestablecidos ("restored") cuando se completa la operación de bajo nivel.

Las acciones de construcción de estructura sobre los arcos especifican cambios en el contenido de los registros, en función de su contenido previo, el contenido de otros registros, el símbolo actual de entrada y/o el resultado de operaciones de niveles inferiores. Además de servir para contener partes de estructura que serán eventualmente

empleadas para construir la estructura de salida, los registros pueden ser empleados para almacenar banderas y otros indicadores que podrán ser empleados en las condiciones de otros arcos con el fin de tomar decisiones que, en este momento, ya dependerán del contexto.

Cada estado final de la red aumentada tiene asociadas una o más condiciones a satisfacer para que tal estado cause la salida a un nivel superior de operación (llamado "pop"). Cada una de estas condiciones es una función que averigua el valor a ser entregado por la operación apenas realizada. El resultado de la operación de bajo nivel es almacenado en un registro especial -llamémoslo el registro *- que normalmente contiene la entrada actual al analizar una palabra. Esto implica que el registro * siempre contendrá una representación del "objeto" (palabra o frase) que causó la transición apenas realizada.

La siguiente forma de representación de las redes de transición aumentadas se basa en la presentación original de las mismas que realizó Woods en "Transition network grammars for natural language analysis" [WOO70].

3.4.3.1 DESCRIPCION DE LAS REDES DE TRANSICION AUMENTADAS. -

Para concretar la discusión sobre las redes de transición aumentada, se hará una especificación de las mismas en notación BNF:

```
⟨red de transición⟩  ->  ((⟨grupo de arcos⟩
                           ⟨grupo de arcos⟩^)

⟨grupo de arcos⟩     ->  ((⟨estado⟩ ⟨arco⟩^)

⟨arco⟩               ->  (CAT ⟨nombre de categoria⟩
                              ⟨prueba⟩ ⟨acción⟩^
                              ⟨acción terminal⟩) |
                         (PUSH ⟨estado⟩ ⟨prueba⟩ ⟨acción⟩^
                              ⟨acción terminal⟩) |
                         (TST  ⟨prueba arbitraria⟩ ⟨prueba⟩
                              ⟨acción⟩^
                              ⟨acción terminal⟩) |
                         (POP ⟨forma⟩ ⟨prueba⟩)

⟨acción⟩             ->  (SETR ⟨registro⟩ ⟨forma⟩) |
                         (SENDR ⟨registro⟩ ⟨forma⟩) |
                         (LIFTR ⟨registro⟩ ⟨forma⟩)

⟨acción terminal⟩    ->  (HACIA ⟨estado⟩) |
                         (BRINCA ⟨estado⟩)

⟨forma⟩              ->  (GETR ⟨registro⟩) |
                         * |
                         (GETF ⟨característica⟩) |
                         (BUILDQ ⟨fragmento⟩ ⟨registro⟩^) |
                         (LIST ⟨forma⟩^) |
                         (APPEND ⟨forma⟩ ⟨forma⟩) |
                         (QUOTE ⟨estructura cualquiera⟩)
```

FIGURA 3.5.-Gramática para especificar una red de transición aumentada.

La especificación anterior está expresada en forma de gramática libre de contexto, con algunos aumentos: la barra "|" separa formas alternas de derivación, el exponencial "^" indica elementos que se pueden repetir un número arbitrario de veces (operador estrella de Kleene). Los símbolos no terminales de la gramática están representados por descripciones en español encerradas entre paréntesis angulares "<>", y los demás son símbolos terminales (incluyendo los paréntesis derecho e izquierdo).

La primera regla de la gramática indica que una red de transición se representa mediante un paréntesis izquierdo, seguido de un grupo de arcos, seguido de un número cualquiera de grupos de arcos, y finalmente un paréntesis derecho.

Cada grupo de arcos, como lo indica la segunda regla está formado por un paréntesis izquierdo, un estado y un número cualquiera de arcos seguidos de un paréntesis derecho.

Un arco, cuya definición es la tercera regla de la gramática, puede ser una de las cuatro funciones indicadas en ella.

Las expresiones generadas por esta gramática para expresar la red de transición aumentada tienen forma de lista enmarcada entre paréntesis, de tal forma que la lista

de los elementos A, B, C y D, se representa por la expresión (A B C D). La red de transición se representa como una lista de grupos de arcos, cada cual es en sí una lista cuyo primer elemento es un nombre de estado, y cuyos elementos restantes son arcos que parten de esos estados. Los arcos son también representados como listas, cuyas formas se indican en la misma gramática.

El primer elemento de un arco es una palabra que define el tipo de arco, y el tercer elemento es una condición arbitraria que debe ser satisfecha para poder "atravesar" el arco. Los tipos de arcos son los siguientes:

- CAT.- Es un arco que se sigue sólo si el símbolo actual de entrada es miembro de la categoría léxica denominada en el arco (y si la condición se satisface).

- PUSH.- Es un arco que provoca una llamada recursiva al estado indicado.

- TST.- Es un arco que permite que se realice una prueba arbitraria para determinar si se puede seguir o no.

En cualquiera de estos arcos se realizan acciones de construcción de estructuras, y la acción final especifica el estado al que se transferirá el control como resultado de la transición. Las dos acciones terminales posibles son:

- **VACIA.- El control se transferirá al estado indicado, avanzando al mismo tiempo el apuntador de entrada.**

- **BRINCA.- El control se transferirá al estado indicado, sin avanzar el apuntador de entrada.**

Lo anterior implica que las dos acciones terminales sólo difieren en la determinación de si en el estado siguiente se trabajará con la palabra actual de entrada o con la siguiente.

El último tipo de arco pendiente es el siguiente:

- POP.- Es un arco "fantasma" que indica bajo qué condiciones el estado actual puede ser considerado un estado final, y la forma en que se entregará el resultado de la operación de bajo nivel, si se toma esta alternativa.

Las acciones y las funciones presentadas en la red se estructuran en notación polaca o prefija: una notación en que una llamada funcional se indica como una lista entre paréntesis cuyo primer elemento es el nombre de la función, y cuyos elementos subsecuentes son sus argumentos.

Las tres acciones indicadas en la gramática descrita causan la actualización del registro indicado al valor entregado por la función indicada:

- SETR.- Provoca que lo anterior se realice en el nivel actual de operación.

- SENDR.- Provoca que ésto se haga al siguiente nivel inferior de inclusión (para mandar información a un nivel inferior de operación).

- LIFTR.- Provoca que la operación indicada se realice al siguiente nivel superior de inclusión (para entregar información a niveles superiores de operación).

Las funciones, así como las condiciones (pruebas) de la red de transición pueden ser funciones arbitrarias del contenido de los registros, representados en algún lenguaje de especificación funcional, tal como el LISP [WIN84], un lenguaje de programación de procesamiento de listas basado en el cálculo Lambda [GRA84]. Los siete tipos de funciones listadas son un conjunto básico que resulta suficiente para ilustrar las características principales del modelo:

- GETR.- Es una función cuyo valor es el contenido del registro indicado.

- *.- Es una forma que cuyo valor es normalmente la palabra actual de entrada. (En las acciones ocurridas en un arco "PUSH", * tiene el valor de la operación de nivel inferior que permitió la transición).

- CAT.- Es una función que determina el valor de una característica especificada de la palabra actual de entrada.

- BUILDQ.- Es una función de construcción de estructuras que toma una lista que representa un fragmento de un árbol sintáctico con nodos marcados especialmente y entrega como su valor el resultado de sustituir esos nodos especialmente marcados con el contenido de los registros indicados como sus argumentos subsecuentes.

- LIST.- Es una expresión que construye una lista con sus argumentos.

- APPEND.- Es una forma que une dos listas para formar una sola.

- QUOTE.- Produce como su valor la forma que es su argumento (sin evaularse).

EJEMPLO.

Supongamos la siguiente porción de una red de transición aumentada:

```
(O/  (PUSH SNOM/T
          (SETR SUJETO *)
          (SETR TIPO (QUOTE DECLARATIVA))
          (HACIA Q1))
     (CAT INT T
          (SETR INT *)
          (SETR TIPO (QUOTE INTERROGATIVA))
```

```
        (HACIA Q2)))
(Q1  (CAT VERBO T
        (SETR INT NIL)
        (SETR V *)
        (HACIA Q4)))
(Q2  (PUSH SNOM/T
        (SETR SUJETO *)
        (HACIA Q3))
     (BRINCA Q3)))
(Q3  (CAT VERBO T
        (SETR V *)
        (HACIA Q4)))
(Q4  (POP
        (BUILDQ (ORACION + + + (PREDICADO +))
                TIPO SUJETO INT V)
        T)
     (PUSH SNOM/T
        (SETR PREDICADO
              (BUILDQ (PREDICADO (VERBO +) *) VERBO)
        (HACIA Q5)))
(Q5  (POP
        (BUILDQ (ORACION + + + +)
                TIPO SUJETO INT PREDICADO)
        T)
     (PUSH PREPOSICIONAL/T
        (SETR PREDICADO (APPEND (GETR PREDICADO) (LIST *)))
        (HACIA Q5)))
```

FIGURA 3.6.- Segmento de una red de transición.

En el segmento mostrado, los PUSH llaman rutinas de reconocimiento de las frases indicadas. En realidad, indican un cambio importante en la forma de trabajar: se "empujan" los contenidos de los registros un nivel hacia abajo ("PUSH") y se cambia a la porción de la red denominada por el estado inicial que se pasa como argumento a la función. Como se indicó anteriormente, el registro * contiene un subárbol sintáctico que se forma al realizar las operaciones implicadas por la subred llamada al iniciar la

transición. En la figura no se muestran las subredes. Simplemente se asume existencia, y su funcionamiento satisfactorio.

La porción de la red consta de los estados O/, Q1, Q2, Q3, Q4 y Q5. La red construye una representación estructural de la oración de entrada en que el primer elemento indica el tipo de oración (declarativa o interrogativa). El segundo elemento de la estructura es el sujeto. El tercer elemento es la palabra empleada para realizar la pregunta (Qué, Quién, Cuál, etc.), y el cuarto es el predicado de la oración.

El orden de esta representación no está influido por el orden de los elementos de la oración de entrada, lo que le da su potencia expresiva a las redes de transición aumentadas.

Antes de ralizar un ejemplo, se debe analizar de cerca el tipo de árboles que presenta a su salida esta red, y su representación. El árbol se expresa en forma de una lista entre paréntesis, cuyo primer elemento es el nombre del nodo padre, y sus siguientes elementos son sus árboles hijos. La definición recursiva de un árbol es la siguiente.

Un árbol es un nodo con árboles como hijos, o el conjunto vacio.

Por lo tanto, el árbol representado en la siguiente figura:

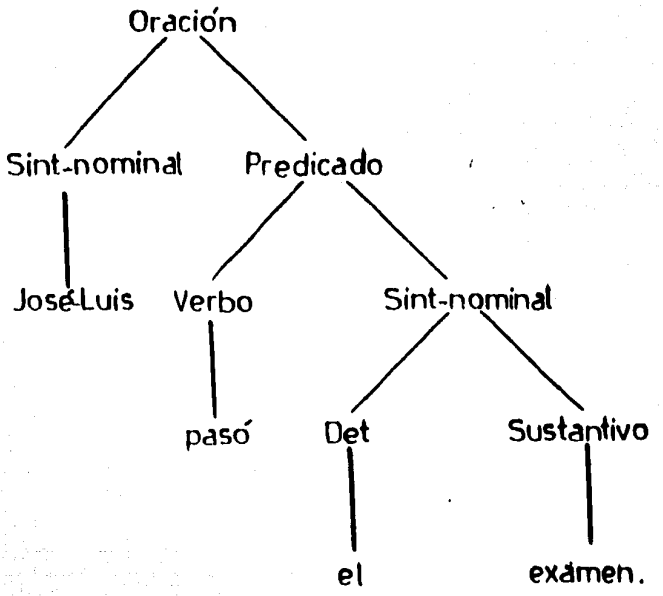


FIGURA 3.7.- El árbol sintáctico de la oración "José Luis pasó el examen".

estará representado en la siguiente expresión: (ORACION

(SINT-NOMINAL José-Luis)

(PREDICADO (VERBO pasó)

(SINT-NOMINAL (DET el)

(SUSTANTIVO examen)))

Consideremos ahora la oración:

"¿Quién pasó el examen?"

A continuación se desglosa el proceso de análisis:

1. El proceso se inicia en el estado 0/, con la primera palabra (Quién). Debido a que ésta palabra es un pronombre interrogativo, se siguen las operaciones indicadas en el arco etiquetado (CAT INT T...), en el siguiente orden:

ACCION	REGISTRO	VALOR
(SETR INT *)	INT	Quién
(SETR TIPO ...)	TIPO	INTERROGATIVA

Posteriormente, la acción (HACIA Q2) provoca que se transfiera el control al estado Q2 avanzando el apuntador de la entrada a la siguiente palabra (pasó).

2. En el estado Q2, el arco etiquetado (PUSH SNOM/T ...) resulta fallido, debido a que lo que sigue no es un sintagma nominal. Por lo tanto se sigue el arco (BRINCA Q3), que transfiere el control al estado Q3 sin que se

avance en la oración de entrada. •

3. En Q3 se verifica la condición (CAT VERBO T ...), debido a que la categoría léxica de "pasó" es VERBO. Se realizan las siguientes acciones:

ACCION	REGISTRO	VALOR
(SETR V ⋆)	V	pasó

Posteriormente se ejecuta (HACIA Q4), lo que provoca que el control se transfiera al nodo Q4, y el contenido del registro ⋆ sea "el".

4. En el estado Q4 no se puede seguir el arco POP, debido a que el registro ⋆ no se encuentra vacío. Por lo tanto se sigue el arco (PUSH SNOM/T ...). El PUSH resulta exitoso, ya que "el examen" es un sintagma nominal.

Ahora la función ⋆ tendrá como valor:

(SINT-NOMINAL (DET el) (SUSTANTIVO examen))

y al ejecutar la acción:

(SETR PREDICADO (BUILDQ (PREDICADO (VERBO +) ⋆)
VERBO))

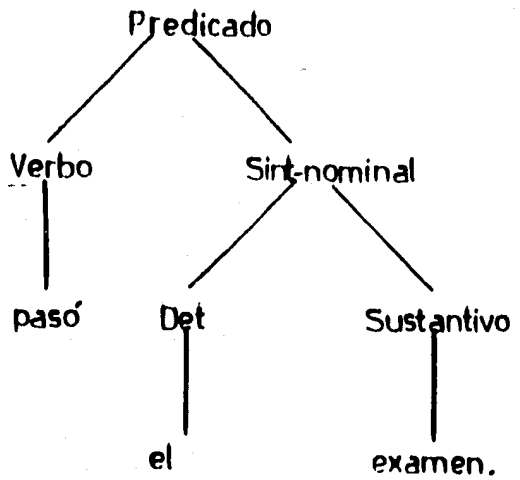
el registro PREDICADO tendrá el siguiente contenido:

(PREDICADO (VERBO pasó)

(SINT-NOMINAL (DET el)

(SUSTANTIVO examen)))

que equivale al subárbol:



Al realizar la acción (HACIA Q5), se avanzará en la cadena de entrada, lo que ocasionará que la forma * entregue el valor NIL (nulo) debido a que ya no hay elementos en la entrada.

5. En el estado Q5 ya se puede seguir el arco marcado por POP, debido a que la cadena de entrada se ha agotado y el stack está vacío (no hay ninguna llamada recursiva pendiente).

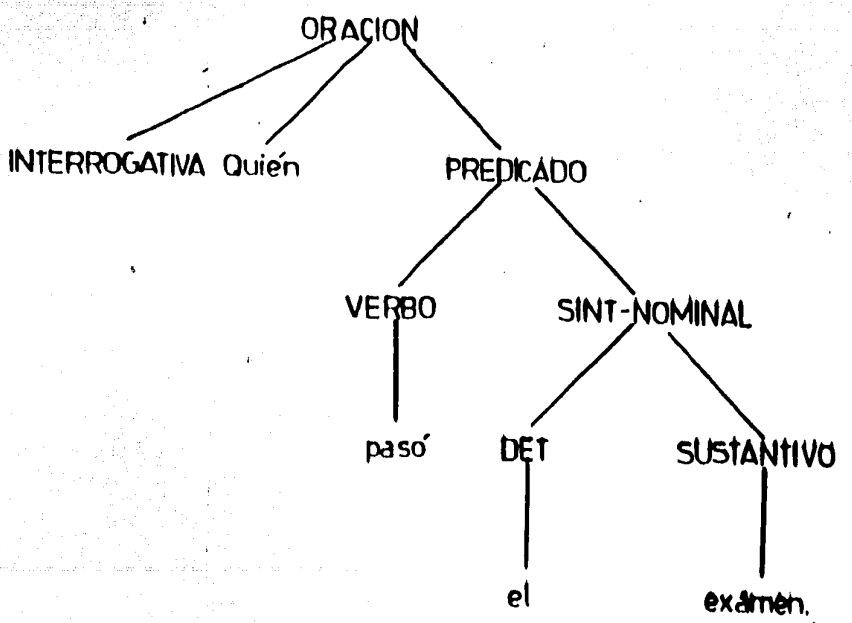
Se realizará entonces la siguiente acción:

(BUILDQ (ORACION + + + +)TIPO SUJETO INT PREDICADO)

Lo que entregará como valor:

```
(ORACION
   INTERROGATIVA
   Quién
   (PREDICADO (VERBO pasó)
              (SINT-NOMINAL (DET el)
                            (SUSTANTIVO exámen))))
```

que equivale al siguiente árbol:



que es el árbol final de la oración.

La gran claridad de este tipo de estructuras es prácticamente imposible de obtener a partir de una gramática libre de contexto, y por ende a partir de una red de transición recursiva.

Una vez demostrado el gran potencial de las redes de transición aumentada, se enuncian las características que lo han hecho tan popular.

3.4.3.2 CARACTERISTICAS PRINCIPALES. -

El modelo de redes de transición aumentadas para la representación de las gramáticas tiene muchas ventajas cuando se emplea como modelo para el análisis del lenguaje natural, algunas de las cuales pueden también ser útiles para el análisis de lenguajes de programación. A continuación se enuncian brevemente algunas de sus características principales, que lo han convertido en un modelo atractivo para el lenguaje natural.

1. CLARIDAD: Las redes de transición aumentadas proveen el poder expresivo de una gramática transformacional, manteniendo mucho de la perspicuidad del modelo de las gramáticas libres de contexto.

2. **POTENCIA GENERATIVA: Aún sin las condiciones en los arcos, las redes recursivas tienen mayor poder generativo que las gramáticas libres de contexto.** Cuando se aumentan las condiciones en los arcos, el modelo obtiene la potencia generativa de una máquina de Turing, aunque las operaciones básicas que realice sean naturales para el análisis del lenguaje.

3. EFICIENCIA DE REPRESENTACION: Esta es una de las grandes ventajas de las redes de transición, y consiste en su habilidad para mezclar las partes comunes a muchas reglas de una gramática libre de contexto, permitiéndoles ser más eficientes en su representación.

4. CAPTACION DE REGULARIDADES: Una de las metas lingüísticas de una gramática es la de poder captar las regularidades del lenguaje que describen. Esto consiste en lo siguiente: si hay algún procedimiento regular que opera en un número de ambientes, la gramática debe tener ese proceso en un sólo mecanismo o regla, y no en varias copias independientes del mismo proceso para cada contexto en el que ocurre.

El modelo de redes de transición aumentadas, con las condiciones arbitrarias en sus arcos y el uso de registros que contienen banderas y construcciones parciales, provee de un mecanismo para reconocer y captar regularidades.

5. EFICIENCIA DE OPERACION: Además de la eficiencia de operación resultante de reunir partes comunes de reglas diferentes, el modelo da la posibilidad de posponer las decisiones mediante la alteración de los registros.

6. FLEXIBILIDAD PARA EXPERIMENTACION: El conjunto de operaciones básicas empleadas en los arcos permite el desarrollo de un conjunto fundamental de operaciones "naturales" para el análisis del lenguaje natural, mediante la experiencia obtenida al escribir gramáticas, y permite también la investigación de diversos tipos de representaciones estructurales. Adicionalmente, es posible producir algunas representaciones semánticas mediante las acciones de construcción de la estructura contenidas en los arcos.

Finalmente, es posible emplear las condiciones en los arcos para experimentar con diversos tipos de condiciones semánticas que quien el análisis, reduciendo así el número de análisis insignificantes.

Las razones anteriores han permitido que las redes de transición aumentadas tengan el gran auge que tienen actualmente, que las ha convertido en el formalismo más comúnmente empleado en los sistemas de comprensión del lenguaje natural. Existen, como siempre, alternativas que pretenden superar algunas de las limitaciones que presenta

este modelo, como la teoría del análisis determinístico de Marcus. Sin embargo, las redes de transición aumentadas han sido empleadas con tanto éxito que todavía no se aprecia un formalismo que pueda competir con ellas. De hecho se puede afirmar que constituyen el paradigma actual en la representación de gramáticas para el análisis de lenguajes naturales.

El punto principal en el desarrollo de sistemas de comprensión de lenguaje natural está siendo desplazado del análisis sintáctico al análisis semántico. Es por ello que no se esperan cambios significativos en ésta área, razón por la cual se decidió presentar este formalismo de representación gramatical como el propuesto en el presente trabajo.

4.1 INTRODUCCION.

El objetivo de este capitulo es el de presentar el lenguaje que será empleado como salida del analizador semántico de la interfaz para consultar bases de datos en lenguaje natural.

Para ello se hablará primeramente de algunos conceptos básicos y objetivos de las bases de datos.

El algebra relacional es un lenguaje matemático desarrollado especificamente para bases de datos de estructura relacional. Este es el tema que se aborda después, para finalmente definir algunas operaciones del álgebra relacional que serán las empleadas en el analizador.

4.2 BASES DE DATOS.

Una de las principales características de una computadora es su capacidad para almacenar grandes cantidades de datos. Cuando se desea almacenar información en la computadora, se debe dar a esos datos una estructura determinada. No se puede usar la computadora como un almacén desorganizado, debido a que sería imposible recuperar la información. Es necesario estructurar los datos almacenados con el fin de que la información se pueda recuperar ordenada, eficiente y confiablemente.

La teoría y la práctica relativa a bases de datos se desarrolló al final de la década de los 60's y a lo largo de la de los 70's.

Esta necesidad surgió por la rápida evolución de lenguajes de programación, que facilitó el desarrollo de aplicaciones que requerían **compartir datos**.

La evolución de los sistemas de almacenamiento de datos se puede clasificar -a grosso modo- en tres etapas (como lo hace Rohde en "Language tools for data access...past, present, and future" [ROH81]):

- La fase orientada a **archivos**, en que se asumen los datos como del dominio de un solo programa, no como parte integral de una aplicación; el problema es que este enfoque resulta difícil de emplear y además poco adaptable.

- La **primera generación de bases de datos**, en que los datos son vistos como la parte principal de una aplicación. En esta etapa se plantea la posibilidad de que los datos sean independientes de los programas, y se concibe que los programas teóricamente "floten" sobre una estructura de datos definida con antelación. Los primeros sistemas de CODASYL y los sistemas jerárquicos de administración de datos reflejan este concepto.

- **En la segunda generación de bases de datos** se hace énfasis en la independencia de los datos. Esto se logró mediante el aislamiento de los atributos físicos de los datos. La base de datos se aprecia como una estructura estratificada de tres niveles (interno, conceptual y externo) que prácticamente aísla los programas de los datos, además de proporcionar al usuario la posibilidad de apreciar la estructura de su base de datos.

- La **tendencia actual** es la de intentar aplicar toda la teoría desarrollada en las etapas anteriores para representar conocimientos en bases de datos. Otro de los enfoques modernos en bases de datos se avoca al desarrollo de sistemas de bases de datos distribuidos.

C.J.Date define una base de datos como un conjunto de datos almacenados por sistemas de aplicación para algún fin particular [DAT82]. Una base de datos sirve para modelar una parte del mundo, incluyendo objetos y las relaciones existentes entre ellos; estos se encuentran representados mediante una colección de datos interrelacionados, almacenados sin redundancias perjudiciales. Su finalidad es la de servir a una aplicación o más, de la mejor manera posible. Los datos se almacenan de tal manera que sean independientes de los programas que los emplean, existiendo procedimientos bien definidos para incluir datos nuevos y para modificar o extraer los datos almacenados.

diversas opiniones sobre cuáles deben ser los objetivos de una base de datos. En estas discusiones se han involucrado grupos tales como la comisión de sistemas del CODASYL, y el grupo especializado de la ANSI, y se pueden plantear algunos objetivos en que coinciden la mayoría de ellos [DAT82, ULL80]:

- Reducir redundancia.
- Eliminar inconsistencias.
- Compartir datos entre usuarios.
- Reforzar la implantación de estándares.
- Aplicar restricciones de seguridad.
- Mantener la integridad de los datos.
- Balancear requerimientos conflictivos.
- Lograr independencia entre datos y programas.

El modelo de arquitectura de tres niveles propuesto por la ANSI [ANS75], y que se utiliza como referencia en la mayoría de la literatura especializada es el que se muestra en la figura 4.1.

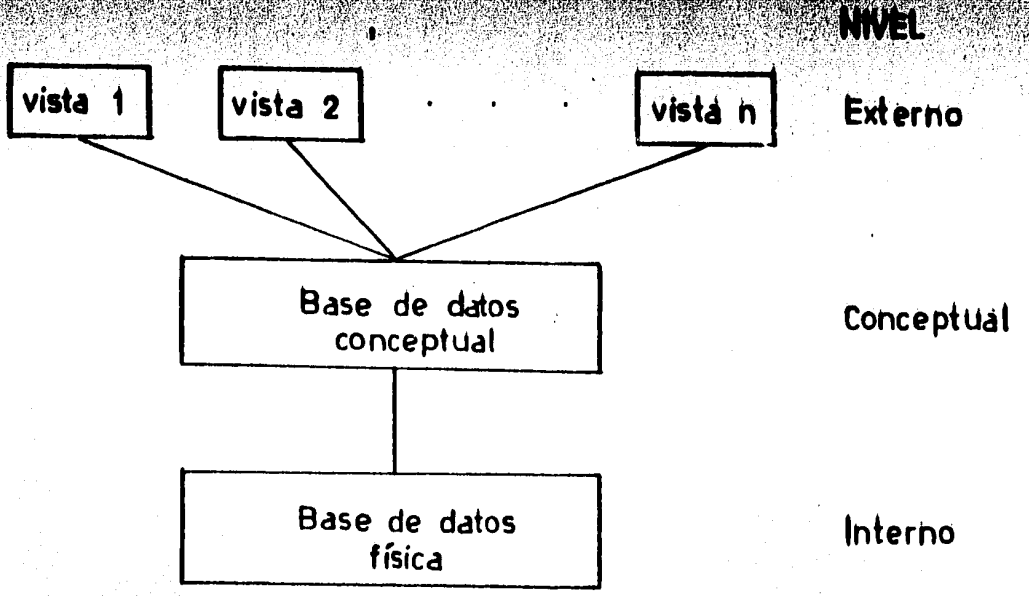


FIGURA 4.1.- Arquitectura de tres niveles para bases de datos.

El nivel interno es el que maneja directamente la información en los medios físicos de almacenamiento (generalmente discos magnéticos).

El nivel conceptual de un sistema de bases de datos es una representación canónica de la información contenida en la base de datos, y contiene un grado mayor de abstracción con respecto al nivel interno.

El nivel externo es el más cercano al usuario, y se compone de una serie de vistas individuales que describen la base de datos tal y como la concibe el usuario (desplegados, reportes, tablas, redes, etc.).

El modelo conceptual de una base de datos es la vista total de su contenido. Existen dos tipos de modelos:

- El modelo conceptual activo [DAT82] también llamado **esquema**, que se encuentra directamente acoplado a un sistema de manejo de datos, y es utilizado como nivel de indirección entre los niveles interno y externo.

- El modelo conceptual pasivo, tambien llamado **diccionario de datos**, que es independiante del sistema de manejo de datos.

En la visión externa (que se presenta al usuario) se omiten detalles de la manera en que se representan los datos en los dispositivos de almacenamiento. El tipo de estructuras de datos que se manejen a nivel usuario (externo o conceptual) es un factor que afecta de manera directa a muchos componentes del sistema. En particular, dicta el diseño de los lenguajes de manejo de datos, ya que cada operación en ellos estará definida en función de sus efectos sobre las estructuras de datos que componen la base de datos. Por eso es que la desición sobre qué estructuras de datos y operadores debe presentar el sistema es crucial en

su diseño. Se puede entonces categorizar los sistemas de bases de datos de acuerdo con el enfoque empleado en el diseño de sus estructuras de datos. Los cuatro enfoques más difundidos son los siguientes:

- El enfoque jerárquico.
- El enfoque de redes.
- El enfoque relacional.
- El enfoque semántico.

Brevemente estos modelos se pueden describir como sigue:

El modelo jerárquico conceptualiza o modela el mundo en jerarquías. Este fue el primer modelo formal de información desarrollado para bases de datos y surgió de la suposición de que el mundo está estructurado en jerarquías: los consorcios se componen de compañías, las que se componen de departamentos, los que tienen empleados; todo en una relación padre-hijo, en diferentes niveles.

Esta técnica es apropiada para muchas aplicaciones, mas rápidamente encontró limitaciones, siendo la principal la dificultad para modelar relaciones n-a-m (por ejemplo, un proyecto tiene n personas asignadas y una persona puede colaborar en m proyectos).

Posteriormente se desarrollaron los modelos de red para bases de datos (CODASYL). Este modelo conceptualiza el mundo como conjuntos de información ligados en red por caracteristicas comunes. Por ejemplo, el conjunto de plantas generadoras de un sistema eléctrico lo se puede estructurar en redes por el área o centro de control que las superviza. Esto permite viajar ("navegar") por una red para visitar por ejemplo todas las plantas del área Occidental.

Este modelo es particularmente eficiente cuando los datos se estructuran pensando en caminos de acceso que serán frecuentemente utilizados por las aplicaciones. Su desventaja principal es que impone al diseñador de la base de datos el conocimiento de las aplicaciones, por una parte, y una especificación detallada y compleja de la estructuración de su información en el modelo.

En 1969 Codd propuso el modelo relacional para bases de datos. Este modelo, al contrario de los anteriores, primero tuvo un desarrollo teórico formal y después surgieron las implementaciones. Esto le da la base matemática con la solidez de que carecen el modelo jerárquico y el modelo de redes.

El modelo relacional conceptualiza el mundo como conjuntos de relaciones (en el sentido matemático del término) y su teoría se apoya en la de conjuntos.

Las principales ventajas del modelo relacional son su simplicidad, que lo hace particularmente accesible a diseñadores y usuarios, y el hecho de que forma un sistema cerrado (en el sentido matemático del término).

La crítica principal que se ha hecho al modelo relacional, o mejor dicho a sus implementaciones, es el gran consumo de recursos computacionales y el tiempo de respuesta resultante al hacer consultas a la base de datos. Sin embargo, debido a la utilización de técnicas de optimación y el desarrollo de técnicas de indexado ("B-trees" y "hashing", por ejemplo), este argumento ha perdido fuerza y tiende a no ser aplicable.

De hecho, en la actualidad el modelo relacional es el que tiene más difusión y aceptación, sobre todo debido a la disponibilidad de manejadores de bases de datos relacionales en micro-computadoras. En el ambiente de centros de control para sistemas eléctricos de potencia también se visualiza el modelo relacional como el ideal para utilizarse como estándar [WIN85].

En la actualidad se han desarrollado modelos semánticos de bases de datos, que persiguen ser capaces de almacenar conocimientos. Esta tendencia ha surgido debido al gran desarrollo alcanzado por los sistemas expertos, y se plantea como una alternativa para la representación del conocimiento. Su problema básico es semejante al de las

redes semánticas: los conocimientos y el procedimiento que los aplica son muy difíciles de separar uno del otro. A semejanza de las redes semánticas, que necesitan procedimientos ad-hoc para su aplicación, las bases de conocimientos con enfoque semántico contienen información procedural muy especifica.

Por estas razones se eligió el álgebra relacional como la salida más apropiada de la interfaz para la consulta de bases de datos en lenguaje natural, enfocando su aplicación a la consulta de bases de datos relacionales.

4.3 EL MODELO DE DATOS RELACIONAL.

El concepto matemático en que se basa el modelo relacional es el conjunto teórico llamado **relación**, que es un subconjunto del producto cartesiano de una lista de dominios.

Un **dominio** es simplemente un conjunto de valores; por ejemplo, el conjunto de los números naturales es un dominio. Así, se pueden definir los dominios que se desee como el conjunto de cadenas de caracteres, el conjunto de cadenas de caracteres de longitud igual a 20, los números reales, el conjunto {0,1}, etc.

El **producto cartesiano** de los dominios D_1, D_2,... D_k, escrito $D_1 \times D_2 \times \ldots \times D_k$, es el conjunto de todos los k-tuplos $(v_1, v_2, \ldots, v_k)$ tales que v_1 está contenido en D_1,

v_2 está contenido en D_2 y así sucesivamente.

Por ejemplo, si k=2, D_1={0, 1}, D_2={a, b, c} y $D_3 = D_1 \times D_2$, entonces:

$$D_3 = \{(0,a), (0,b), (0,c), (1,a), (1,b), (1,c)\}$$

Una relación es entonces un subconjunto finito cualquiera del producto cartesiano de uno o más dominios. Por ejemplo:

$$R_1 = \{(0,a), (0,b), (1,a)\}$$

es una relación, un subconjunto de D_3, definido arriba. El conjunto vacío es otro ejemplo de una relación.

Los miembros de una relación se llaman **tuplos**. Si se concibe la relación como una tabla, los tuplos corresponden a los renglones que constituyen dicha tabla.

El número de tuplos de una relación se denomina **cardinalidad** de una relación. Siguiendo la analogía establecida, la cardinalidad de una relación será el número de líneas que tenga la tabla.

El **grado** de una relación que es subconjunto del producto $D_1 \times D_2 \times \ldots \times D_n$ es n, o sea el número de columnas de la tabla. Es así como se puede decir que la relación R_1 -definida arriba- es de grado 2 o binaria; un dominio tiene grado uno, o es unario, las relaciones de grado 3 serán terciarias, y las relaciones de grado n serán n-arias.

Es por esto que Ullman y Aho le llaman "aridad" al grado de una relación [ULL80].

Podemos ahora redefinir relación de la siguiente manera:

> "Dada una colección de conjuntos D_1, D_2, ..., D_n (no necesariamente diferentes), R es una **relación** sobre esos n conjuntos si es un conjunto de n-tuplos ordenados $\langle d_1, d_2, \ldots, d_n \rangle$, tal que d_1 está contenido en D_1, d_2 está contenido en D_2, ... y d_n está contenido en D_n. Los conjuntos D_1, D_2, ..., D_n son los dominios de R, y el valor n es el grado de R." [DAT82].

No hay ningún orden definido entre los tuplos de una relación, ya que una relación es un conjunto y los conjuntos no son ordenados. Sin embargo, resulta útil establecer un orden determinado cuando se trata particularmente de una base de datos con esta estructura. Sin embargo, el orden de los elementos de un tuplo si se explicita, y resulta necesario respetarlo.

4.3.1 ATRIBUTOS. -

Es necesario apreciar la diferencia entre dominio y atributo -que se obtiene de un dominio-. Un atributo representa el **uso** de un dominio en una relación. De hecho, se puede dar un nombre a los atributos y otro diferente a los dominios subyacentes para enfatizar esta diferencia.

las relaciones en una base de datos deben satisfacer la siguiente condición:

- **Cada valor en la relación** -cada valor de atributo en cada tuplo- es atómico (no se puede descomponer, en lo que concierne al sistema).

Una relación que satisfaga ésta condición se denomina **normalizada.**

4.3.2 LLAVES. -

En todas las relaciones se da el caso de que un conjunto de atributos tiene valores únicos para cada tuplo dentro de una relación. Se dice que este conjunto no vacío de atributos es la **llave primaria** de esa relación. La existencia de tal combinación está garantizada por el hecho de que una relación es un conjunto; ya que los conjuntos no pueden tener elementos duplicados, cada tuplo de una relación dada es único dentro de esa relación, y la combinación de cuando mucho todos sus atributos tiene la propiedad de unicidad. Por lo tanto, todas las relaciones tienen una llave primaria (posiblemente unaria). Se asumirá que la llave primaria es no redundante, en el sentido que ninguno de sus atributos constitutivos es superfluo para el propósito de identificación única.

En algunas ocasiones se puede encontrar una relación en la que haya más de una combinación de atributos con la propiedad de unicidad, y por lo tanto, habrá más de una llave candidato. En tal caso, se puede elegir arbitrariamente uno de los candidatos como la llave primaria para la relación. Una llave candidato que no es la llave primaria, se denomina **llave alterna**.

La llave es simplemente un identificador de los tuplos de una relación. De hecho, estos tuplos representan entidades en el mundo real, y la llave primaria sirve realmente como identificador único de esas entidades. Esto lleva a imponer la siguiente regla para las llaves:

PRIMERA REGLA DE INTEGRIDAD (Integridad de las Entidades).

Ningún componente del valor de la llave primaria puede ser nulo.

El razonamiento que apoya esta regla es el siguiente: Por definición, todas las entidades deben ser distinguibles -deben tener una identificación única-. Las llaves primarias realizan esta función en una base de datos relacional. Un identificador (valor de llave primaria) con valor nulo sería una contradicción, debido a que estaría implicando la existencia de una entidad sin identificación

única -o sea que no sería distinguible de las demás entidades-. Y si dos entidades no se distinguen una de la otra, entonces por definición no son dos entidades, sino sólo una.

Es común que una relación contenga referencias a otra relación. Para explicarlo con claridad, se definirá la siguiente noción:

- **Dominio primario.**- Un dominio se designa primario si y sólo si existe alguna llave primaria de un solo atributo definida sobre este dominio.

Entonces cualquier relación que incluya un atributo definido sobre un dominio primario debe obedecer la siguiente restricción:

SEGUNDA REGLA DE INTEGRIDAD (Integridad Referencial).

Sea D un dominio primario, y R_1 una relación con un atributo A definido sobre D. Entonces, en cualquier momento, cada valor de A en R_1 debe ser:

a) Nulo, o bien

b) Igual a V, donde V es el valor de llave primaria de algún tuplo en alguna relación R_2 con llave primaria definida sobre D (R_1 y R_2 no son necesariamente distintas).

Se está implicando que R_2 debe existir, por la definición de dominio primario, así como el hecho de que la restricción es satisfecha trivialmente si A es la llave primaria de R_1.

El atributo A se denomina **llave externa**. Las llaves primarias y externas permiten representar las relaciones entre los tuplos.

4.3.3 EXTENSION E INTENSION. -

Una relación en una base de datos relacional tiene dos componentes principales, que en ocasiones se denominan indiferentemente mediante el término "relación":

- La **extensión** de una relación dada es el grupo de tuplos que la conforman en un momento dado. La extensión varía en función del tiempo (si se destruyen, alteran o crean tuplos).

- La **intensión** de una relación dada es independiente del tiempo. Es básicamente la parte permanente de la relación, y constituye el contenido del **esquema**

relacional. Define entonces todas las extensiones posibles. La intención es la combinación de dos cosas:

- La **estructura nominativa**, que consiste en el nombre de la relación y los nombres de los atributos (cada una con su nombre de dominio asociado).
- Las **restricciones de integridad** que pueden ser divididos en restricciones de llave, restricciones referenciales y otras restricciones.

4.3.4 RESUMEN. -

Para finalizar, se puede decir que, en términos tradicionales, una relación redefine un archivo, un tuplo redefine un registro (ocurrencia, no tipo) y un atributo redefine un campo (tipo, no ocurrencia). En otras palabras, las relaciones pueden ser concebidas como archivos altamente disciplinados.

En la siguiente figura se concentran las analogías realizadas:

RELACION	TABLA	ARCHIVO
Tuplo	Linea	Registro
Nombre de atributo	Numero de columna	Nombre de campo
Nombre de dominio	Tipo de columna	Tipo de campo

El siguiente es un resúmen de las restricciones presentadas:

i) Ningún par de tuplos en una relación pueden ser iguales: cada uno debe tener un valor diferente para su llave primaria, que no debe contener ningún valor nulo.

ii) Todos los tuplos de una relación deben tener el mismo número de atributos en el mismo orden.

iii) Los valores de cada atributo deben ser extraidos de un dominio fijo.

iv) Los valores de los atributos deben ser atómicos -no pueden tener componentes-; las relaciones no pueden tener otras relaciones como componentes.

v) Una base de datos relacional será consistente si se respetan algunas restricciones extra tales como la "integridad referencial".

4.4 ALGEBRA RELACIONAL.

El álgebra relacional es un lenguaje diseñado para trabajar con bases de datos relacionales y consiste en un conjunto de operaciones a realizar sobre las relaciones. Cada operación toma una o más relaciones como argumento(s) y

produce otra relación como resultado. Esta relación resultante puede a su vez ser empleada como argumento de otras operaciones algebraicas. Los argumentos de cualquier operación dada pueden ser entonces nombres de relaciones o bien expresiones que sean evaluadas como relaciones. Las expresiones del álgebra relacional pueden estar anidadas a cualquier nivel de profundidad. En términos matemáticos, el hecho de que cualquier operación algebraica resulte en otra relación, se expresa diciendo que las relaciones forman un sistema cerrado bajo el álgebra.

Codd, en "Relational Completeness of Database Sublanguages" [COD72], propone las operaciones principales del álgebra relacional, demostrando que es "Relacionalmente completa" mediante su algoritmo de reducción (que presenta en el mismo capítulo). A partir de entonces se han desarrollado muchas variantes de las mismas [DAT82, ULL80, GRA84]. El enfoque propuesto es el desarrollado en la referencia [GRA84], consistente en ASTRID, un lenguaje basado en el álgebra relacional.

El lenguaje ASTRID constituye un álgebra relacional aumentada. Sin embargo, esta sección se concentrará en las operaciones básicas del álgebra relacional, sin tomar en cuenta los aumentos mencionados. La sintaxis completa del lenguaje ASTRID se puede encontrar en [GRA84].

4.4.1 OPERACIONES BÁSICAS DEL ALGEBRA RELACIONAL. -

Para explicar estas operaciones las relaciones serán consideradas como tablas de líneas y columnas. Dos de las operaciones toman sólo una relación como argumento (selección y proyección) y producen versiones modificadas de la misma. Las otras dos operaciones -join e intersección- toman dos argumentos.

4.4.1.1 SELECCION. -

La selección reduce el número de líneas de una tabla, y puede ser concebida como una rutina que corta la tabla horizontalmente removiendo los tuplos no deseados. Formalmente se escribe de la siguiente forma:

RELACION selected_on [⟨predicado⟩] {sintaxis ASTRID}

RELACION [⟨numat⟩ ⟨comp⟩ ⟨numat⟩] {sintaxis de Codd}

⟨predicado⟩ es una expresión booleana en función de predicado, que puede contener comparaciones entre valores de atributos y otros valores de atributos en el mismo tuplo, o también constantes. Sólo aquellos tuplos que satisfagan ⟨predicado⟩ se conservarán de la relación original.

4.4.1.2 Proyección.-

La proyección reduce el número de columnas de la tabla y puede concebirse como una rutina que la corta verticalmente. Su nombre se deriva de la noción de proyectar un número de puntos en un espacio de n dimensiones a otro de menos dimensiones y, por lo tanto, de menos componentes. Nótese que los valores proyectados de tales puntos pueden coincidir; esto sucede cuando se ha eliminado en la proyección alguna columna que forme parte de la llave principal de la relación. Entonces, se remueven los tuplos duplicados y por lo tanto se reduce el número de renglones de la tabla. Sin embargo, si por lo menos queda intacta una llave candidato en todas las lineas, entonces no habrá duplicados.

La operación se escribe como:

RELACION projected_to <nomat> [,<nomat>] {sintaxis ASTRID}

RELACION [<numat> [,<numat>]] {sintaxis de Codd}

Donde la lista de nombres de atributos (<nomat>s) denota los nombres de las columnas de la tabla a conservar; Codd, en cambio, emplea números de columnas. Los <nomat>s deben ser, por supuesto, nombres de atributos de RELACION.

4.4.1.3 INTERSECCION. -

La intersección de dos relaciones (compatibles para unión) S y R es la relación formada por los tuplos contenidos en R y en S. Funciona de manera idéntica a la intersección de conjuntos, y se escribe:

S intersect_with R

4.4.1.4 JOIN. -

Cuando dos tablas (relaciones) tienen una columna (atributo) definida sobre el mismo dominio, pueden ser "acopladas" sobre estas columnas; el resultado del join es una nueva tabla (relación) de mayor grado, en la que cada linea se forma concatenando dos lineas, una de cada una de las tablas originales, de tal manera que las dos tablas tengan el mismo valor en esas dos columnas (atributos). Esta es la definición particular de un join en que la **condición de acoplamiento** se basa en la igualdad entre los valores de la columna común. Esta clase de join recibe el nombre de **equijoin**.

En la relación que resulta de acoplar dos relaciones mediante un equijoin, se tiene el problema de que el atributo sobre el que se realizó el acoplamiento estará repetido en cada tuplo. Si se elimina dicha repetición, se

tendrá como resultado un join natural.

Se expresará a la salida un join natural. Sin embargo, tendrá una característica particular: dado que los atributos que se deben emplear en el join son los atributos llamados "llaves externas", ya sea heredados a o de otra relación, el join expresado a la salida del analizador semántico implicará exclusivamente el uso de los atributos de relación.

Formalmente:

Sean:

R y S	dos relaciones.
R.k	el atributo llave de R.
S.k	un atributo de S.
R.k y S.k	definidos sobre el mismo dominio.

Entonces:

R joined_to S = (R produced_with S) selected_on [R.k = S.k]

Donde:

R produced_with S	Es el producto cartesiano de R con S [GRA84, DAT82].

Se llamará a k el **atributo de relación** entre R y S.

S.k será el **atributo heredado** por R a S.

4.4.2 RESUMEN. -

El álgebra relacional es un lenguaje relacionalmente completo inventado por E.F.Codd para trabajar con relaciones. Las relaciones forman un sistema matemáticamente cerrado bajo el álgebra relacional.

Los operadores empleados para expresar la salida del analizador semántico serán los siguientes:

- Selección.- Corta horizontalmente la relación.
- Proyección.- Corta verticalmente la relación.
- Unión.- Une dos relaciones compatibles para la unión.
- Join.- Acopla dos relaciones. La versión modificada trabajará sólo con los atributos de relación.
- Intersección.- Entrega una relación con los tuplos contenidos en ambas relaciones.

5 COMPILACION DEL LENGUAJE NATURAL.

5.1 INTRODUCCION.

Este capítulo se ocupa de la teoría desarrollada para poder apoyar el diseño del analizador semántico de un compilador para consultar bases de datos en lenguaje natural. El código fue desarrollado en VAX-LISP, y se encuentra funcionando en un ambiente VAX/VMS.

Dicha teoría será expuesta en el mismo orden en que se desarrolló, con el fin de que se pueda seguir la misma secuela de razonamiento empleada en el proceso.

Las bases de esta teoría se encuentran en la consideración del modelo conceptual ELKA como una manera de representar el esquema de una base de datos. Es por ello que la parte inicial de este capítulo se ocupa de exponer los conceptos más elementales de este modelo conceptual.

El cuestionamiento que dio principio al planteamiento es el siguiente: ¿Qué información se puede obtener de una pregunta original en lenguaje natural que permita encontrar la respuesta buscada por el usuario en la base de datos?. Una de las limitaciones para responder era el objetivo para el que se deseaba extraer dicha información. En un

principio, se concluyó que la información necesaria se limitaba a las referencias que se hacían a los objetos de la base de datos mediante sustantivos, sinónimos, nombres de roles, etc. De esta premisa se desprendieron conceptos como el de "macroentidad" y el de "viaje" por el esquema que se exponen en la sección 5.3. La conclusión a la que se arribó no es trivial, ya que se basa en la información obtenida de una encuesta realizada para tal fin; el análisis de los resultados de esta encuesta se muestra en el apéndice A.

El siguiente paso fue el de formalizar una estructura en la que se reflejaran los conceptos obtenidos. Dicha estructura se denomina matriz de caminos, y es el tema de la sección 5.4.

Posteriormente se encontró que la información extraida de la pregunta en lenguaje natural llevaba a problemas de ambigüedad semántica. Fue entonces cuando se decidió desarrollar el concepto de matriz semántica, que aprovecha la información semántica del verbo de la pregunta. Este es el siguiente tema presentado en la sección 5.5.

Finalmente, en la sección 5.6, se conjuntan estos conceptos para enumerar las acciones que realiza el prototipo de analizador semántico desarrollado, para terminar presentando el código fuente del mismo en el que se señalan las acciones que ennumeradas. En el apéndice B de esta tesis se muestran algunos ejemplos de la operación del

programa.

5.2 EL MODELO CONCEPTUAL ELKA.

Los elementos principales del modelo ELKA [ROD81], como su nombre lo indica, son las entidades, las ligas, las llaves, y los atributos. La exposición siguiente se concentra en la presentación de estos conceptos, además de los simbolos gráficos empleados para representarlos.

5.2.1 ENTIDADES, LLAVES Y ATRIBUTOS. -

Una **entidad** es un ser real o virtual sobre el que se tiene información. Se puede decir que es un objeto descrito por propiedades cuyos valores se pueden considerar fijos durante algún tiempo. Una **clase de entidades** es un conjunto de entidades que el usuario y/o el modelador de la base de datos han decidido agrupar debido a que tienen el mismo tipo de propiedades de interés. Cada clase de entidades tiene un nombre único y cada entidad (elemento) de esta clase tiene identidad única en esa clase.

Una **entidad** se representa como un conjunto de atributos considerados generalmente como sus propiedades. Un **atributo** se representa como un par compuesto por un nombre de clase de atributo y un valor de atributo. Se dice que una entidad despliega sus atributos.

En el simbolismo gráfico del modelo ER, una clase de entidades se representa como una tabla encerrada en un rectángulo. El nombre de la clase de entidades aparece enmarcado en la esquina inferior izquierda del rectángulo; los nombres de atributos se muestran como nombres de columnas en el encabezado de la tabla, y los valores de los atributos de cada entidad forman las lineas de la tabla. Si se remueven los valores de los atributos y sólo se conserva el nombre de la clase de entidades y los nombres de los atributos, se obtendrá un **esquema de una clase de entidades.** En la figura 5.1 se muestra un esquema de una clase de entidades llamada PROFESOR.

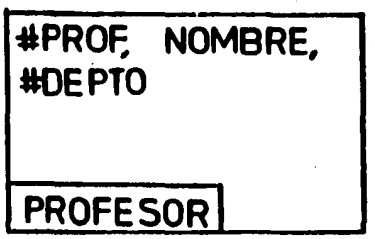


FIGURA 5.1.- Esquema de la clase de entidades PROFESOR.

Con frecuencia se afirma que los elementos de una clase de entidades son "instancias" de su intensión. Por ejemplo, una instancia de la clase de entidades PROFESOR sería la siguiente:

[#prof:10, nombre:DOMINGUEZ, #depto:0205]

Dentro de una clase de entidades, las entidades se caracterizan por sus **llaves**. La llave de una entidad es un conjunto no vacio de atributos que permite distinguirla de todas las demás entidades dentro de su misma clase de entidades. Una práctica común en el simbolismo del modelo ELKA es la de subrayar los atributos de llave. Asi, si el atributo #prof es la llave de la clase de entidades PROFESOR, esto se expresa como se muestra en la figura 5.2.

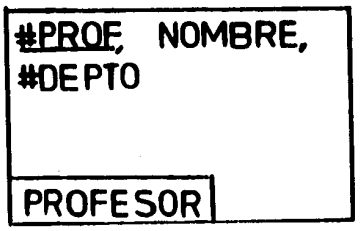


FIGURA 5.2.- Representación gráfica de las llaves.

En el modelo ELKA, las entidades deben tener una llave primaria, pudiendo tener llaves alternas. En este caso, las llaves se agrupan y se identifican por un prefijo. Por ejemplo:

$$K1:(a_{11},\ a_{12},\ \ldots,a_{1n})$$

representa la llave primaria de una clase de entidades, mientras que

$$K2:(a_{21},\ a_{22},\ \ldots,a_{2n})$$

representa una llave alterna.

5.2.2 LIGAS. -

La técnica ELKA de modelado emplea una liga para establecer relaciones entre clases de entidades. Se puede pensar que una liga es una referencia hecha por una entidad a otra, usando una llave de la entidad referenciada. La entidad referenciante se dice que es existencialmente dependiente de la entidad referenciada; esto significa que su existencia depende de la existencia de la entidad referenciada.

Suponiendo que en la misma base de datos en que existe la clase de entidades PROFESOR ilustrada anteriormente, existe una clase de entidades DEPARTAMENTO, cuyo esquema se muestra en la figura 5.3.

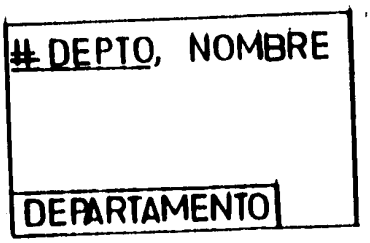


FIGURA 5.3.- La clase de entidades DEPARTAMENTO.

Una instancia de esta clase de entidades podría ser:

{ #depto:0205, nombre:LINGUISTICA }

Si se observa el número del departamento (#depto) de la instancia de la clase de entidades PROFESOR mostrada con anterioridad:

{ #prof:10, nombre:DOMINGUEZ, #depto:0205 }

resulta evidente que esta instancia de la clase de entidades PROFESOR referencia a la instancia citada de la clase de entidades DEPARTAMENTO a través del valor de uno de sus atributos (#depto). Este concepto es el mismo que se introdujo como llave externa cuando se habló de bases de datos relacionales en la sección 4.3.

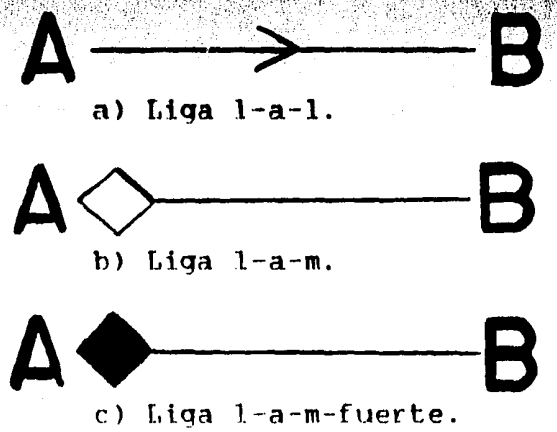
El profesor es **existencialmente dependiente** del departamento; en otras palabras, no puede haber un profesor en la clase de entidades PROFESOR sin que exista el departamento correspondiente en la clase de entidades DEPARTAMENTO; como se puede apreciar, ésta es una restricción de integridad aplicable en el mundo real.

Una clase de ligas es un conjunto de ligas entre entidades. El modelo ELKA tiene tres tipos de clases de ligas que pueden ser descritas como funciones de una clase de entidades A a una clase de entidades B con las siguientes características:

1. Liga **1-a-1** (figura 5.4 a): Para cada miembro de A existe exactamente un miembro de B. Para cada miembro de B existe cero o un miembro de A.

2. Liga **1-a-m** (figura 5.4 b): Para cada miembro de A existe exactamente un miembro de B. Para cada miembro de B existen cero, uno o más miembros de A.

3. Liga **1-a-m-fuerte** (figura 5.4 c): Para cada miembro de A existe exactamente un miembro de B. Para cada miembro de B existen uno o más miembros de A.



a) Liga 1-a-1.

b) Liga 1-a-m.

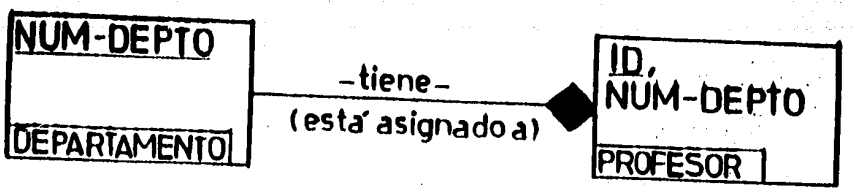
c) Liga 1-a-m-fuerte.

FIGURA 5.4.- Simbolos gráficos de las clases de ligas.

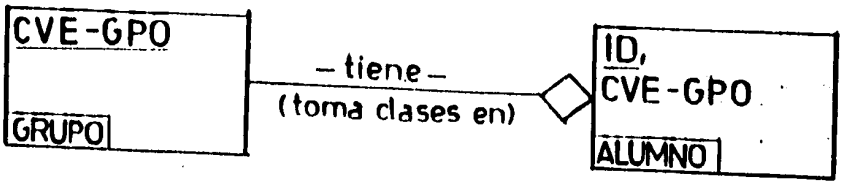
En todos los casos de la figura 5.4 se dice que el lado de la liga en donde se encuentra A es el lado de **atrás** de la liga y el otro es el **frente** de la misma. Se dice entonces que la liga va de A a B. El recorrido de una liga de un elemento de A a un elemento de B será denominado **travesia directa** y en sentido opuesto, se denominará **travesia inversa**.

Las ligas constituyen un puente fisico, asi como una relación conceptual entre dos clases de entidades. Asi, es posible "**navegar**" entre dos entidades por sus ligas.

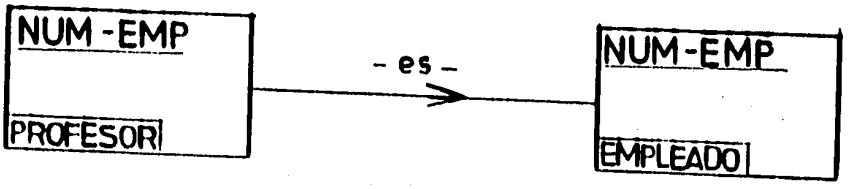
Un **atributo heredado** es un atributo de relación en la parte de atrás de la liga. En la figura 5.5 se muestran algunos ejemplos de ligas entre pares de entidades.



a) Liga 1-a-m.



b) Liga 1-a-m-fuerte.



c) Liga 1-a-1.

FIGURA 5.5.- Ejemplos de ligas entre clases de entidades.

Las ligas tienen un verbo asignado, como se muestra en la figura anterior. Cuando una liga se recorre en travesía directa, el verbo asociado a este recorrido será el que se encuentra entre paréntesis; en cambio, el verbo asociado a la travesía inversa de la liga está encerrado entre guiones. Los nombres de las ligas se acompañan de los nombres de las clases de entidades del frente y atrás. Así, por ejemplo,

los nombres de las ligas de la figura 5.5 son los siguientes:

En travesia directa:

a) ALUMNO.toma-clases-en.GRUPO

b) PROFESOR.está-asignado-a.DEPARTAMENTO

c) PROFESOR.es.EMPLEADO

En travesia inversa:

a) GRUPO.tiene.ALUMNO

b) DEPARTAMENTO.tiene.PROFESOR

Los nombres de las ligas deben ser seleccionados para sugerir la **semántica** de las ligas cuando son recorridas. Es necesario hacer notar que los atributos desplegados en los esquemas que se muestran como ejemplos en éste capitulo son únicamente aquellos que resultan de interés para el caso. De ninguna manera se pretende restringir la posibilidad de que las clases de entidades tengan otros atributos.

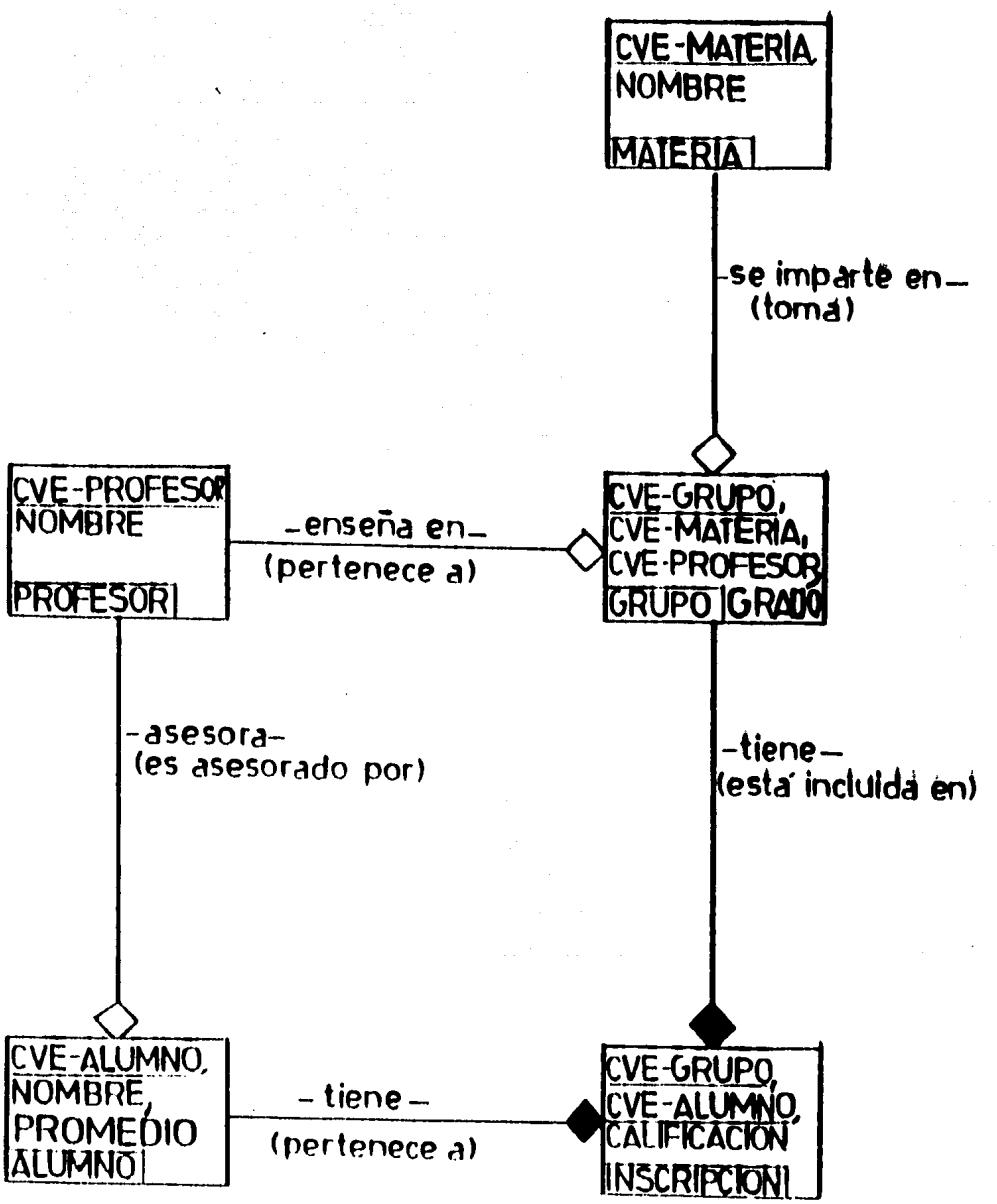


FIGURA 5.6.- Ejemplo de un modelo ELKA.

Se analizará ahora el ejemplo del modelo ELKA con el que se trabajará el resto del capitulo; la razón por la que se eligió trabajar con él es su riqueza estructural y semántica, a pesar de constituirse de sólo cinco clases de entidades. Cabe subrayar que la tesis desarrollada en este capitulo es independiente tanto de la estructura, como del contenido de una base de datos esquematizada en un modelo ELKA; de hecho, el propósito de esta tesis es demostrar dicha generalidad. El modelo se muestra en la figura 5.6 y su semántica es la siguiente:

En una escuela, todos los profesores, miembros de la clase de entidades PROFESOR, enseñan en cero, uno ó más grupos, miembros de la clase de entidades GRUPO. Cada materia, miembro de la clase de entidades MATERIA, se imparte en cero, uno o más grupos. Cada grupo tiene uno o más alumnos inscritos (miembros de la clase de entidades ALUMNO), y cada alumno puede estar inscrito a uno o más grupos. Cada profesor asesora a cero, uno, o más alumnos.

En esta figura se puede ver la forma de representar relaciones n-a-m en el modelo ELKA: un grupo puede tener n alumnos, y un alumno puede estar inscrito en m grupos. Esta relación se lleva a cabo a través de la clase de entidades INSCRIPCION y con la ayuda de ligas 1-a-m.

Los verbos encerrados entre guiones indican la semántica de la liga cuando ésta es recorrida en forma directa; así, la semántica de todas las travesías directas posibles del modelo será:

MATERIA.se-imparte-en.GRUPO

GRUPO.tiene.INSCRIPCION

ALUMNO.tiene.INSCRIPCION

PROFESOR.asesora.ALUMNO

PROFESOR.enseña-en.GRUPO

Los verbos encerrados entre paréntesis indican la semántica de la liga cuando ésta es atravesada en forma inversa; así, la semántica de todas las travesías inversas en el modelo de la figura 5.6 será:

GRUPO.pertenece-a.PROFESOR

ALUMNO.es-asesorado-por.PROFESOR

INSCRIPCION.pertenece-a.ALUMNO

INSCRIPCION.se-incluye-en.GRUPO

GRUPO.toma.MATERIA

5.3 DEDUCCION DE UN VIAJE POR EL MODELO ELKA.

5.3.1 INTRODUCCION. -

La teoría desarrollada a lo largo de este trabajo se basa en el empleo del modelo conceptual ELKA para la representación del esquema de la base de datos.

El esquema será considerado como una estructura reticular que contiene información sobre la organización real de la base de datos. Dicha información está expresada en términos de las tres ligas básicas del modelo conceptual ELKA. Se intenta comprobar que toda la información semántica necesaria para averiguar las relaciones existentes entre las clases de entidades de la base de datos se encuentra implícita en el esquema.

Supongamos que se tiene una base de datos con información sobre un hospital (pacientes, enfermedades, médicos, etcétera). Si se deja "fijo" el nombre de un paciente (Juan Pérez, por ejemplo) en una clase de entidades que contenga a los pacientes, se podrá navegar por el esquema visitando todas las demás clases de entidades del mismo que se encuentren ligadas a ella, encontrando entidades relacionadas con Juan Pérez en todas ellas. Toda la información almacenada sobre Juan Pérez estará contenida en esta "fotografía" de la base de datos.

Si se intenta resolver una pregunta realizada a la base de datos, se debe hacer lo siguiente:

- Obtener la "fotografía" de la base de datos (llamada **macroinstancia**) para las instancias definidas en la pregunta.

- Llegar hasta el lugar de la macroinstancia que resuelve la pregunta.

Lo anterior implica que es necesario realizar un viaje por la estructura del esquema, con el fin de ir obteniendo la parte de la macroinstancia que interesa. En este viaje se "difunden" los efectos del "fijado" de una clase de entidades.

El propósito de esta sección del capítulo es precisamente el de explicar la forma en que se realizará este viaje, partiendo de la pregunta en lenguaje natural, hasta su formulación en álgebra relacional.

5.3.2 DEFINICIONES. -

Con el fin de unificar criterios, se definirán los conceptos necesarios para la clara comprensión del procedimiento descrito.

- **CLASE DE ENTIDAD FIJA: En una pregunta cualquiera, nosotros siempre se dan valores a atributos. Por ejemplo, si se pregunta: ¿Qué edad tiene Salvador? se** está proporcionando el valor de un atributo: el nombre de una persona.

Esta afirmación se respalda en el análisis de la encuesta realizada y documentada en el apéndice A.

Las **clases de entidades fijas** son aquéllas clases de entidades de las que se proporcionan valores de alguno de sus atributos, de tal forma que se hace mención (de manera indirecta) de instancias de dicha clase. Se denominarán clases de entidades fijas, debido a que da la sensación de que han sido "fijadas" poniéndoles una condición.

El resultado de fijar una clase de entidades es el mismo que el de realizar una selección sobre ella en álgebra relacional. Veamos un ejemplo:

Supóngase que el esquema de la base de datos es el representado por el modelo ELKA de la figura 5.6. La pregunta: ¿Qué calificaciones tiéne Carlos Suarez? define una instancia de la clase de entidades ALUMNO. Esta instancia se obtiene al realizar la siguiente operación:

ALUMNO selected_on [NOMBRE = "Carlos Suarez"]

Entonces, la clase de entidades ALUMNO será la clase de entidades fija, y la operación anterior será la equivalente al "fijado" de ésta clase de entidades.

Es conveniente hacer notar que el "fijado" de una clase de entidades puede resultar en más de una entidad -en una relación-, cuando el atributo empleado en la selección no es atributo llave de la clase de entidades aludida.

- MACROINSTANCIA.- En el momento en que se fija una clase de entidades determinada del esquema, se define de manera indirecta un conjunto de instancias de las demás clases de entidades de dicho esquema.

Estas instancias podrán ser obtenidas "difundiendo" los efectos de la fijación realizada. Esto significa que, si se considera que dicha clase de entidades está relacionada con otras clases de entidades, se podrá obtener un conjunto de instancias de las mismas si se "viaja" hacia ellas.

El viaje resulta sencillo, tomando en cuenta el hecho de que las clases de entidades se encuentran relacionadas a través de sus atributos de relación heredados a, o por otras entidades. A continuación se verá un ejemplo para aclarar lo anterior.

Supongamos que la pregunta que se realiza es la siguiente:

¿Qué calificaciones tiene José Luis?

y que el esquema es el que se muestra en la figura 5.7.

Será necesario fijar la clase de entidades ALUMNO en el esquema, mediante la siguiente selección:

S1 = ALUMNO selected_on [NOMBRE = "José Luis"]

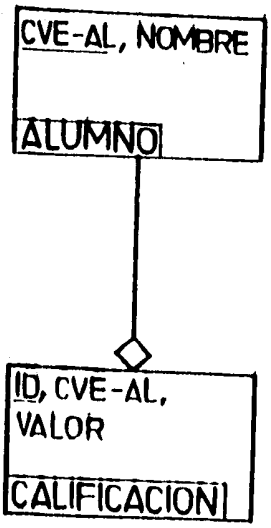


FIGURA 5.7.- Ejemplo del esquema de una base de datos.

El efecto de esta fijación podrá ser difundido al realizar las siguientes operaciones:

(S1 projected_to CVE-AL) joined_to CALIFICACION

y de esta forma se habrá obtenido parte de una "fotografía" de la base de datos para el alumno elegido.

- **CLASE DE ENTIDADES SOLUCION.- Es aquella clase de entidades de la cual se puede obtener la información que responde a la pregunta realizada.**

Normalmente esta clase de entidades es diferente a la clase de entidades fija, por lo que habrá que difundir el efecto de la clase de entidades fija hasta la clase de entidades solución.

En el ejemplo de la definición anterior, CALIFICACION es la clase de entidades solución a la pregunta.

- VIAJE.- Un **viaje** por el esquema de datos es la difusión del efecto sobre las clases de entidades fijas por el esquema.

Si se viaja de la(s) clase(s) de entidad(es) fijas(s) hacia todas las demás clases de entidades del esquema, se tendrá una macroinstancia de la base de datos definida por estas fijaciones. Sin embargo, no interesa realizar tantos viajes, ya que hay sólo una clase de entidades solución. Es por eso que si se viaja de las clases de entidades fijas a la clase de entidades solución al final se obtendrá una relación con información suficiente para responder a la pregunta que definió la macroinstancia.

5.3.3 PROCEDIMIENTO. -

Con el fin de ilustrar la manera en que interactúan estos conceptos para responder a una pregunta a la base de datos, se presentarán algunos ejemplos seleccionados de entre los obtenidos en la encuesta (apéndice A).

El modelo de datos utilizado será el de la figura 5.8, que es el mismo que el presentado anteriormente en la figura 5.6, y que se repite para facilidad del lector.

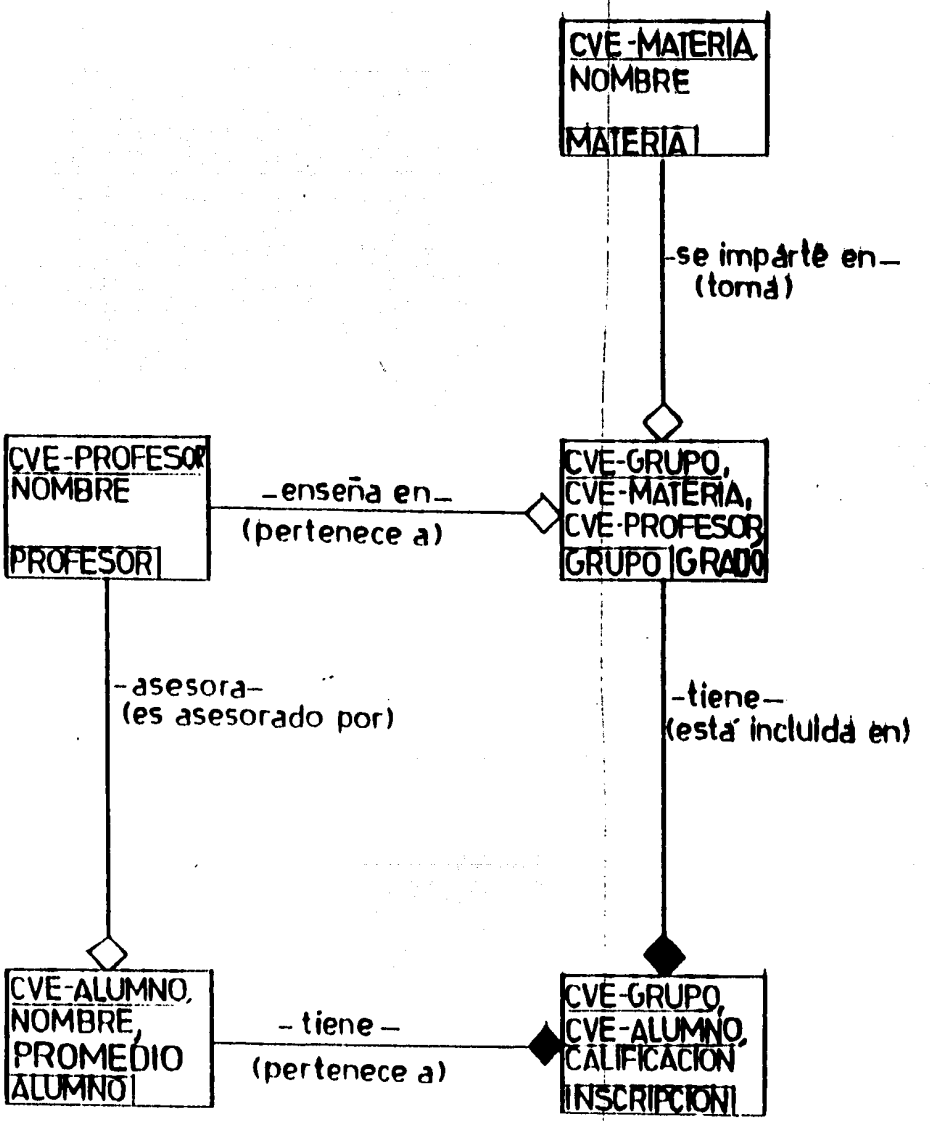


FIGURA 5.8.- Esquema ejemplo de la base de datos de una escuela.

Los ejemplos serán preguntas realizadas a una base de datos con este esquema, y se explicará el procedimiento para encontrar su solución.

EJEMPLO 5.3.1.

¿Qué calificaciones tiene Juan Cardini?

Lo primero por hacer es encontrar las clases de entidades fijas y la clase de entidades solución. En este caso, la clase de entidades fija es ALUMNO, y la clase de entidades solución es INSCRIPCION. Por lo tanto, los pasos a seguir serán:

1. Fijar la clase de entidades ALUMNO con una selección, tomando como atributo de selección el NOMBRE (Juan Cardini).

2. Difundir este fijado, viajando desde ALUMNO hacia INSCRIPCION. Si se tiene en cuenta que alumno hereda CVE-ALUMNO a INSCRIPCION, el viaje consiste en:

   - Proyectar el tuplo obtenido en (1) al atributo de relación (CVE-ALUMNO).

   - Realizar un JOIN, entre lo obtenido e INSCRIPCION.

3. Una vez en la clase de entidades solución, es necesario proyectar al atributo que soluciona la pregunta (CALIFICACION).

La expresión final en álgebra relacional será la siguiente:

1-> (((ALUMNO selected_on [NOMBRE = "Juan Cardini"])

2-> projected_to CVE-ALUMNO) joined_to INSCRIPCION)

3-> projected_to CALIFICACION

EJEMPLO 5.3.2.

¿Qué promedio tienen los alumnos de segundo año?

Se realizará el mismo procedimiento que en el ejemplo anterior:

CLASE DE ENTIDADES FIJAS: GRUPO (GRADO = 2)

CLASE DE ENTIDAD SOLUCION: ALUMNO (PROMEDIO)

Si se observa el esquema de la base de datos, se aprecia que será necesario realizar un viaje más largo de GRUPO a ALUMNO. El viaje tendrá que pasar por INSCRIPCION (más adelante se justificará esta afirmación) resultando en lo siguiente:

Fijado ->(((((GRUPO selected_on [GRADO = 2])

VIAJE1 -> projected_to CVE-GRUPO) joined_to INSCRIPCION)

VIAJE2 -> projected_to CVE-ALUMNO) joined_to ALUMNO)

Final -> projected_to PROMEDIO

La difusión del fijado puede entonces consistir en uno o más viajes, ya que las clases de entidades fijas pueden estar unidas a la clase de entidades solución a través de otras. En tal caso habrá que hacer viajes a clases de entidades intermedias, permitiendo así la difusión del fijado inicial hasta la clase que interesa para resolver la pregunta (clase de entidades solución).

EJEMPLO 5.3.3.

¿Qué profesores imparten matemáticas?

Se empleará ahora una nueva manera de organizar la información de la pregunta:

	CLASE DE ENTIDADES	ATRIBUTO	VALOR
SOLUCION	PROFESOR	NOMBRE	?
FIJAS	MATERIA	NOMBRE	MATEMATICAS

De la tabla anterior se puede obtener el resultado final en álgebra relacional. Se parte de MATERIA, pasando por GRUPO hasta llegar a PROFESOR en el viaje:

Fijado -> (((((MATERIA selected_on [NOMBRE = Matemáticas])

VIAJE1 -> projected_to CVE-MATERIA) joined_to GRUPO)

VIAJE2 -> projected_to CVE-PROFESOR) joined_to PROFESOR)

Final -> projected_to NOMBRE

EJEMPLO 5.3.4.

¿Cuáles fueron los profesores de segundo año de Gómez?

El cuadro de información es el siguiente:

	CLASE DE ENTIDADES	ATRIBUTO	VALOR
SOLUCION	PROFESOR	NOMBRE	?
FIJAS	GRUPO	GRADO	2
	ALUMNO	NOMBRE	GOMEZ

En este caso ya se cuenta con más de una clase de entidades fija. Obviamente, habrá que viajar desde cada una de las clases de entidades fijas hasta la clase de entidades solución. Este es el momento de presentar una definición más.

- INTERSECCION DE VIAJES.- Cuando se proporcionan datos que fijan más de una clase de entidades, es necesario realizar más de un viaje hasta la clase de entidades

solución. Estos viajes tendrán uno o más puntos en común. Al primer punto común entre dos viajes se le denomina intersección de dichos viajes.

Se eligió este nombre debido a su parecido con la intersección de dos caminos. De hecho, la solución (como se verá a continuación) será la operación intersección del álgebra relacional entre las relaciones obtenidas hasta ese punto.

Siguiendo adelante con el ejemplo, se obtiene lo siguiente:

(1) GRUPO -> PROFESOR

```
(((GRUPO selected_on [GRADO = 2])
  projected_to CVE-PROFESOR) joined_to PROFESOR)
 projected_to NOMBRE
```

Esta expresión proporcionará los nombres de todos los profesores de segundo año.

(2) ALUMNO -> PROFESOR

```
((((((ALUMNO selected_on [NOMBRE = Gómez])
      projected_to CVE-ALUMNO) joined_to INSCRIPCION)
     projected_to CVE-GRUPO) joined_to GRUPO)
    projected_to CVE-PROFESOR) joined_to PROFESOR)
   projected_to NOMBRE
```

Esta expresión define una relación con todos los profesores de Gómez.

La solución es, obviamente, la relación que resulta de intersectar la relación obtenida en el viaje (1) con la relación obtenida en el viaje (2).

Sin embargo, resulta evidente que, dado que los caminos tienen varios puntos en común (de hecho el camino (1) es subconjunto del camino (2)), se puede afirmar que la selección aplicada en el camino (1) puede ser aplicada sobre el camino (2) en el momento en que estos se encuentran.

Lo anterior se justifica debido a que el camino (1) realiza una fijación sobre una clase de entidades por la que pasa el otro camino, por lo que se puede hacer esta fijación sobre la misma clase de entidades, pero en el camino (2).

La solución final será entonces:

```
((((((((ALUMNO selected_on [NOMBRE = Gómez])
          projected_to CVE-ALUMNO) joined_to INSCRIPCION)
        projected_to CVE-GRUPO) joined_to GRUPO)

->      selected_on [GRADO = 2])

     projected_to CVE-PROFESOR) joined_to PROFESOR)
   projected_to NOMBRE
```

Lo que define una relación con los nombres de los profesores que dieron clase a Gómez en segundo año, que es la información que se nos había solicitado.

Ejemplo 5.3.3.

¿Quién le enseñó matemáticas a Viciedo?

	CLASE DE ENTIDADES	ATRIBUTO	VALOR
SOLUCION	PROFESOR	NOMBRE	?
FIJAS	MATERIA	NOMBRE	MATEMATICAS
	ALUMNO	NOMBRE	Viciedo

En este ejemplo se presenta una segunda forma de intersección. Si se observan los caminos que unen ALUMNO con PROFESOR y MATERIA con PROFESOR en la figura 5.8, se apreciará que ambos caminos se unen en GRUPO, para luego llegar juntos a la clase de entidades PROFESOR. Sus expresiones en álgebra relacional serán las siguientes:

(1) MATERIA -> PROFESOR

```
(((((MATERIA selected_on [NOMBRE = Matemáticas])
    projected_to CVE-MATERIA) joined_to GRUPO)
   projected_to CVE-PROFESOR) joined_to PROFESOR)
  projected_to NOMBRE
```

(2) ALUMNO -> PROFESOR

```
(((((((ALUMNO selected_on [NOMBRE = Viciedo])
      projected_to CVE-ALUMNO) joined_to INSCRIPCION)
     projected_to CVE-GRUPO) joined_to GRUPO)
    projected_to CVE-PROFESOR) joined_to PROFESOR)
   projected_to NOMBRE
```

Como se verá más tarde, esta es la forma general de la **intersección. Todos los demás tipos de intersección son casos particulares de esta forma.**

La respuesta se obtendrá mediante la intersección de los dos caminos, debido a que (1) define la relación con los nombres de profesores de Matemáticas, y (2) define la relación con los nombres de todos los profesores que dan clase a Viciedo. El procedimiento de solución será el siguiente:

1. Considérese que la clase de clase de entidades donde se realiza la intersección es la clase de entidades solución, y obténganse los caminos hasta ella:

```
(1)     MATERIA -> GRUPO

        (((MATERIA selected_on [NOMBRE = Matemáticas])
         projected_to CVE-MATERIA) joined_to GRUPO)

(2)     ALUMNO -> GRUPO

    (((((ALUMNO selected_on [NOMBRE = Viciedo])
        projected_to CVE-ALUMNO) joined_to INSCRIPCION)
       projected_to CVE-GRUPO) joined_to GRUPO)
```

2. Realícese la intersección de ambas soluciones:

```
    ((((MATERIA selected_on [NOMBRE = Matemáticas])
       projected_to CVE-MATERIA) joined_to GRUPO)
-->  intersect_with
     (((((ALUMNO selected_on [NOMBRE = Viciedo])
         projected_to CVE-ALUMNO) joined_to INSCRIPCION)
        projected_to CVE-GRUPO) joined_to GRUPO))
```

3. Ahora sólo resta hacer el viaje de GRUPO A PROFESOR; esto se logra proyectando el resultado anterior a CVE-PROFESOR y realizando el JOIN correspondiente. Finalmente se realiza la proyección al atributo que resuelve la pregunta (NOMBRE en este caso).

```
((((((MATERIA selected_on [NOMBRE = Matemáticas])
     projected_to CVE-MATERIA) joined_to GRUPO)
   intersect_with
   (((((ALUMNO selected_on [NOMBRE = Viciedo])
       projected_to CVE-ALUMNO) joined_to INSCRIPCION)
      projected_to CVE-GRUPO) joined_to GRUPO))
--> projected_to CVE-PROFESOR) joined_to PROFESOR)
-->projected_to NOMBRE
```

5.4 FORMALIZACION DE LOS CAMINOS POR EL MODELO ELKA.

5.4.1 INTRODUCCION. -

Esta sección tiene como fin el plantear y demostrar la existencia de un número finito de caminos dirigidos a través del modelo ELKA.

Los caminos a través del modelo ELKA son muy útiles en la definición de una expresión en álgebra relacional para resolver una pregunta planteada a una base de datos. Esto se debe (como se vió en la sección anterior) a que los caminos son una forma de expresar las relaciones entre las clases de entidades del esquema. Esto significa que el esquema tiene un gran contenido semántico; de hecho es tan

grande como se haya decidido en su diseño.

Si se tiene el esquema de una base de datos determinada, se podrá averiguar la semántica contenida en ella. Esta semántica está expresada en forma de relaciones, que a su vez están compuestas de una o más ligas básicas del modelo ELKA.

Por lo tanto, si los caminos a través del esquema tienen contenido semántico, será necesario recorrerlos con el fin de resolver una pregunta cualquiera. Es decir, que la pregunta formulada tiene un contenido semántico que debe ser compatible con el contenido semántico de algún camino por el esquema.

Surge entonces la necesidad de expresar formalmente estos caminos, así como de desarrollar un modelo que los contenga de manera sencilla y accesible. El resto de esta sección utiliza la notación y los formalismos elementales establecidos en "The ELKA model approach to the design of database conceptual models" [ROD81], con el fin de sentar las bases para exponer los conceptos de "camino", "travesía" y "matriz de caminos", desarrollados para los fines de esta tesis.

5.4.2 NOTACION Y DEFINICIONES. -

5.4.2.1 ALGUNAS FUNCIONES DEL MODELO ELKA. -

A continuación se dará la definición de algunas funciones contenidas en el modelo conceptual ELKA.

Sea LC el conjunto definido por las clases de ligas.

Sea E el conjunto de todas las clases de entidades comprendidas en el modelo.

Entonces:

- FE(x). Es una función que, siendo x $\in$ LC, entrega la clase de entidades que se encuentra **al frente** de la liga x ("Front Entity").

- BE(x). Es una función que, siendo x $\in$ LC, entrega la clase de entidades que se encuentra **atrás** de la liga x ("Back Entity").

- FA(x). Es una función que, siendo x $\in$ LC, entrega los atributos de relación del **frente** de la liga.

- BA(x). Es una función que, siendo x $\in$ LC, entrega los atributos de relación de **atrás** de la liga.

5.4.2.2 DEFINICIONES. -

- **ENTIDADES CONECTADAS.-** Se dice que dos clases de entidades E1 y E2 están conectadas a través de una liga $x \in LC$ si y solo si:

  E1 = FE(x) y E2 = BE(x) ó

  E1 = BE(x) y E2 = FE(x)

- CAMINO.- Un **camino** C es la secuencia de ligas $\langle x_1, x_2, \ldots, x_n \rangle$ donde $x_1, x_2, \ldots, x_n \in$ LC y n > 1 o n = 1, tal que

  $E_o \text{ --}x_1\text{--> } E_1 \text{ --}x_2\text{--> } E_2 \ldots\text{--}x_n\text{--> } E_n$

  o sea:

  E_{i-1} y E_i están conectadas a través de x_i

  para i = 1, 2, ..., n

  La **longitud** del camino será n. Un camino de longitud 1 es una liga.

- TRAVESIA DE UNA LIGA.- Sea $x \in$ LC. Si E_1 y E_2 están conectadas a través de x, entonces existen cero, uno, o más tuplos $t_{E1} \in E_1$ tales que:

  si E_1 = BE(x) y E_2 = FE(x) entonces

$$t_{E1}(BA(x)) = t_{E2}(FA(x))$$

la notación:

$$t_{E1} = x(t_{E2}) \quad \text{ó} \quad t_{E1} = t_{E2} \cdot x$$

será la empleada para denotar la travesía de una liga x desde

t_{E1} hasta t_{E2}.

- TRAVESIA INVERSA DE UNA LIGA.- Sea $x \in LC$. Si E_1 y E_2 están conectados a través de x, entonces existen cero, uno o más tuplos $t_{E1} \in E_1$ tales que:
si $E_1 = FE(x)$ y $E_2 = BE(x)$ entonces

$$t_{E1}(FA(x)) = t_{E2}(BA(x))$$

la notación:

$$t_{E1} = x'(t_{E2}) \quad \text{ó} \quad t_{E1} = t_{E2} \cdot x'$$

será la empleada para denotar la travesía inversa de una liga x
desde t_{E1} hasta t_{E2}.

- **TRAVESIA DE UN CAMINO.- Es una secuencia de travesias de ligas tal que, si c es un camino compuesto por** $\langle x_1, x_2, \ldots, x_n \rangle$, E_o es la entidad de inicio de la travesia y E_n es la entidad de destino, entonces:

Dado un tuplo $T_o \in E_o$, existen cero, uno o más tuplos $t_n \in E_n$ donde:

$$t_n = \mathbf{c}(t_o)$$
$$t_n = x_n(x_{n-1}\ldots x_2(x_1(t_o))\ldots)$$
$$t_n = x_1 \cdot x_2 \cdot \ldots \cdot x_n(t_o)$$

ó

$$t_n = t_o\mathbf{c} = t_o \cdot x_1 \cdot x_2 \cdot \ldots \cdot x_n$$

Nótese que para poder accesar el tuplo t_n, será necesario accesar las clases de entidades intermedias $E_1, E_2, \ldots, E_{n-1}$.

5.4.3 LA MATRIZ DE CAMINOS. -

La matriz de caminos **C** es una estructura que permite almacenar los caminos que unen las clases de entidades entre si. Los elementos de la matriz de caminos representan caminos a través del modelo ELKA.

Una matriz de ligas es una matriz de caminos construida de la siguiente forma:

Sean E_i, $E_j \in E$, entonces:

$$C(E_i, E_j) = x_1+x_2+\ldots+x_n$$

si y sólo si:

$$x_1, x_2, \ldots, x_n \in LC$$

y las clases de entidades E_i y E_j están ligadas por las ligas x_k para $k = 1, 2, \ldots, n$

de otra manera:

$$C(E_i, E_j) = \emptyset$$

Un elemento de la matriz representa todas las ligas que unen a la clase de entidades E_i con la clase de entidades E_j. Si el elemento $C(E_i, E_j)$ contiene dos o más ligas, se tratará de **ligas paralelas.**

El símbolo "+" puede ser leído como "o". Los elementos de la matriz C serán denominados **expresiones de caminos.** C es una matriz cuadrada de rango $|E|$.

Las expresiones de caminos (expresiones-c en lo subsiguiente) se definen de la siguiente forma.

Una expresión-c es:

a) El nombre de una liga, o

b) El nombre de una liga inversa, o bien

c) Si x, y son expresiones-c, entonces:

- $x \cdot y$ es una expresión-c si y sólo si:

$$FE(x) = BE(y)$$

- $x+y$ es una expresión-c si y sólo si:

$$FE(x) = FE(y) \text{ y } BE(x) = BE(y)$$

Como en el caso del "+", que significa "o", el simbolo punto ($\cdot$) significa "es seguido en la secuencia por". Así, $x \cdot y$ puede ser leído como "x es seguido en la secuencia por y".

Las siguientes reglas sintácticas definen los operadores "." (punto) y "+" (mas):

Sean x, y, z expresiones-c válidas, entonces:

- **Reglas del "·":**

  - **Distribución:**

    $$x \cdot (y+z) = x \cdot y + x \cdot z$$

    $$(y+z) \cdot x = y \cdot x + z \cdot x$$

  - Cancelación:

    $$x \cdot \emptyset = \emptyset \cdot x = \emptyset$$

- Reglas del "+":

  - Conmutativa:

    $$x+y = y+x$$

  - Absorción y duplicación:

    $$x+x = x$$

    $$x+\emptyset = \emptyset+x = x$$

    $$\emptyset+\emptyset = \emptyset$$

De hecho, la regla de distribución del "·" es particularmente útil para factorizar colas o cabezas comúnes de caminos (en intersecciones).

La enésima potencia de C denotada C^n se define como sigue:

a) C^1 es C

b) C^{n+1} es $C^n \cdot C$ para $n > 1$ o $n = 1$

$C^n(E_i, E_j)$ representa todos los caminos de longitud n que unen E_i con E_j.

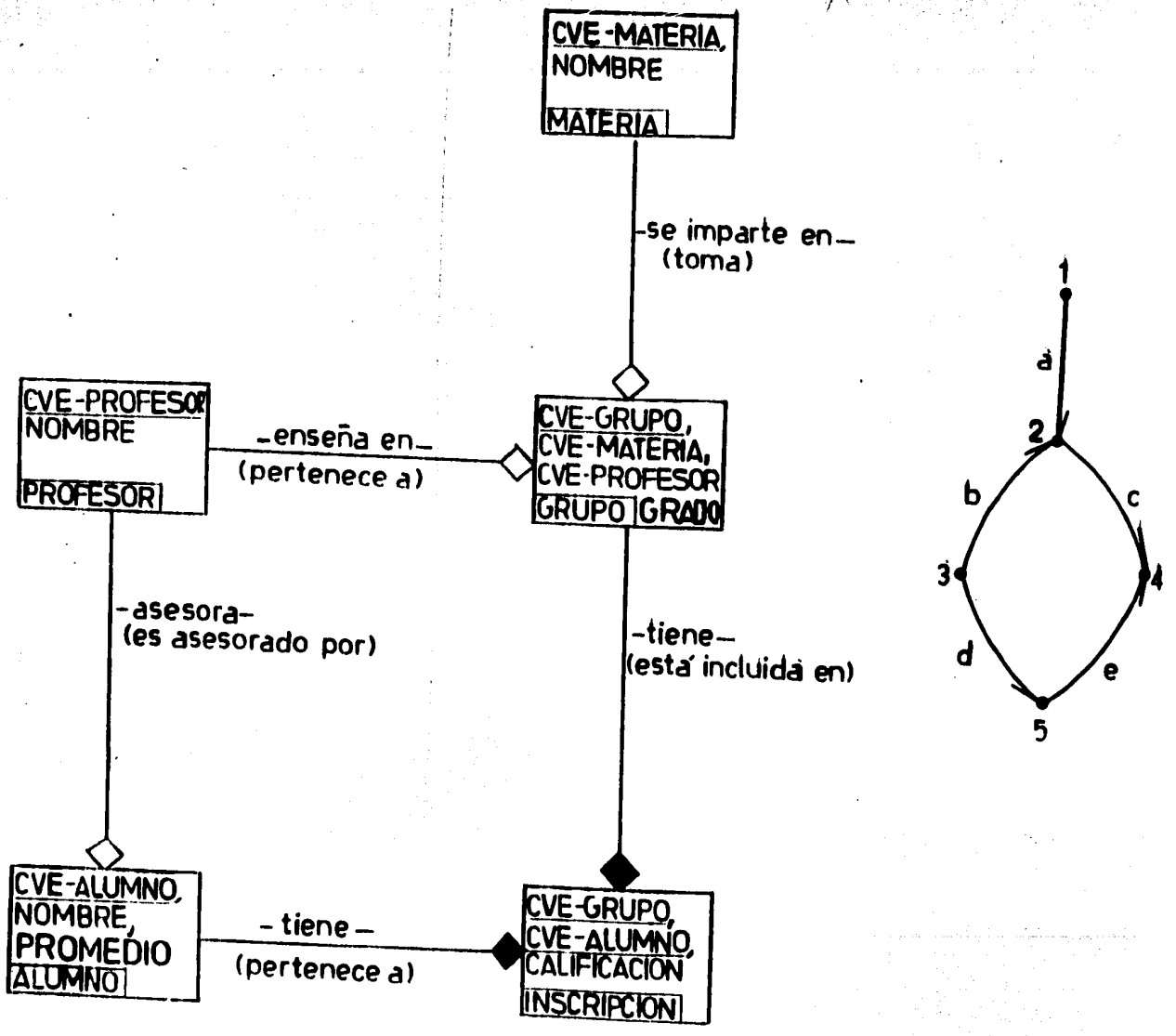
La matriz C^+ se define de la siguiente forma:

$$C^+ = \sum_{i=1}^{|E|} C^i$$

Entonces $C^+(E_i, E_j)$ representa los caminos de longitud mayor o igual a 1 que unen E_i con E_j. En este trabajo no se toman en cuenta los ciclos de longitud mayor que 1, o sea que ningún camino de longitud mayor que 1 podrá tener secuencias de ligas repetidas (más adelante se justificará esta restricción -al final del capitulo y en el segundo ejemplo de la sección complementaria del apéndice B-).

EJEMPLO 5.4.1.

Tomemos el modelo ELKA de la figura 5.9 a), y se tratará de encontrar su matriz de caminos. Para lograrlo, se construye una gráfica inversa del mismo (que se muestra en la figura 5.9 b).



a) Modelo ELKA del ejemplo. b) Su gráfica inversa.

FIGURA 5.9.- Modelo ELKA del ejemplo 5.4.1.

En la gráfica inversa, los nodos representan a las clases de entidades, y los arcos representan ligas (dirigidos en sentido inverso a ellas). Para realizarla, se emplearon los siguientes mapeos:

Entidades:

MATERIA = 1 , GRUPO = 2 , PROFESOR = 3 ,

INSCRIPCION = 4 , ALUMNO = 5

Ligas:

MATERIA.se-imparte-en.GRUPO = a

PROFESOR.enseña-en.GRUPO = b

GRUPO.tiene.INSCRIPCION = c

PROFESOR.asesora.ALUMNO = d

ALUMNO.tiene.INSCRIPCION = e

Ligas inversas:

GRUPO.toma.MATERIA = a'

GRUPO.pertenece-a.PROFESOR = b'

INSCRIPCION.esta-incluida-en.GRUPO = c'

ALUMNO.es-asesorado-por.PROFESOR = d'

INSCRIPCION.pertenece-a.ALUMNO = e'

La matriz de caminos de primer grado (C) está constituida por las ligas del modelo empleado, y se muestra en la página siguiente, junto con las matrices de caminos de longitud 2, 3, 4 y 5. La matriz de caminos de longitud 5 es cero, debido a que no existen caminos de longitud 5 (compuestos por 5 ligas) que no sean cíclicos. La matriz C^+ se muestra en la siguiente página.

$$\mathbf{C} = \begin{bmatrix} 0 & a & 0 & 0 & 0 \\ a' & 0 & b' & c & 0 \\ 0 & b & 0 & 0 & d' \\ 0 & c' & 0 & 0 & e' \\ 0 & 0 & d' & e & 0 \end{bmatrix}$$

$$\mathbf{C}^2 = \begin{bmatrix} 0 & 0 & a \cdot b' & a \cdot c & 0 \\ 0 & 0 & 0 & 0 & b' \cdot d' + c \cdot e' \\ b \cdot a' & 0 & 0 & b \cdot c + d \cdot e & 0 \\ c' \cdot a' & 0 & c' \cdot b' + e' \cdot d' & 0 & 0 \\ 0 & d' \cdot b + e \cdot c' & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{C}^3 = \begin{bmatrix} 0 & 0 & 0 & 0 & a \cdot b' \cdot d' + a \cdot c \cdot e' \\ 0 & 0 & c \cdot e' \cdot d' & b' \cdot d \cdot e & 0 \\ 0 & d \cdot e \cdot c' & 0 & 0 & b \cdot c \cdot e' \\ 0 & e' \cdot d' \cdot b' & 0 & 0 & e' \cdot b' \cdot d \\ d' \cdot b \cdot a' + e \cdot c' \cdot a' & 0 & e \cdot c' \cdot b' & d' \cdot b \cdot c & 0 \end{bmatrix}$$

$$\mathbf{C}^4 = \begin{bmatrix} 0 & 0 & a \cdot c \cdot e' \cdot d' & a \cdot b' \cdot d \cdot e & 0 \\ 0 & 0 & 0 & 0 & 0 \\ d \cdot e \cdot c' \cdot a' & 0 & 0 & 0 & 0 \\ c' \cdot d' \cdot b \cdot a & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

$$\mathbf{C}^5 = \mathbf{0}$$

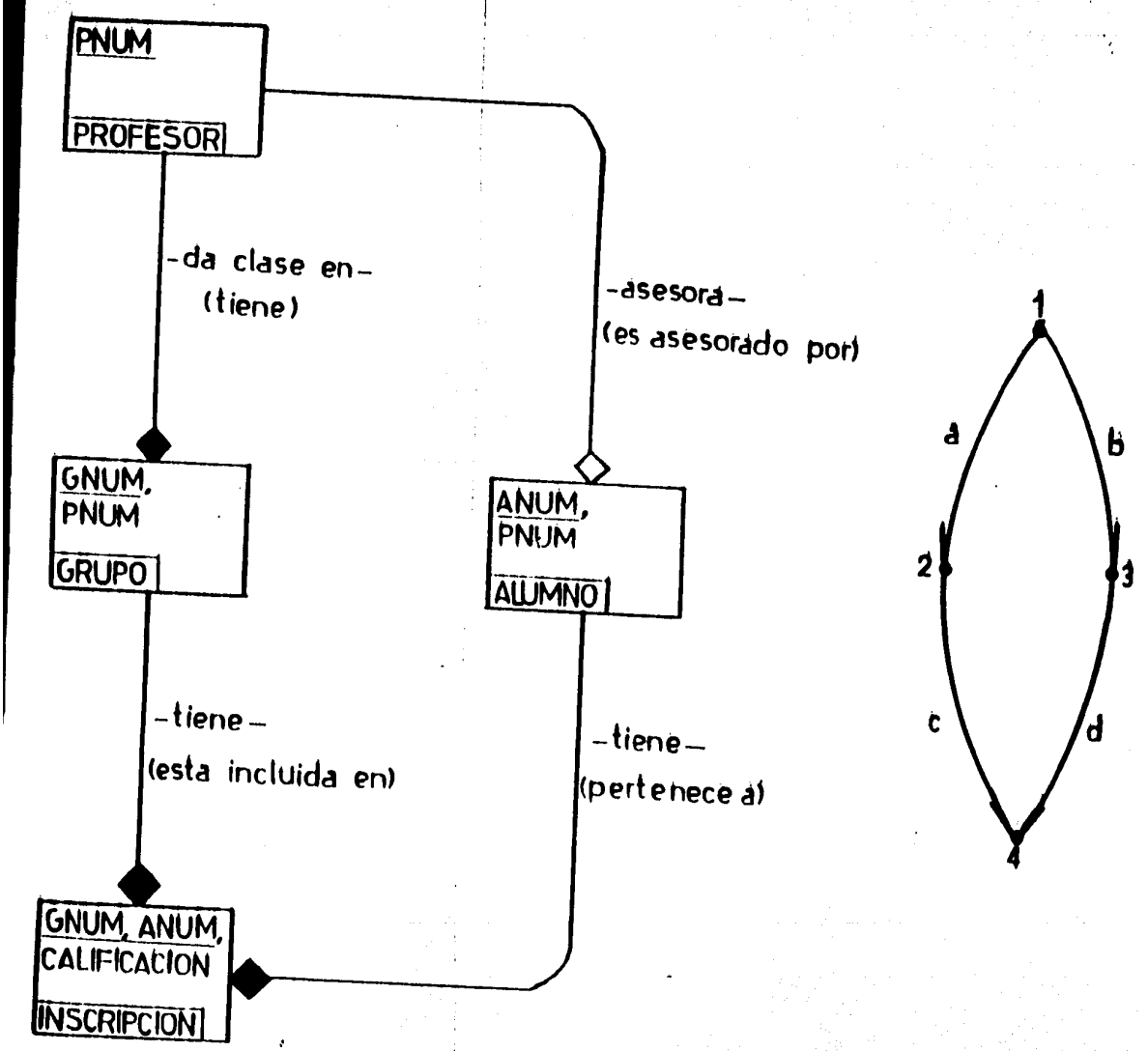
$$= \begin{bmatrix} 0 & a & a \cdot b' + a \cdot c \cdot e' \cdot d' & a \cdot c + a \cdot b' \cdot d \cdot e & a \cdot b' \cdot d + a \cdot c \cdot e' \\ a' & 0 & b' + c \cdot e' \cdot d' & c + b' \cdot d \cdot e & b' \cdot d + c \cdot e' \\ b \cdot a' + d \cdot e \cdot c' \cdot a' & b + d \cdot e \cdot c' & 0 & b \cdot c + d \cdot e & d + b \cdot c \cdot e' \\ c' \cdot a' + c' \cdot d' \cdot b \cdot a' & c' + e' \cdot d' \cdot b & c' \cdot b' + e' \cdot d' & 0 & e' + c' \cdot b' \cdot d \\ d \cdot b \cdot a' + e \cdot c \cdot a' & d' \cdot b + e \cdot c' & d' + e \cdot c' \cdot b' & e + d' \cdot b \cdot c & 0 \end{bmatrix}$$

EJEMPLO 5.4.2.

Se empleará ahora el modelo de la figura 5.10. La semántica de éste modelo es la siguiente:

En una escuela, cada profesor, enseña en uno o más grupos. Cada grupo tiene inscritos uno o más alumnos. Cada alumno puede estar inscrito en uno o más grupos. Además, un profesor asesora a cero, uno o más alumnos.

En la figura 5.10 b) se muestra su gráfica inversa.



a) Modelo ELKA de ejemplo. b) Su gráfica inversa.

FIGURA 5.10.- Modelo ELKA del ejemplo 5.4.2.

Los mapeos realizados son los siguientes:

Entidades:

PROFESOR = 1 , GRUPO = 2 , ALUMNO = 3 ,

INSCRIPCION = 4

Ligas:

PROFESOR.enseña-en.GRUPO = a

PROFESOR.asesora.ALUMNO = b

GRUPO.tiene.INSCRIPCION = c

ALUMNO.tiene.INSCRIPCION = d

Ligas inversas:

GRUPO.pertenece-a.PROFESOR = a'

ALUMNO.es-asesorado-por.PROFESOR = b'

INSCRIPCION.esta-incluida-en.GRUPO = c'

INSCRIPCION.pertenece-a.ALUMNO = d'

La matriz de caminos correspondiente será:

$$C^{+} = \begin{bmatrix} 0 & a + b \cdot b \cdot d \cdot ct & b + a \cdot c \cdot dt & a \cdot c + b \cdot d \\ at + c \cdot dt \cdot bt & 0 & at \cdot b + c \cdot dt & c + at \cdot b \cdot d \\ bt + d \cdot ct \cdot at & bt \cdot a + d \cdot ct & 0 & d + bt \cdot a \cdot c \\ ct \cdot at + dt \cdot bt & ct + dt \cdot bt \cdot a & dt + ct \cdot at \cdot b & 0 \end{bmatrix}$$

5.5 EL CONTENIDO SEMANTICO DEL MODELO.

5.5.1 INTRODUCCION. -

Hasta ahora se ha utilizado la información contenida en la pregunta que se refiere a objetos de la base de datos y se ha encontrado la manera de emplear dicha información en la elaboración de una expresión en álgebra relacional que defina una relación con los datos que contestan a la pregunta original. Sin embargo, sólo se han empleado aquellas partes de la pregunta que definen las clases de entidades fijas y la clase de entidades solución.

Esta información ha ayudado a conocer los puntos de partida y el punto de llegada de los viajes que a realizar por el modelo para formular la expresión de salida.

Tal parece que estos son los únicos datos necesarios para realizar la travesía. Sin embargo no es así, ya que puede haber más de un camino uniendo a las clases de entidades involucradas en la pregunta, como se puede

apreciar en la matriz de caminos C^+. Estos caminos alternos se reflejan en la matriz como caminos paralelos unidos por una disyunción "+", de la siguiente forma:

$$C^+(E_i, E_j) = c_1+c_2+\ldots+c_n$$

Donde:

$C^+(E_i, E_j)$ define un elemento de la matriz C^+.

c_i es un camino que une E_i con E_j para $i = 1, 2, \ldots, n$ y $j = 1, 2, \ldots, n$.

Surge entonces la necesidad de encontrar alguna información en la pregunta original que ayude a elegir el camino correcto.

El objetivo de esta sección es el de exponer la manera de discriminar entre los caminos paralelos que unen a las clases de entidades en cuestión para encontrar aquel que, al seguirlo, nos proporcione la información correcta para contestar la pregunta formulada.

5.5.2 LA MATRIZ SEMANTICA. -

Para poder comprender el sentido de la matriz semántica, se desarrollará el problema que se intenta resolver con ella. Recordemos el modelo ELKA del ejemplo 2 (página 183).

Los mapeos realizados fueron los siguientes:

Entidades:

PROFESOR = 1 , GRUPO = 2 , ALUMNO = 3 ,

INSCRIPCION = 4

Ligas:

PROFESOR.enseña-en.GRUPO = a

PROFESOR.asesora.ALUMNO = b

GRUPO.tiene.INSCRIPCION = c

ALUMNO.tiene.INSCRIPCION = d

Ligas inversas:

GRUPO.pertenece-a.PROFESOR = a'

ALUMNO.es-asesorado-por.PROFESOR = b'

INSCRIPCION.esta-incluida-en.GRUPO = c'

INSCRIPCION.pertenece-a.ALUMNO = d'

La matriz de caminos correspondiente fue:

$$C^{+} = \begin{bmatrix} 0 & a + b \cdot b \cdot d \cdot c\prime & b - a \cdot c \cdot d\prime & a \cdot c + b \cdot d \\ a\prime + c \cdot d\prime \cdot b\prime & 0 & a\prime \cdot b - c \cdot d\prime & c + a\prime \cdot b \cdot d \\ b\prime + d \cdot c\prime \cdot a\prime & b\prime \cdot a + d \cdot c\prime & 0 & d + b\prime \cdot a \cdot c \\ c\prime \cdot a\prime + d\prime \cdot b\prime & c\prime + d\prime \cdot b\prime \cdot a & d\prime - c\prime \cdot a\prime \cdot b & 0 \end{bmatrix}$$

tomemos ahora un elemento de esta matriz, digamos

C+(1,3) = b+a·c·d'

Se puede ver que la clase de entidades PROFESOR está unida a la clase de entidades ALUMNO por dos caminos distintos. Examinando las ligas y su significado, y desglosando los caminos mediante el mapeo realizado, se llega a lo siguiente:

(1) b = PROFESOR.asesora.ALUMNO

(2) a·c·d' = PROFESOR.enseña-en.GRUPO.
.tiene.INSCRIPCION.pertenece-a.ALUMNO

Lo anterior pone de manifiesto que cada uno de los caminos paralelos tiene implicaciones semánticas diferentes.

Mientras que el camino (1) implica la relación en que el profesor asesora al alumno, el camino (2) implica la relación en que el profesor enseña en un grupo al alumno inscrito en él.

Por lo tanto, si la pregunta formulada es:

¿Quiénes dan clase al alumno J.P.?

el camino a seguir será el camino (2).

Por otro lado, si la pregunta es:

¿Quién asesora al alumno J.P.?

habrá que seguir el camino (1).

Se puede afirmar que **el verbo de la pregunta dá la pauta para elegir el camino a seguir entre dos clases de entidades.**

De hecho, las ligas en un modelo ELKA se acompañan de un verbo y, algunas veces, de una frase verbal.

Se puede afirmar lo siguiente:

- La validez semántica de una pregunta, en el contexto de un modelo de datos, depende de su compatibilidad con la semántica de dicho modelo.

- La semántica de una pregunta basada en el verbo que enlaza la referencia a la entidad solución con las referencias a las entidades fijas, será útil para determinar el camino a seguir para formular la expresión final en álgebra relacional.

La primera afirmación supone que la semántica del esquema es suficiente para determinar si una pregunta está correctamente formulada o no, a menos que haya un enlace semántico puro que no se encuentre explícito en el mismo. Un ejemplo de tal tipo de enlace es el siguiente:

MAESTRO.enseña.ALUMNO

ya que este enlace está implícito en el modelo anterior, mediante el camino (2) de los analizados con anterioridad.

La validez de una pregunta formulada estará entonces fuertemente ligada al contexto definido por la base de datos. De ninguna manera se intenta decir que una pregunta que no sea correcta en este ámbito sea incorrecta en el

sentido más absoluto. Es decir, que si la pregunta formulada a la base de datos está fuera de su área de conocimientos, entonces será considerada como semánticamente inválida.

De la segunda afirmación se desprende la necesidad de definir una estructura que contenga la información semántica involucrada en cada camino del modelo. Esta estructura debe ser compatible (por razones de manejo) con la estructura en que se almacenan los caminos en sí (la matriz C^+). Es por todo lo anterior que se decidió definir la **matriz semántica** de un esquema.

5.5.2.1 DEFINICIONES. -

- ENLACE SEMANTICO.- Es la relación de significado representada explícita o implícitamente en el esquema, y que se establece explícitamente entre entidades conectadas por medio de ligas, e implícitamente entre objetos del mundo representados en el esquema. En otras palabras, si la liga que une dos clases de entidades en el esquema expresa de manera explícita su relación semántica, este se denominará **enlace semántico** simplemente.

- **ENLACE SEMANTICO PURO.- Una base de datos es un modelo de objetos del mundo y la manera en que se relacionan. Existen casos en que una relación semántica compuesta** (por ejemplo GRUPO tiene INSCRIPCION que pertenece a ALUMNO) tiene una forma más simple y usual (GRUPO tiene ALUMNO). A ésta forma simplificada de expresar una relación semántica compuesta será denominada **relación semántica pura.**

5.5.2.2 DEFINICION DE LA MATRIZ SEMANTICA. -

La matriz semántica es la estructura que contiene la información semántica de los diferentes caminos a través del esquema (expresados en la matriz C^+).

En otras palabras, la matriz semántica contiene los enlaces semánticos entre las diferentes clases de entidades del modelo.

La **matriz semántica** S se define como una matriz con las siguientes caractaristicas:

Sea S(i,j) un elemento de la matriz S. Entonces:

$$S(i,j) = E_i.enlace.E_j$$

Donde:

E_i, E_j **$\in$ E son las entidades de partida y de llegada del camino cuya semántica se expresa en enlace.**

enlace es:

- Un verbo que expresa la conexión semántica entre las entidades a sus extremos, o
- .enlace.E_n.enlace. en el caso de que haya entidades intermedias en la implicación semántica.

5.5.2.3 SEMANTICA DIRECTA Y SEMANTICA INVERSA -

Basados en la información del modelo, se pueden plantear dos preguntas básicas sobre, por ejemplo la relación entre la clase de entidades MATERIA y la clase de entidades GRUPO:

(1) ¿Qué materia toma el grupo 3001?

(2) ¿Qué materia se imparte en el grupo 3001?

Como se puede apreciar, ambas preguntas exigen el mismo viaje para ser resueltas. Tal viaje es justamente el descrito en:

$$C^{+}(2,1) = a'$$

debido a que "grupo" es la entidad fija y MATERIA es la entidad solución en ambas preguntas.

Por otro lado, según la definición de MATRIZ SEMANTICA:

S(2,1) = GRUPO.toma.MATERIA.

S(1,2) = MATERIA.se imparte en.GRUPO.

Se dan las siguientes relaciones:

PREGUNTA	MATRIZ SEMANTICA	VIAJE A REALIZAR
(1)	S(2,1)	$C^{+}(2,1)$
(2)	S(1,2)	$C^{+}(2,1)$

Debido a lo anterior, ya que en la pregunta (1):

$$S(i,j) \Rightarrow C^{+}(i,j)$$

y en la pregunta (2):

$$S(j,i) \Rightarrow C^{+}(i,j)$$

se afirma que la pregunta (1) tiene una **semántica directa.**

Por otro lado, la pregunta (2) tiene semántica inversa.

Es necesario distinguir ambos casos, ya que ello ayudará a elegir el camino correcto a través del modelo.

Si se analizan más de cerca las preguntas (1) y (2), se podrá apreciar que la diferencia básica entre ellas es el lugar del sujeto (además, oviamente, del verbo):

En la pregunta (1) el sujeto es LA ENTIDAD FIJA.

En la pregunta (2) el sujeto es LA ENTIDAD SOLUCION.

De hecho, esta diferencia da la clave para saber qué tipo de semántica se emplea. Esto es lógico, puesto que la semántica de un modelo es dirigida; por ejemplo, si se viaja de MATERIA a GRUPO, la semántica de este viaje será:

MATERIA.se imparte en.GRUPO

Por otra parte, el viaje inverso tendrá la siguiente semántica:

GRUPO.toma.MATERIA

Lo anterior se relaciona con una pregunta de la siguiente manera:

Supongamos que:

Ef = Entidad fija

Es = Entidad solución

- **Para solucionar la pregunta hay que viajar de Ef a Es.**

- **La semántica del viaje Ef->Es será de la siguiente forma:**

  S(f,s) = Ef.conexión.Es

- Esta semántica tiene como sujeto de la acción verbal a Ef.

- Por lo tanto:

  - Si la pregunta tiene como sujeto a Ef, se estará usando la semántica:

    S(f,s)

    que implicará el viaje:

    C+(f,s) [SEMANTICA DIRECTA]

  - Si la pregunta tiene como sujeto a Es, se estará empleando la semántica:

    S(s,f)

que implicará además el mismo viaje:

C+(f,s) [SEMANTICA INVERSA]

5.5.3 EJEMPLO. -

Del modelo de la figura 5.9 se tienen los siguientes datos:

La semántica de las ligas es la siguiente:

MATERIA.se-imparte-en.GRUPO = a

PROFESOR.enseña-en.GRUPO = b

GRUPO.tiene.INSCRIPCION = c

PROFESOR.asesora.ALUMNO = d

ALUMNO.tiene.INSCRIPCION = e

GRUPO.toma.MATERIA = a'

GRUPO.pertenece-a.PROFESOR = b'

INSCRIPCION.esta-incluida-en.GRUPO = c'

ALUMNO.es-asesorado-por.PROFESOR = d'

INSCRIPCION.pertenece-a.ALUMNO = e'

A continuación se muestra una tabla con los valores de los elementos de la matriz semántica y su equivalencia con los caminos de la matriz de caminos.

i	j	S(i,j)	camino	C
1	1		0	
1	2	MATERIA.se-imparte-en.GRUPO	a	
1	3	MATERIA.se-imparte-por.PROFESOR	a·b′	X
		MATERIA.se-imparte-a.ALUMNO. .asesorado-por.PROFESOR	a·c·e′·d′	
1	4	MATERIA.se-imparte-en.GRUPO. .tiene.INSCRIPCION	a·c	
		MATERIA.se-imparte-por.PROFESOR. .asesora.ALUMNO.tiene.INSCRIPCION	a·b′·d·e	
1	5	MATERIA.se-imparte-por.PROFESOR. .asesora.ALUMNO	a·b′·d	
		MATERIA.se-imparte-a.ALUMNO	a·c·e′	X
2	1	GRUPO.toma.MATERIA	a′	
2	2		0	
2	3	GRUPO.pertenece-a.PROFESOR	b′	
		GRUPO.tiene.ALUMNO. .es-asesorado-por.PROFESOR	c·e′·d′	
2	4	GRUPO.tiene.INSCRIPCION	c	
		GRUPO.pertenece-a.PROFESOR. .asesora.ALUMNO.tiene.INSCRIPCION	b′·d·e	
2	5	GRUPO.pertenece-a.PROFESOR. .asesora.ALUMNO	b′·d	
		GRUPO.tiene.ALUMNO	c·e′	X
3	1	PROFESOR.ensena.MATERIA	b·a′	X
		PROFESOR.asesora.ALUMNO. .toma.MATERIA	d·e·c′·a′	
3	2	PROFESOR.ensena-en.GRUPO	b	
		PROFESOR.asesora.ALUMNO. .inscrito-en.GRUPO	d·e·c′	

3	3		0	
3	4	PROFESOR.ensena-en.GRUPO. .tiene.INSCRIPCION	b·c	
		PROFESOR.asesora.ALUMNO. .tiene.INSCRIPCION	d·e	
3	5	PROFESOR.asesora.ALUMNO	d	
		PROFESOR.ensena-a.ALUMNO	b·c·e'	X
4	1	INSCRIPCION.pertenece-a.ALUMNO. .es-asesorado-por.PROFESOR. .ensena-en.GRUPO.toma.MATERIA	e'·d'·b·a'	
		INSCRIPCION.esta-incluida-en.GRUPO .toma.MATERIA	c'·a'	
4	2	INSCRIPCION.pertenece-a.ALUMNO. .es-asesorado-por.PROFESOR. .ensena-en.GRUPO	e'·d'·b	
		INSCRIPCION.esta-incluida-en.GRUPO	c'	
4	3	INSCRIPCION.esta-incluida-en.GRUPO. .pertenece-a.PROFESOR	c'·b'	
		INSCRIPCION.pertenece-a.ALUMNO. .es-asesorado-por.PROFESOR	e'·d'	
4	4		0	
4	5	INSCRIPCION.esta-incluida-en.GRUPO. .pertenece-a.PROFESOR. .asesora.ALUMNO	c'·b'·d	
		INSCRIPCION.pertenece-a.ALUMNO	e'	
5	1	ALUMNO.es-asesorado-por.PROFESOR. .ensena.MATERIA	d·b·a'	
		ALUMNO.toma.MATERIA	e·c'·a'	X
5	2	ALUMNO.es-asesorado-por.PROFESOR. .ensena-en.GRUPO	d'·b	
		ALUMNO.pertenece-a.GRUPO	e·c'	X
5	3	ALUMNO.tiene.PROFESOR	e·c'·b'	X

		ALUMNO.es-asesorado-por.PROFESOR	**d'**	
5	**4**	**ALUMNO.tiene.INSCRIPCION**	**e**	
		ALUMNO.es-asesorado-por.PROFESOR. .ensena-en.GRUPO.tiene.INSCRIPCION	**d'·b·c**	
5	5		0	

Los elementos marcados con una X en la última columna son enlaces semánticos puros, lo que significa que un camino compuesto tiene una implicación semántica distinta -o abreviada- a la de la compuesta por las semánticas de sus ligas. Estos son los **enlaces semánticos puros**. Por ejemplo:

En el elemento (5,3) de la tabla, la semántica compuesta de todas las ligas es la siguiente:

ALUMNO.tiene.INSCRIPCION.esta-incluida-en.GRUPO.

.pertenece-a.PROFESOR

se convierte en:

ALUMNO.tiene.PROFESOR

En la siguiente figura se esquematizan las siguientes ligas semánticas puras que, por supuesto, no son explícitas en el modelo.

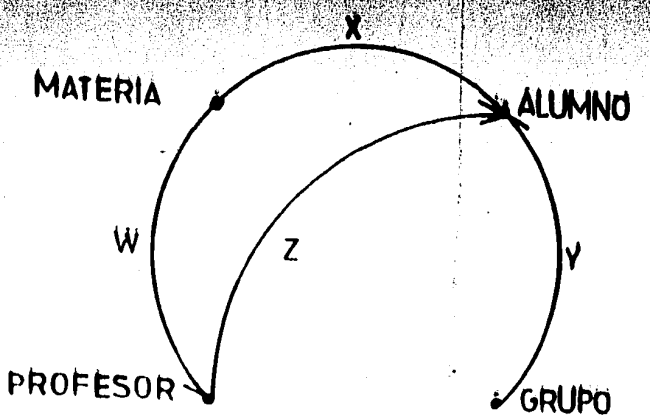


FIGURA 5.11.- Ligas semánticas puras en el modelo.

Las equivalencias son las siguientes:

w = MATERIA.se-imparte-por.PROFESOR
w' = PROFESOR.enseña.MATERIA
x = MATERIA.se-imparte-a.ALUMNO
x' = ALUMNO.toma.MATERIA
y = GRUPO.tiene.ALUMNO
y' = ALUMNO.pertenece-a.GRUPO
z = PROFESOR.enseña-a.ALUMNO
z' = ALUMNO.tiene.PROFESOR

5.6 EL ANALIZADOR SEMANTICO.

En el presente capitulo se ha expuesto la teoria desarrollada para el diseño del analizador semántico de una interfaz que permita al usuario interrogar sobre una base de datos en lenguaje natural.

Los conceptos principales fueron:

- Viajes por un modelo de datos.

- La matriz de caminos C^+.

- La matriz semántica S.

Estos tres conceptos son fundamentales en la elaboración de una definición en álgebra relacional de la pregunta de entrada.

La información necesaria para poder elaborar entonces la expresión de salida será la siguiente:

- Entidades fijas definidas en la pregunta.

- Entidad solución definida por la pregunta.

- Verbo que implica la relación semántica entre la entidad fija y entidades solución (que es el verbo usado en la pregunta).

Los elementos con los que contará el analizador semántico para realizar su tarea son las dos matrices definidas arriba. Con la ayuda de la matriz semántica se podrán relacionar las entidades solución, entidad fija, y verbo de la pregunta, con los viajes a realizar por el esquema. Dichos viajes se expresarán en términos de las operaciones del álgebra relacional con el fin de que la expresión obtenida pueda ser confrontada finalmente contra

la base de datos, y así obtener la solución a la pregunta.

La matriz de caminos del esquema se puede obtener mediante un procesamiento de la información referente a relaciones contenida en el esquema, evitando los ciclos de más de una liga. Los ciclos de una sóla liga pueden tener implicaciones semánticas importantes, por lo que no se descartarán; sin embargo, los ciclos de más de una liga pueden ser expresados mediante semánticas compuestas. Debido a que el prototipo realizado tiene la capacidad de manejar semánticas compuestas, no es necesario incluir los ciclos en la matriz de caminos (ver apéndice B, en su sección complementaria).

La matriz semántica de un esquema, puede ser obtenida a partir del mismo y mediante una interacción con el diseñador de la base de datos con el fin de definir las relaciones semánticas no explícitas en el esquema. Estas relaciones semánticas se expresarán en forma de un verbo, y tendrán como implicación todo un viaje compuesto por el modelo.

Por lo tanto, es posible construir un analizador semántico sobre estas bases, confiando que sus fuentes de información se pueden obtener a partir de:

- Procesamiento de la información del esquema para definir la matriz de caminos y la matriz semántica.

- **Análisis sintáctico que resuelva las referencias a las clases de entidades del modelo con ayuda de un diccionario de sinónimos, y mediante nombres de roles, que asignen complementos de verbos a partes de la base de datos. El verbo pasará de manera prácticamente directa a esta etapa del análisis.**

El programa realizado se basa en los conceptos anteriores para, con ayuda de unas estructuras equivalentes a la matriz de caminos y a la matriz semántica, construir una expresión final en álgebra relacional; cuenta además con otra fuente de información, que consiste en una estructura en la que se almacenan las equivalencias, en álgebra relacional, de la travesía de una liga. Como se había observado, la travesía de una liga implicaba las siguientes operaciones:

1. Una proyección al atributo de relación sobre la clase de entidades de partida.
2. Un join con la clase de entidades de llegada.

Esta fuente de información se puede generar de manera automática, si se toma en cuenta que la información necesaria sobre la liga entre dos clases de entidades es sólamente el nombre del atributo involucrado en esta relación para cada una de las clases a los extremos de la

liga, además de la orientación de la misma, datos que ordinariamente se encuentran en el esquema de la base de datos.

Recordando la estructura propuesta por nosotros en el capítulo 2, la etapa anterior al analizador semántico es la del análisis sintáctico. Esta etapa no fue desarrollada para este trabajo; sin embargo, dada la gran cantidad de información sobre el tema, y tomando en cuenta que existen analizadores sintácticos desarrollados para ser acoplados a analizadores semánticos como el realizado en este trabajo, se considera que esta limitación es temporal, y podrá ser eliminada en el futuro. Es por lo anterior que en el prototipo implantado el usuario tendrá que proporcionar directamente la información requerida por el analizador semántico.

Dicha información de entrada (teniendo en cuenta que ya se cuenta con las matrices de caminos y semántica) proporcionada por el usuario será:

1. La entidad solución: Se requiere el nombre de la clase de entidad solución definida en la pregunta.

2. Las entidades fijas: Se requieren los nombres de las clases de entidades fijadas en la pregunta.

3. **Las fijaciones de las entidades: Se requieren los datos proporcionados en la pregunta sobre los atributos y valores de fijación de las clases de entidades fijas.**

4. La semántica que une las diferentes entidades fijas con la entidad solución: Se requiere el enlace semántico entre las entidades fijas y la entidad solución, en término de verbos estandarizados, pudiéndose dar casos de semántica compuesta.

El analizador semántico realiza las siguientes operaciones:

1. Construye una lista con las operaciones en álgebra relacional a que equivalen las fijaciones indicadas.

2. Busca los caminos referenciados por la semántica indicada, y construye una lista con ellos.

3. Ordena los caminos de forma tal que se puedan resolver las intersecciones satisfactoriamente.

4. Ordena las fijaciones de las clases de entidades de acuerdo con el resultado del punto anterior.

5. Resuelve las intersecciones que se presenten, aplicando el criterio que más se apeque al caso de intersección presentado.

6. Muestra al usuario el resultado del punto anterior, así como el orden final de las entidades fijas (resultado del punto 4).

7. Elabora la expresión final en álgebra relacional, conjuntando las fijaciones con los caminos, construyendo paulatinamente una lista con dicha expresión.

8. Muestra al usuario el resultado.

En el apéndice B se muestran algunos ejemplos de la operación del programa.

A continuación se muestra el listado de la rutina de análisis semántico, con comentarios que muestran los lugares donde se realizan las acciones desglozadas arriba.

```
(DEFUN analiza ()
  (FORMAT T "~%*** ANALISIS SEMANTICO ***")
  (LET
  ((ent-fin (lee-ent-fin))      <----- se pide entidad solu-
   (atr-fin (lee-atr-fin))             cion, atributo solu--
   (ent-ini (lee-ent-ini)))            y entidades fijas.
    (COND
      ((no-existe-alguna ent-ini)  <-- se verifica la exis-
                                       tencia de las enti--
                                       dades fijas
       (THROW 'CONTINUA))
      (T
       (analiza-1
         (selecciones ent-ini)     <-- se piden las fijacio-
                                       nes de las entidades.
         (viajes ent-ini
                 (semantica        <-- se pide la semanti-
                                       ca que une las en-
                                       tidades fijas con
                                       la solucion.
                            ent-ini
                            ent-fin))
                 ent-ini
                 atr-fin)))))


(DEFUN analiza-1 (selecciones viajes ent-ini atr-fin)
  (analiza-2 selecciones
             viajes
             ent-ini
             atr-fin
             (ordena                  <- se realiza el ordena-
                                         miento de los viajes
                                         para optimar las inter-
                                         secciones.
                     (separa viajes))))
```

```
(DEFUN analiza-2 (selecciones viajes ent-ini atr-fin
                  viajes-ordenados)
  (LET
    ((ent-ord                    <----- se ordenan las enti-
                                        dades fijas de acuer-
                                        do con el nuevo orden
                                        de los viajes.
       (ordena-entidades viajes
                         viajes-ordenados
                         ent-ini))

     (viaje-fin                  <----- se calcula la inter-
                                        seccion de los via--
                                        jes

               (interseccion viajes-ordenados)))

     (FORMAT                     <----- se despliegan los re-
                                        sultados obtenidos.

             T "~%~%ENTIDADES FINALES: ~A" ent-ord)
     (FORMAT T   "~%VIAJE FINAL      : ~A~%" viaje-fin)
     (FORMAT T "~%Codificacion:~%")

     (PPRINT                     <----- se codifican los re-
                                        sultados y se des-
                                        pliega la expresion
                                        en algebra relacional.
           (codifica ent-ord
                     viaje-fin
                     atr-fin
                     selecciones))))
```

6 CONCLUSIONES.

6.1 RESUMEN.

En este trabajo se han planteado las limitaciones de las interfaces actuales para consulta de bases de datos en lenguaje natural, destacando su dependencia del dominio de aplicación y de la estructura de la base de datos, que las hace difíciles de acoplar a nuevos dominios. Se han expuesto la arquitectura típica de los sistemas de esta clase, y se han propuesto una arquitectura diferente para solucionar este problema.

El planteamiento realizado se basa en la posibilidad de obtener información de un diccionario de datos, con el fin de que éste pueda ser empleado como auxiliar en el análisis de las preguntas de entrada. Se propone el empleo del modelo conceptual ELKA para representar el esquema de la base de datos, y se establece y demuestra la posibilidad de viajar por este modelo para encontrar una expresión que define la relación que responde a la pregunta original en lenguaje natural. Se justificó la conveniencia de utilizar una estructura de tipo relacional para el modelo de la base de datos, y se diseñó un analizador semántico que realiza el viaje por el esquema, y cuya salida es una expresión en álgebra relacional que define el conjunto de datos que integran la información que responde a una pregunta en lenguaje natural, previamente transformada en una estructura

intermedia en términos de objetos de la base de datos.

6.2 RESULTADOS.

El esquema de la base de datos expresado en forma de un modelo ELKA contiene suficiente información semántica para realizar el análisis semántico de una pregunta en lenguaje natural sobre la información de la base de datos.

Para poder formar una expresión equivalente a la pregunta original es necesario realizar un viaje por el modelo de datos, lo que equivale a generar operaciones en álgebra relacional.

En toda pregunta en lenguaje natural se proporciona información que involucra objetos de la base de datos, representados mediante clases de entidades contenidas en ella. Cuando se fija el valor de un atributo en una clase de entidades, se difine de manera indirecta un conjunto de información relacionada con las entidades definidas por esta fijación. Este es el concepto de macroinstancia de una base de datos para los valores definidos.

Existe una clase de entidades que contiene la información requerida para contestar la pregunta de entrada. El viaje de las clases de entidades fijas (definidas por valores asignados a atributos) a la clase de entidades en la que se encuentra la información que soluciona la pregunta

original equivale a una serie de operaciones que, al ser realizadas sobre la base de datos, definen el conjunto de datos que responde a la pregunta en lenguaje natural.

Cuando hay más de una clase de entidades fija, habrá que realizar más de un viaje a la clase de entidades solución; el número de viajes a realizar será igual al número de clases de entidades fijadas en la pregunta.

Estos viajes se intersectan en alguna clase de entidades del modelo. Esta intersección entre los viajes realizados define una intersección entre las relaciones obtenidas hasta la clase de entidades donde se encuentran los caminos, y se expresa como la operación llamada intersección en algebra relacional.

En algunos casos, la intersección de dos caminos puede ser expresada en forma de una selección sobre uno de ellos. Esto sucede cuando uno de los dos viajes está contenido en el otro.

Se ha planteado una estructura llamada matriz de caminos por el esquema, que contiene todos los caminos que unen entre sí las clases de entidades de la base de datos. La pregunta en lenguaje natural tiene un contenido semántico que debe ser compatible con el contenido semántico de los caminos que se recorran para solucionarla.

En los casos de que la pregunta original contiene una semántica que no está implicada en el esquema de la base de datos a la que se aplica, se considera que esta pregunta es semánticamente incorrecta para dicha base de datos, lo que significa que la pregunta se ha planteado fuera del contexto definido por la base de datos consultada.

Cuando dos clases de entidades se encuentran relacionadas mediante más de un camino, cada uno de estos caminos puede tener implicaciones semánticas diferentes. En tal caso, la relación semántica entre ellas -expresada en la pregunta- dará la clave para elegir el camino que se debe recorrer para solucionar este conflicto. La semántica de los caminos que unen las clases de entidades fijas con la clase de entidades solución en la pregunta está implícita en el verbo de dicha pregunta.

La semántica de un modelo ELKA se expresa mediante los verbos asignados a cada liga del modelo; se define una estructura llamada matriz semántica de la base de datos, para contener esta información.

Un diccionario de datos puede ser empleado como fuente de informacion que auxilia el análisis de una pregunta en lenguaje natural, de tal manera que se logre independencia de la interfaz tanto del dominio de aplicación, como de la estructura de la base de datos.

Lo anterior sustenta la arquitectura propuesta en este trabajo como solución al problema de la independencia de una interfaz en lenguaje natural para la consulta de bases de datos.

6.3 TRABAJOS FUTUROS.

A continuación se enlistan los trabajos que será necesario realizar en el futuro para, a partir de los resultados de la presente tesis, continuar la investigación y el desarrollo sobre el tema hasta lograr un sistema operacional a escala real.

El prototipo del analizador semántico realizado constituye la base para un analizador semántico operacional; en él se incluyen las operaciones básicas del álgebra relacional útiles para definir la macroinstancia, que resulta de realizar las fijaciones en el esquema de la base de datos. La mayoría de las preguntas de la encuesta fueron solucionadas eficazmente con este prototipo. Algunos de los aumentos que deberán de hacerse a este prototipo con el fin de que sus respuestas sean más completas -ya no más eficaces- se muestran a continuación:

a) Considerar la posibilidad de que se presente más de una clase de entidades de las cuales extraer la información que responda a la pregunta.

b) **Incluir operaciones no algebraicas a la salida del analizador, tales como cuantificadores para obtener cardinalidades de relaciones resultantes, promedios,** operaciones aritméticas simples, que le proporcionarán la posibilidad de contestar mayor cantidad de preguntas muy comunes.

El trabajo necesario para incorporar este prototipo en un sistema completo de interpretación de preguntas en lenguaje natural a bases de datos es el siguiente:

a) Acoplar un analizador sintáctico a la entrada del prototipo realizado, empleando las técnicas en existentes, como las documentadas en [HAD71, MAC86, MAR85a, WIN83].

b) Encontrar la manera de que la información obtenida a la salida sea "amigable", mediante la implantación de criterios que permitan seleccionar información a incluir a la salida (se recomienda consultar "Natural Language Database Interfaces: The User View" [FOG81] para este fin).

Adicionalmente, para llegar al desarrollo completo de un sistema de base de datos en lenguaje natural será necesario realizar proyectos encaminados a resolver el problema de creación y modificación de una base de datos

mediante oraciones en lenguaje natural.

Como se ha visto, el camino que habrá que recorrer para lograr el desarrollo completo de un sistema de definición y acceso de base de datos en lenguaje natural es largo. Sin embargo, se considera factible, y la investigación y el desarrollo realizados en esta tesis conforman cimientos importantes en la construcción de un sistema de tal naturaleza.

[AHO77] AHO, Alfred V., Ullman, Jeffrey D. Principles of Compiler Design. Addison-Wesley Publishing Co. Reading, Mass., 1977.

[ANS75] ANSI/X3/SPRAC Study Group on Database Management Systems, Interim Report. FTD (Bulletin of ACM-Sigmond) Vol. 7, No. 2 1975.

[BAR81] BARR, Avron, et. al.The Handbook of Artificial Intelligence Vol. I. Heuristech Press Stanford, Cal. 1981.

[BAT84a] BATORY, D.S., Buchmann A.P. Molecular objects, abstract data types and data models: a framework. VLDB X. Signapore, Aug. 1984.

[BL033] BLOOMFIELD, Leonard. Language. Holt, Rinehart and Winston. New York, 1933.

[BOB77] BOBROW, Daniel G., et.al. Gus, A Frame-Driven Dialog System. Artificial Intelligence 8(2). April, 1977.

[CHA79] CHAMPINE, G.A. Current Trends in Data Base Systems. Computer. IEEE. May, 1979.

[CH069] CHOMSKY, Noam. Syntactic Structures. Mouton. La Haya, 1969.

[COD72] CODD, E.F. Relational Completeness of Data Base Sublanguages. Data Base Systems. Courant Computer Science Simposia Series, Vol. 6. Prentice-Hall. Englewood Cliffs, N.J., 1972.

[COD74] COOD, E.F. Seven Steps to RENDEZVOUS with the casual user. Proc. IFIP TC-2 Working Conf. on Database Management Systems. North-Holland Publishing Co., Amsterdam, 1974.

[CUR81] CURTICE, Robert M. Data Dictionaries: An Assessment of Current Practice and Problems. Very Large Databases IEEE, 1981.

[DAT82] DATE, C.J. An Introduction to Database Systems. The Systems Programming Series. Addision-Wesley Publishing Co. 1982.

[FAR86] **FARGUES, Jean, Marie-Claude, DUGOURD, Anne, CATACH, Laurent. Conceptual graphs for semantics and knowlegde processing. IBM J. RES. DEVELOP. Vol. 30 No. 1. January, 1986.**

[FOG81] FOGEL, Marc H. Natural Language Database Interfaces: The User View. Tutorial: Data Base Management in the 1980's. IEEE, 1981.

[GON85] GONZALEZ-SUSTAETA J.C. Sistema Integrado de Diccionario de Datos Basado en el Modelo Conceptual ELKA. Tesis de Maestría. ITESM Unidad Morelos, Febrero de 1985.

[GON86] GONZALES-SUSTAETA, J.C.; Buchmamn, Alejandro P. An Automated Database Design Tool Using the ELKA Conceptual Model. 23rd. Design Automation Conference. ACM/IEEE DAC86. Las Vegas, Nevada, Junio de 1986.

[GRA84] GRAY, Peter M. Logic, Algebra and Databases. John WILLEY SONS. Great Britain, 1984.

[HAD71] HADLICH, Roger L. Gramática transformativa del español. Gredos. Madrid, 1971.

[HEN81] HENDRIX, Gary G., et.al. Developing a Natural Language Interface to Complex data. Tutorial: Data Base Management in the 1980's. IEEE, 1981.

[HEN81a] HENDRIX, Gary G. Natural Language Processing The Field in Perspective. BYTE. September, 1981.

[KUH70] KUHN, Thomas. The Structure of Scientific Revolutions. University of Chicago Press. Chicago, 1970.

[LES81] LESMO, Leonardo, et.al. Lexical and Pragmatic Knowledge for Natural language Analysis. Cybernetics and Society. IEEE, 1981.

[LEW78] LEWIS, Phillip M., ROSENKRANTZ, David J., STEARNS, Richard E. Compiler Design Theory. 3a.ed. Addison-Wesley Publishing Co. Reading, Mass. 1978.

[MAC86] MACIAS, Benjamin P. Análisis sintáctico del español por computadora. Tesis de Licenciatura en Actuaria. Facultad de Ciencias, UNAM. México, 1985.

[MAR85] **MARTIN, Paul, et.al. TEAM: An Experimental Transportable Natural-Language Interface. Database Engineering 8(3). IEEE, Sept. 1985.**

[MAR85a] MARCUS, Mitchell P. A Theroy of Syntactic Recognition for Natural Language, 3a.ed. The MIT Press. Cambridge, Massachusetts, 1985.

[MIN75] MINSKY, Marvin. A framework for representing knowledge. WINSTON, P. (Ed). The Psicology of Computer Vision. Mc. Graw-Hill. New York, 1975.

[PYL85] PYLYSHYN, Z.W., KITTREDGE, R.I. Databases and Natural Language Processing. IEEE Database Engineering 8(3). IEEE, Sep. 1985.

[RIC84] RICH, Elaine. Artificial Intelligence. Mc. Graw-Hill, 1984.

[ROH81] ROHDE, Wm. F. Language Tools for data access...Past; Present and Future. Database Management in the 1980's. IEEE, 1981.

[ROD81] RODRIGUEZ Ortiz, Guillermo. The ELKA Model Approach to the Design of Database Conceptual Models. PhD. Thesis. UCLA, 1981.

[SAN82] SANFORD, David I/ROACH, J.W. Evaluating Natural Language Communication to Improve Human-Computer Interaction. Cybernetics and Society. IIE, 1982.

[SCH75] SCHANCK, Roger C. Conceptual information processing North-Holland Publishing Co. New York, 1975.

[SOW86] SOWA, John F., WAY, Eileen C. Implementing a semantic interpreter using conceptual graphs. IBM J. RES. DEVELOP. Vol. 30 No. 1. January, 1986.

[ULL80] ULLMAN, Jeffrey D. Principles of Database Systems. Computer Science Press, 1980.

[VAN80] VAN DIJK, Teun A. Texto y Contexto (Semántica y Pragmática del discurso). Ediciones Cátedra, S.A. Madrid, 1980.

[WAL78] WALTZ David L. An English Language Question Answering System for a Large Relational Database. Comunications of the ACM. July, 1978. Vol. 21, Num 7.

[WEA55] WEAVER, Warren. Translation en. LOCKE, W.N. y BOOTH, A.D. (Eds). Machine Translation of Languages Technology Press of MIT and Wiley. New York, 1955.

[WIE84] WIEDERHOLD, Gio. Databases. Computer. IEEE, October, 1984.

[WIN72] WINOGRAD, Terry. Understanding Natural Language. Academic Press Inc. New York, 1972.

[WIN83] WINOGRAD, Terry. Language as a cognitive process. Volume I: Syntax. Addison-Wesley Publishing Co. Reading Mass, 1983.

[WIN84] WINSTON, Patrick Henry. Artificial Intelligence. 2a. ed. Addison-Wesley Publishing Co. Reading, Mass. Feb. 1984.

[WIN85] WYNNE, N.T. Advances in Energy Management Systems. Scada and Hardware. IEEE 14th Power Industry Computer Applications (PICA85). San Francisco, CA. May, 1985.

[WOO70] WOODS, W.A. Transition network grammars for natural language analysis. Communications of the ACM. Vol. 13, No. 10, October, 1970.

BIBLIOGRAFIA ANOTADA.

(N) ::= Nota
(C) ::= Comentario

[BAL84] BALLARD, Bruce W., Thinkham, Nancy L. A Phrase-Structured Grammatical Framework for Transportable Natural Language Processing. Computational Linguistics 10(2). April-June 1984.

(N) Presentan un formalismo gramatical que se basa en un aumento de las reglas de estructura de frases que permite al analizador sintáctico resolver algunas ambigüedades especificas del dominio, mediante la consulta a una gramática predefinida y un conjunto de archivos producidos durante una sesión inicial con el usuario.

(C) Entra demasiado en detalle. Puede resultar útil como referencia en la construcción de un analizador sintáctico. Su formalismo gramatical contiene elementos de muchas corrientes.

[BAT84] BATES, Madeleine. Accessing a Database with a Transportable Natural Language Interface. Computer Society Conference on Artificial Intelligence Applications. IEEE, 1984.

(N) Describe las diferentes caracteristicas del sistema IRUS (Information Retrieval Using the RUS parser), que contiene muchas semejanzas con el enfoque propuesto en esta tesis.

(C) Como la mayoria de los articulos, resulta demasiado ambicioso y, por lo tanto, vago.

[BER84] BERISTAIN, Helena. Gramática Estructural de la lengua Española 3a.ed. Universidad Nacional Autónoma de México. México, 1984.

(N) Este es un texto programado que sirve para aprender gramática del español con el enfoque estructuralista.

(C) Muy bien realizado. Si se resuelven los ejercicios propuestos, se adquiere una clara idea sobre la estructura del español.

[COL84] **COLOMBETTY, Marco, et.al. Reasoning in Natural Language for Designing a Data Base. Artificial and Human Intelligence. Elsevier Science Publishers, B.V. NATO, 1984.**

(N) El propósito de su trabajo es lograr que el usuario defina una base de datos relacional por medio de oraciones en lenguaje natural, y se denomina NLDA (Natural Language Design Aid).

[EVR81] EVRARD, Fabrice, Henry FARRENY, Henri PRADE Non-Grammatical-Guided System for sentence analysis in a limited context. CYBERNETICS AND SOCIETY. IEEE, 1981.

(N) El proyecto que describen tiene como fin desarrollar una interfaz en lenguaje natural para su robot HILARE. La gramática se emplea sólo para averiguar la estructura de los diferentes tipos de frases que se presentan en la oración ("Noun-phrase", "verb-phrase", etc.). Los autores se basan en la afirmaación de que la clave del significado está en el verbo, y aquellas frases que no se logran "entender" se interpretan como ruido. Emplea gramática de casos aplicada mediante marcos.

[HEN82] HENDRIX, GARY G. Natural Language Interface. American Journal of Computational Linguistics. Volume 8, Number 2. April-June, 1982.

(N) En este artículo se habla con claridad sobre las ventajas de las interfaces de lenguaje natural, así como sus desventajas. Distingue tres líneas de investigación y tres niveles para clasificar los sistemas. Define la transportabilidad de una interfaz como su capacidad para ser fácilmente trasladada de un dominio a otro.

[JAR85] JARKE, Matthias, et.al. A Field Evaluation of Natural for Data Retrieval. IEEE Transactions on Software Engineering. Vol. SE-11 No. 1, January, 1985.

(N) Este artículo, junto con el que sigue del mismo autor, constituyen los estudios más serios sobre la factibilidad, ventajas y desventajas del uso de interfaces de lenguaje natural para la consulta de bases de datos.

[JAR85a] JARKE, Matthias, et.al. Evaluation and assessment of a Domain-Independent Natural Language Query System. Database Engineering 8(5). September,

1985.

(N) En este trabajo se evaluó la interfaz USL, desarrollada en IBM. Los resultados obtenidos son muy interesantes sobre la reacción de los usuarios ante sistemas de esta clase. Afirma que uno de sus problemas principales es que para cambiar de dominio de aplicación es necesario tener conocimientos muy especializados.

[KAP72] KAPLAN, Ronald M. Augmented Transition Networks as Psycological Models of Sentence Comprehension. Artificial Intelligence, Vol. 3, 1972.

(N) Revisa la operación de las ATN's. En este sentido es preferible consultar el artículo de Woods ([WOO70]). Tiene ejemplos que resultan muy interesantes: voz pasiva y oraciones subordinadas. Justifica el empleo de este formalismo gramatical, y finalmente ejemplifica la construcción de una ATN.

[KAR84] KARNA-N., Kamal. Artificial Intelligence for man-machine Interface Guest Editor's Introduction. COMPUTER. IEEE, September 1984.

[KIM85] KIMBRELL, Roy E. English Recognition. BYTE. December, 1985.

(N) En este artículo se trata la construcción de reconocedores sintácticos del inglés, concretamente un "juguete" llamado "Higgins". Habla de los ATN's desde un punto de vista práctico. Contiene el pseudocódigo de los cuatro tipos de transición empleados en las ATN's. Contiene, además, una forma propuesta para clasificar el lexicón.

(C) Es una muy buena introducción y explicación de ATN's. Yo lo recomendaría, junto con [WOO70], como dos referencias imprescindibles para la comprensión del formalismo.

[LAR81] LARSON, James A., Freeman, Harvey. Section I. Introduction. Tutorial: Database Management in the 1980's 1981. IEEE, 1981.

[LEH85] LEHMANN, Hubert, et.al. A Multilingual Interface to Databases. Database Engineering 8(5). September, 1985.

(N) Habla sobre el USL (User Speciality Language), y dan por hecho su "portabilidad". Los idiomas que

puede interpretar son el inglés, el francés, el alemán (original), español e italiano. Desgloza las dos partes del lenguaje para su interpretación: dependiente de la aplicación e independiente de la aplicación. Mencionan la posibilidad de desarrollar teoría sobre "software ergonomics".

(C) Describe la estructura del sistema de una manera que resulta complicada.

[LEI86] LEIGH, William and EVANS, James. Interpretation of Natural Lenguage Database Queries Using Optimization Methods. IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-16 No. 1. IEEE, January-February 1986.

(N) Los autores emplean métodos de optimización con el fin de seleccionar el conjunto de roles de las palabras (significados, estructuras y referencias) que arrojan la mejor solución.

(C) Estas técnicas resultan útiles cuando se desea averiguar la parte de la base de datos aludida por alguna parte de la oración. Sin embargo, este artículo resulta difícil de comprender.

[MAZ81] MAZLACK, Lawrence J. Leigh, William E. Applying pattern templates to the problem of interpreting queries. Cybernetics and Society. IEEE, 1981.

(N) Los autores proponen un enfoque en el que el modelo de datos es concebido como un elemento útil para guiar ela interpretación de consultas en lenguaje natural. Define conceptos como "vista".

[MOS84] MOSER, M.G. Domain Dependent Semantic Aquisition. Computer Society Conference on Artificial Intelligence Applications. IEEE, 1984

(N) En este artículo se concentran en la descripción de un sistema llamado IRACQ (Interpretation Rule Acquisition), y también aluden la gramática del IRUS ([BAT84]). Define clase semántica como "some set of things in the domain". Su análisis se basa en la gramática de casos. El IRACQ produce entonces una representación gramatical mediante interacciones con el usuario.

[PLE72] PLEYAN, Carmen, LOPEZ, José García. Sintagma, 4a ed. TEIDE, Barcelona, 1972.

(N) El sintagma es una parte de la relación

paradigma-sintagma de la gramática estructural. Se refiere a un grupo de palabras organizadas en formas con significado, es decir, una oración.

(C) Este libro constituye un estudio escueto de la estructura de la oración. Resulta muy breve, por lo que conviene emplearlo sólo como referencia.

[PYL85a] PYLYSHYN, Zenon W. Alternatives for the Use of Natural Language in Interfacing to Database. Database Engineering 8(5). September, 1985.

(N) En este artículo se plantean dos alternativas al empleo de lenguaje natural en interfaces. La primera la constituyen los sistemas expertos que asesoren al usuario en su interacción con la base de datos, y la segunda son los sistemas "no inteligentes" guiados por menús.

[RIC84a] RICH, Elaine. Natural-Language Interfaces. COMPUTER, IEEE, September, 1984.

(N) En este artículo se plantea la posibilidad de que las interfaces en lenguaje natural no sean las ideales a emplear en todo momento; se exponen algunos factores que ayudan en la decisión de emplear una interfaz de esta clase. Además se desglozan los componentes de un sistema de comprensión de lenguaje natural, para finalmente proponer tres enfoques alternativos para la realización de una interfaz.

[SIM73] SIMMONS, R.F. Semantic Networks: Computation and Use for Understanding English Sentences in Shanck, et.al. Computer Models of Thought and Language, W.H. Freeman and Company. San Francisco, 1973.

(N) En este artículo se proponen las redes semánticas como un modelo de ideas ligadas entre sí -mediante ligas semánticas-. Las propone como medio para modelar formalmente la comunicación.

[THO82] THOMASON, Michael G. Syntactic/Semantic Pattern Recognition. CYBERNETICS AND SOCIETY. IEEE, 1982.

(N) En este artículo se formaliza el proceso de reconocimiento de patrones semánticos y sintácticos. Se da la definición formal de una gramática y de semántica.

[THO85] THOMPSON, Craig W. Menu-Based Natural Language Interfaces to Databases. Database Engineering 8(5)

September, 1985.

(N) Discute el sistema NLMenu, describiendo su enfoque principal, sus ventajas y aplicaciones.

[WEB83] WEBBER, Bonnie Lynn. Logic and Natural Language Computer. IEEE, October, 1983.

(N) Propone el empleo de tres clases de lógica en el análisis del lenguaje natural:

- Lógica de "default" para resolver presuposiciones.

- Lógica modal para planear expresiones.

- Lógica temporal para razonar sobre el futuro.

[WIN84a] WINOGRAD, Terry. Computer Software for Working with Language. Scientific American, Vol. 251 No. 3, 1984.

(N) Este articulo describe en detalle la estructura de los sistemas de análisis del lenguaje. Contiene además una buena explicación de los diversos tipos de ambigüedades.

APENDICE A

RESULTADOS DE LA ENCUESTA.

A.1 INTRODUCCION.

Al momento de iniciar el proyecto de esta tesis, no se contaba con información sobre las estructuras del lenguaje, ni se conocía con certeza la manera en que una pregunta en lenguaje natural se relaciona con la base de datos de la que se pretende obtener una respuesta que la satisfaga.

Esa fue la razón por la que se decidió consultar a un lingüista, concretamente la lingüista Margarita Palacios, de la Dirección de Investigaciones Lingüísticas de la UNAM. Ella sugirió realizar una encuesta en que personas de diferentes áreas de conocimiento formularan una serie de preguntas a un archivo hipotético. Estas preguntas servirían para definir reglas de significación y búsqueda de núcleos de significado.

Las características de la encuesta realizada son las siguientes:

- La encuesta fué aplicada a personas que de preferencia no tenían experiencia en el manejo de computadoras o lenguajes formales, debido a que se consideró que podía haber cierta influencia de dicho conocimiento en las preguntas formuladas, de tal manera que no fueran tan naturales como se deseaba.

- Se procuró que las preguntas formuladas fueran aisladas entre sí, con el fin de evitar por el momento referencias contextuales a preguntas formuladas con anterioridad.

- Se definieron archivos de información hipotéticos para cada especialidad de las siguientes:

  - Medicina.

  - Arquitectura.

  - Contaduría.

  - Docencia (control escolar).

- Se empleó lenguaje coloquial en su explicación, con el fin de evitar caer en tecnicismos que confundieran o impactaran a la persona que la leyera.

- **Se solicitaron paráfrasis de las preguntas, con el fin de poder encontrar los elementos constantes en preguntas cuyo trasfondo es básicamente el mismo, aún cuando su formulación es distinta.**

A continuación se muestra un ejemplo del formato empleado para las encuestas:

Esta encuesta tiene como objetivo recopilar una serie de preguntas hechas por diferentes personas a un archivo hipotético.

Las preguntas así obtenidas, serán analizadas en su estructura como parte de un proyecto que estudia la factibilidad de sistematizar la comprensión del lenguaje natural.

Le agradecemos de antemano su amable colaboración.

Usted es el Coordinador Académico de la Escuela Técnica No. 20, y tiene un archivo que contiene información sobre:

i) Alumnos, con su número de cuenta, nombre, dirección, teléfono, fecha de nacimiento, fecha de admisión, promedio, lugar de trabajo, etc.

ii) El historial de cada alumno, con las claves de las materias cursadas, el profesor con quien las cursó, las calificaciones obtenidas, etc.

iii) Las materias, su clave, nombre, grupo, semestre, créditos, etc.

iv) Los profesores, su nombre, dirección, teléfono, las materias que imparten, etc.

Una persona -el archivista- se encarga de manejar el archivo para contestar las preguntas que se le formulan sobre la información que contiene.

Deseamos saber ejemplos de preguntas que usted le formularía al archivista. De ser posible, plantee varias formas alternas en que usted haría una misma pregunta, por ejemplo:

a) Cuántos alumnos han reprobado Historia de México ?

b) Dime el número de alumnos que no han aprobado Historia de México.

c) Deseo saber qué cantidad de alumnos tienen calificación reprobatoria en Historia de México.

A.2 TABLA DE RESULTADOS.

Una vez realizada la encuesta, se procedió a la clasificación del material recopilado de la siguiente manera:

- Se construyó un modelo de datos con la información requisitada por las preguntas obtenidas.

- Se tradujeron las preguntas a expresiones en álgebra relacional que definieran el conjunto de datos que respondiera a la pregunta en turno.

- Se procedió a la clasificación de dichas formulaciones en álgebra relacional, tomando en cuenta los siguientes parámetros:

  - El tipo de respuesta pedida, que podía ser el valor de un atributo determinado, o la cardinalidad de una relación resultante.

  - El número de clases de entidades fijadas en la pregunta.

  - El número de selecciones necesarias para fijar dichas clases de entidades, así como las características de dichas selecciones.

- **El número de proyecciones necesarias para realizar los viajes.**

- El número de JOINS empleados para viajar de las clases entidades fijas a la clase de entidades solución.

- El tipo de caminos encontrados, en función del tipo de intersección obtenida en el trayecto.

La tabla obtenida se muestra a continuación:

TIPO	# CLASES DE ENTIDADES FIJAS	# SELEC-CIONES	# PROYEC-CIONES	#JOINS	TIPO DE CAMINO
COUNT	1	1	-	-	0
COUNT	1	1	-	-	0
COUNT	3	1	2	2	1?
COUNT	1	1	-	-	0
COUNT	2	1*	1	1	1
ID	3	1	3	2	1
ID	3	1	3	2	1
COUNT	3	1	2	2	1?
COUNT	2	2	1	1	3a
COUNT	1	1	-	-	0
COUNT	3	2	2	2	1?
COUNT	1	1	-	-	0
COUNT	1	1	-	-	0
ID	1	-	1	-	0
COUNT	3	1	2	2	1?
COUNT	1	-	-	-	0
COUNT	3	1	2	2	1?
COUNT	1	1*	-	-	0
ID	3	1	3	2	1
ID	3	1	3	2	1
COUNT	1	-	-	-	0
COUNT	1	-	-	-	0
ID	1	1	1	-	0
COUNT	3	1	2	2	1?
ID	2	1	2	1	1
ID	3	1	3	2	1
ID	4	1	4	3	1
ID	1	1	1	-	0
ID	3	1	3	2	1
ID	3	1	3	2	1
COUNT	5	2	4	4	2b?
ID	1	1	1	-	0
COUNT	4	2	3	3	3b?
COUNT	3	2	2	2	3a
COUNT	1	1	-	-	0
ID	2	1	2	1	1
COUNT	1	1*	-	-	0
COUNT	4	1	3	3	1
COUNT	2	1*	1	1	1
COUNT	1	1	-	-	0
ID	3	1	3	2	1
ID	2	1	2	1	1
ID	2	1	2	1	1
COUNT	4	2	3	3	3b?
COUNT	2	1*	1	1	1?

COUNT	2	1	1	1	1?
COUNT	3	1	2	2	1?
COUNT	2	1	1	1	1?
ID	3	1	3	2	1
ID	1	1	1	-	0
ID	5	4!	5	4	2b, 3b
ID	3	2	3	2	3b
ID	1	1	1	-	0
ID	3	2★	3	2	2a

A.3 SIMBOLOGIA EMPLEADA.

A.3.1 TIPO DE PREGUNTAS.

COUNT = Significa que hay que contar el número de elementos de la relación resultante (cardinalidad).

ID = Significa que sólo es necesario mostrar la relación resultante.

A.3.2 CLASIFICACION DE LAS SELECCIONES.

! (SELEC) = Selecciones repetidas (agilización del query).

★ (SELEC) = Selecciones con expresión compuesta.

A.3.3 TIPOS DE CAMINO.

A.3.3.1 CLASIFICACION DE LOS VIAJES. -

0 - No se realizó viaje
1 - Viaje sencillo, con una entidad fija hacia la solución.
2 - Intersección de dos caminos.
3 - Es el caso en que un camino está comprendido en otro (s).

Aqui la intersección se convierte en una selección sobre el supercamino.
2a - La intersección es en la entidad solución.
2b - La intersección es antes de la entidad solución.
3a - La selección está en la entidad solución.
3b - La selección está antes de la entidad solución.

A.3.3.2 VIAJES SOBRANTES. -

? (VIAJE) = Hay un viaje de más (ya no se ocupa) (en casi todos los count)

APENDICE B

EJEMPLOS DE OPERACION DEL PROTOTIPO.

A continuación se muestran algunos ejemplos de la operación del prototipo realizado. Los primeros cinco ejemplos fueron extraídos de los ejemplos de la sección 5.3 de la presente tesis. Los siguientes son ejemplos de los casos en que se proporcionan semánticas compuestas:

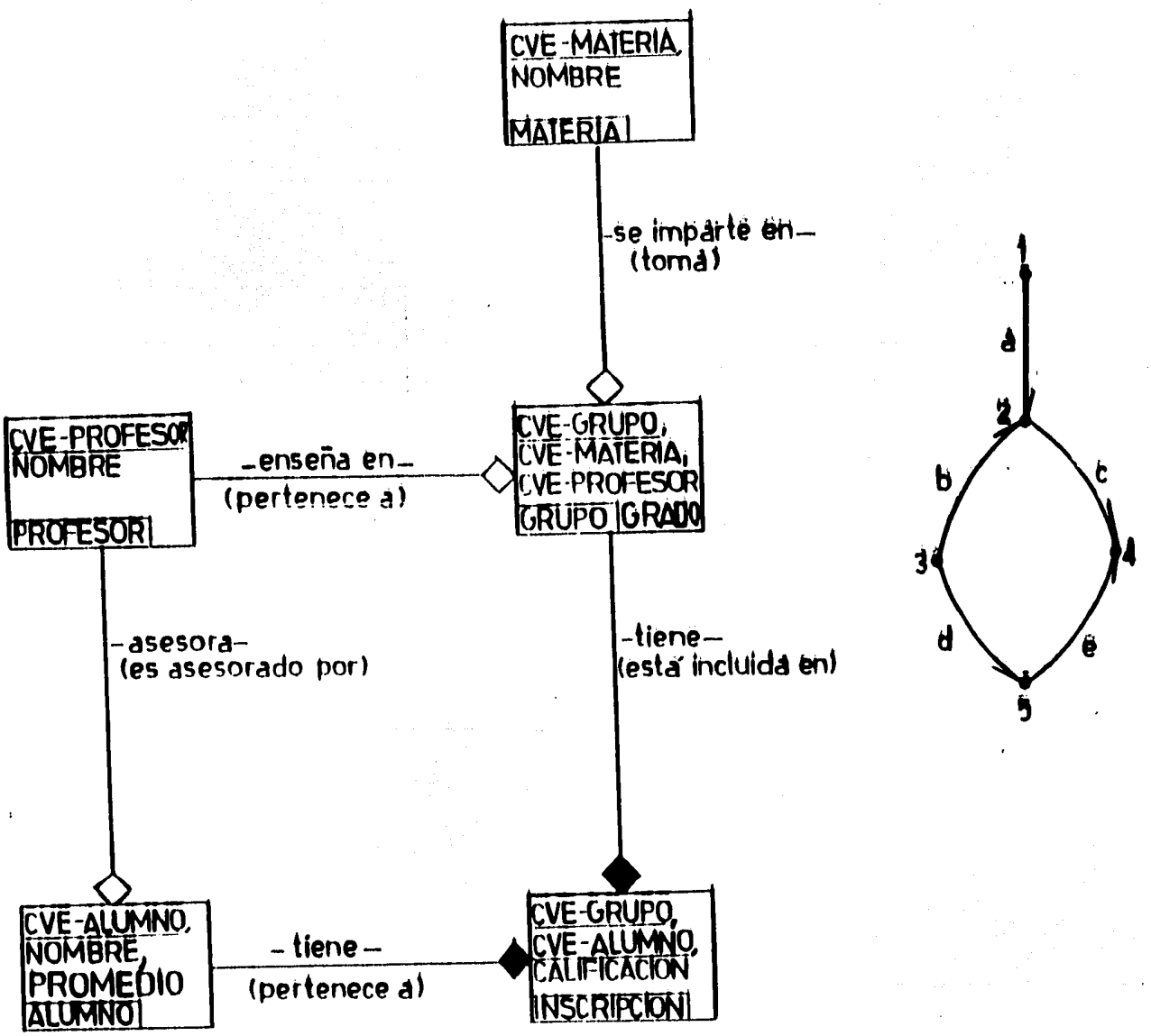
- El primer ejemplo de esta sección complementaria emplea el enlace semántico puro existente entre alumno y materia ("ALUMNO.toma.MATERIA").

- El segundo utiliza la semántica compuesta equivalente a la empleada en el ejemplo anterior. Se demuestra entonces la equivalencia (para el prototipo) entre este camino y el definido en el ejemplo anterior.

- Debido a que puede manejar enlaces semánticos compuestos, el prototipo realizado tiene la capacidad de manejar viajes cíclicos por el esquema. Esto es lo que se demuestra en el tercer ejemplo de la sección

complementaria de este apéndice.

En este apéndice se emplea el modelo de datos de la sección 5.3, y lo reproducioms aquí para facilidad del lector.



a) El modelo ELKA. b) Su gráfica inversa.

FIGURA B.1.- Modelo ELKA de la sección 5.3.

EJEMPLO 5.3.1.

¿Qué calificaciones tiene Juan Cardini?

A continuación se muestra la manera como se realiza el análisis semántico de esta pregunta con el prototipo. La interacción con el prototipo tiene dos etapas: en la primera (etapa de interacción), el prototipo solicita los datos al usuario (información de entrada) que en un sistema completo sería proporcionada por el analizador sintáctico; en la segunda etapa (de salida), el prototipo muestra al usuario el orden final de las clases de entidades fijas, así como el viaje a realizar a partir de ellas; finalmente, y dentro de esta misma etapa, el prototipo muestra el resultado obtenido al codificar el viaje en álgebra relacional.

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
   Entidad solucion>  (INSCRIPCION)
   Atributo solucion> (CALIFICACION)
   Entidades fijas>   (ALUMNO)

*** DATOS SOBRE ALUMNO ***
   Atributo>   NOMBRE
   Comparador> =
```

Valor> "JUAN CARDINI"

Semantica que relaciona ALUMNO con (INSCRIPCION)?
(ALUMNO TIENE INSCRIPCION)

ETAPA DE SALIDA.

```
ENTIDADES FINALES: (ALUMNO)
VIAJE FINAL      : ((E))
```

Codificacion:

```
((((ALUMNO SELECTED_ON (NOMBRE = "JUAN CARDINI"))
   PROJECTED_TO
   CVE-ALUMNO)
  JOINED_TO   INSCRIPCION)
 PROJECTED_TO   CALIFICACION)
```

EJEMPLO 5.3.2.

¿Qué promedio tienen los alumnos de 2o. año?

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (ALUMNO)
Atributo solucion> (PROMEDIO)
Entidades fijas>   (GRUPO)

*** DATOS SOBRE GRUPO ***
Atributo>   GRADO
Comparador> =
Valor>      "2o."
```

Semantica que relaciona GRUPO con (ALUMNO):
(ALUMNO PERTENECE-A GRUPO)

ETAPA DE SALIDA.

```
ENTIDADES FINALES: (GRUPO)
VIAJE FINAL      : ((C E1))
```

Codificacion:

```
((((((GRUPO SELECTED_ON (GRADO = "2o.")) PROJECTED_TO CVE-GRUPO)
     JOINED_TO
     INSCRIPCION)
    PROJECTED_TO    CVE-ALUMNO)
   JOINED_TO    ALUMNO)
  PROJECTED_TO    PROMEDIO)
```

EJEMPLO 5.3.3.

¿Qué profesores imparten Matemáticas?

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (PROFESOR)
Atributo solucion> (NOMBRE)
Entidades fijas>   (MATERIA)

*** DATOS SOBRE MATERIA ***
Atributo>    NOMBRE
Comparador> =
Valor>       "MATEMATICAS"

Semantica que relaciona MATERIA con (PROFESOR):
 (PROFESOR IMPARTE MATERIA)

ETAPA DE SALIDA.


ENTIDADES FINALES: (MATERIA)
VIAJE FINAL        : ((A B1))

Codificacion:

((((((MATERIA SELECTED_ON (NOMBRE = "MATEMATICAS"))
      PROJECTED_TO
      CVE-MATERIA)
     JOINED_TO    GRUPO)
    PROJECTED_TO    CVE-PROFESOR)
   JOINED_TO    PROFESOR)
  PROJECTED_TO    NOMBRE)
```

EJEMPLO 5.3.4.

¿Cuáles fueron los profesores de segundo año de Gomez?

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (PROFESOR)
Atributo solucion> (NOMBRE)
Entidades fijas>   (GRUPO ALUMNO)

*** DATOS SOBRE GRUPO ***
Atributo>    GRADO
Comparador> =
Valor>       "2o."

*** DATOS SOBRE ALUMNO ***
Atributo>    NOMBRE
Comparador> =
Valor>       "GOMEZ"

Semantica que relaciona GRUPO con (PROFESOR):
 (PROFESOR ENSENA-EN GRUPO)

Semantica que relaciona ALUMNO con (PROFESOR):
 (PROFESOR ENSENA-A ALUMNO)


ETAPA DE SALIDA.


ENTIDADES FINALES: (GRUPO ALUMNO)
VIAJE FINAL      : ((((SEL) (E C1)) B1))

Codificacion:

(((((((((ALUMNO SELECTED_ON (NOMBRE = "GOMEZ"))
         PROJECTED_TO
         CVE-ALUMNO)
        JOINED_TO
        INSCRIPCION)
       PROJECTED_TO   CVE-GRUPO)
      JOINED_TO   GRUPO)
     SELECTED_ON (GRADO = "2o."))
    PROJECTED_TO   CVE-PROFESOR)
   JOINED_TO   PROFESOR)
  PROJECTED_TO   NOMBRE)
```

EJEMPLO 5.3.5.

¿Quién le enseñó matemáticas a Viciedo?

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (PROFESOR)
Atributo solucion> (NOMBRE)
Entidades fijas>   (MATERIA ALUMNO)

*** DATOS SOBRE MATERIA ***
Atributo>    NOMBRE
Comparador> =
Valor>       "MATEMATICAS"

*** DATOS SOBRE ALUMNO ***
Atributo>    NOMBRE
Comparador> =
Valor>       "VICIEDO"
```

Semantica que relaciona MATERIA con (PROFESOR):
(PROFESOR IMPARTE MATERIA)

Semantica que relaciona ALUMNO con (PROFESOR):
(PROFESOR ENSENA-A ALUMNO)

ETAPA DE SALIDA.

```
ENTIDADES FINALES: (MATERIA ALUMNO)
VIAJE FINAL      : ((((A) (E C1)) B1))
```

Codificacion:

```
(((((((MATERIA SELECTED_ON (NOMBRE = "MATEMATICAS"))
       PROJECTED_TO
       CVE-MATERIA)
      JOINED_TO   GRUPO)
     INTERSECT_WITH
     (((((ALUMNO SELECTED_ON (NOMBRE = "VICIEDO"))
         PROJECTED_TO
         CVE-ALUMNO)
        JOINED_TO
        INSCRIPCION)
       PROJECTED_TO   CVE-GRUPO)
      JOINED_TO   GRUPO))
    PROJECTED_TO   CVE-PROFESOR)
   JOINED_TO   PROFESOR)
  PROJECTED_TO   NOMBRE)
```

B.2 SECCION COMPLEMENTARIA.

B.2.1 EJEMPLO B.1.

Las dos interacciones siguientes se refieren a la misma pregunta:

¿Qué alumnos toman Matemáticas?

La primera de ellas emplea el enlace semántico puro entre ALUMNO y MATERIA que pasa por GRUPO.

En la segunda interacción se emplea la misma semántica expresada en función de enlaces semánticos más elementales.

Como se puede observar, la respuesta obtenida en ambas interacciones es exactamente la misma, lo que demuestra dos cosas:

- El analizador semántico acepta caminos indicados mediante semánticas compuestas y los maneja correctamente.
- Un enlace semántico puro es equivalente a la expresión del mismo camino mediante enlaces semánticos básicos.

PRIMERA INTERACCION.

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (ALUMNO)
Atributo solucion> (NOMBRE)
Entidades fijas>   (MATERIA)

*** DATOS SOBRE MATERIA ***
Atributo>    NOMBRE
Comparador> =
Valor>       "MATEMATICAS"

Semantica que relaciona MATERIA con (ALUMNO):
 (ALUMNO TOMA MATERIA)


ETAPA DE SALIDA.


ENTIDADES FINALES: (MATERIA)
VIAJE FINAL      : ((A C E1))

Codificacion:

((((((((MATERIA SELECTED_ON (NOMBRE = "MATEMATICAS"))
       PROJECTED_TO
       CVE-MAT)
      JOINED_TO   GRUPO)
     PROJECTED_TO   CVE-GRUPO)
    JOINED_TO   INSCRIPCION)
   PROJECTED_TO   CVE-AL)
  JOINED_TO   ALUMNO)
 PROJECTED_TO   NOMBRE)
```

SEGUNDA INTERACCION.

ETAPA DE INTERACCION.

```
*** ANALISIS SEMANTICO ***
Entidad solucion>  (ALUMNO)
Atributo solucion> (NOMBRE)
Entidades fijas>   (MATERIA)

*** DATOS SOBRE MATERIA ***
Atributo>    NOMBRE
Comparador> =
Valor>       "MATEMATICAS"

Semantica que relaciona MATERIA con (ALUMNO):
 ((ALUMNO PERTENECE-A GRUPO) (GRUPO TOMA MATERIA))


ETAPA DE SALIDA.


ENTIDADES FINALES: (MATERIA)
VIAJE FINAL      : ((A C E1))

Codificacion:

(((((((MATERIA SELECTED_ON (NOMBRE = "MATEMATICAS"))
      PROJECTED_TO
       CVE-MATERIA)
      JOINED_TO
      GRUPO)
     PROJECTED_TO   CVE-GRUPO)
    JOINED_TO   INSCRIPCION)
   PROJECTED_TO   CVE-ALUMNO)
  JOINED_TO   ALUMNO)
 PROJECTED_TO   NOMBRE)
```

B.2.2 EJEMPLO B.2.

La siguiente interacción corresponde al análisis semántico de la pregunta:

¿Qué alumnos son asesorados por los profesores de Juan Perez?

y en ella se muestra la posibilidad de expresar ciclos en función de los enlaces semánticos que los componen.

```
ETAPA DE INTERACCION.

*** ANALISIS SEMANTICO ***
Entidad solucion>  (ALUMNO)
Atributo solucion> (NOMBRE)
Entidades fijas>   (ALUMNO)

*** DATOS SOBRE ALUMNO ***
Atributo>    NOMBRE
Comparador> =
Valor>       "Juan Perez"

Semantica que relaciona ALUMNO con (ALUMNO):
 ((ALUMNO TIENE PROFESOR) (PROFESOR ASESORA ALUMNO))


ETAPA DE SALIDA.

ENTIDADES FINALES: (ALUMNO)
VIAJE FINAL      : ((E C1 B1 D))
```

Codificacion:

```
(((((((((ALUMNO SELECTED_ON (NOMBRE = "Juan Perez"))
          PROJECTED_TO
          CVE-ALUMNO)
         JOINED_TO
         INSCRIPCION)
        PROJECTED_TO    CVE-GRUPO)
       JOINED_TO    GRUPO)
      PROJECTED_TO    CVE-PROFESOR)
     JOINED_TO    PROFESOR)
   PROJECTED_TO    CVE-PROFESOR)
  JOINED_TO    ALUMNO)
 PROJECTED_TO    NOMBRE)
```