



Universidad Nacional Autónoma de México

FACULTAD DE INGENIERÍA

**Sistema Inteligente para la gestión
de información Bibliográfica
sobre etnografía en México**

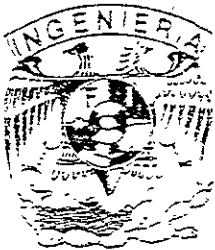
T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A N:

**OMAR CALVILLO RODRÍGUEZ
FABIÁN MERCADO CASTILLO**

DIRECTOR DE TESIS:
M.I. NICOLAS KEMPER VALVERDE



MEXICO, D.F.

2000

280923



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Dedicatorias

A mis padres, por su devoción para enseñarme
el camino que debo seguir. Los admiro.

A mi amigo Alejandro, por su eterna compañía.

Fabián

Dedico esta tesis a mis padres que me han
brindado su apoyo, comprensión, consejo y amor.

Siempre los llevaré en mi corazón.

Gracias papá y mamá.

Omar

Agradecimientos

A mi Universidad:

Por abrirme sus puertas y permitirme tener una formación profesional íntegra.

A mi Facultad:

Por facilitarme sus instalaciones para realizar mis estudios.

A mi asesor de tesis Nicolás Kemper:

Por la confianza me brindó en la realización de este trabajo.

A mis profesores:

Por todo los conocimientos que me transmitieron.

A toda mi familia:

Por sus palabras de aliento y deseos de buena suerte.

A mis amigos:

Por todo su apoyo y consejos.

Fabián

A Dios:

Por ser el guía que me ha permitido llegar a este punto de la vida.

A mi Familia:

Por el apoyo y cariño que siempre me han dado.

A mis amigos:

Por su amistad, porque ellos son los hermanos que da la vida.

A M.I. Nicolás Kemper Valverde:

Por su amistad, apoyo y todo el conocimiento que compartió para realizar este proyecto ya que sin su ayuda esto no hubiera sido posible.

A mis profesores:

Por transmitir la experiencia y conocimientos necesarios para realizar mi formación profesional.

A la UNAM y a la Facultad de Ingeniería:

Por permitir realizar mi carrera profesional en sus aulas.

Omar

TABLA DE CONTENIDO

INTRODUCCIÓN	1
1. INTELIGENCIA ARTIFICIAL	3
1.1 Introducción	3
1.2 Antecedentes	4
1.3 ¿Qué es la inteligencia artificial?	4
1.4 Agentes inteligentes	7
1.4.1 Estructura	8
1.4.2 Programas de agentes	8
1.5 Diferencia entre programación convencional y programación en IA	9
1.6 Técnicas	9
1.7 Aplicaciones	10
2. SISTEMAS EXPERTOS	11
2.1 Introducción	11
2.2 <i>Tipología de problemas</i>	12
2.3 Ingeniería del conocimiento en sistemas expertos	13
2.3.1 Adquisición del conocimiento	13
2.3.2 Representación del conocimiento	15
2.3.3 Razonamiento	18
2.4 Estructura	20
2.4.1 Base de conocimiento	20
2.4.2 Máquina de inferencias	20
2.4.3 Interfaz de usuario	20
2.5 ¿Cuándo usar un sistema experto?	21
2.6 Ventajas	22
2.7 Problemas y limitaciones	23

3. INGENIERÍA DE SOFTWARE	25
3.1 Introducción.....	25
3.2 Definición.....	26
3.3 Sistema de información	27
3.4 Ciclo de vida de los sistemas	27
3.4.1 Análisis	27
3.4.2 Diseño	30
3.4.3 Desarrollo.....	32
3.4.5 Mantenimiento	34
4. BASES DE DATOS.....	35
4.1 Introducción.....	35
4.2 Sistema de bases de datos	35
4.3 Definición.	36
4.4 <i>Arquitectura</i>	36
4.4.1 Nivel interno.....	37
4.4.2 Nivel conceptual.....	37
4.4.3 Nivel externo	37
4.5 Estructuras	38
4.5.1 Enfoque relacional.....	38
4.5.2 Enfoque jerárquico	39
4.5.3 Enfoque de red.....	40
4.6 Modelo entidad/relación	40
4.6.1 Relación uno a muchos.....	40
4.6.2 Relación uno a uno	41
4.6.3 Relación muchos a muchos.....	41
4.7 Normalización	42
4.7.1 Primera forma normal.....	43
4.7.2 Segunda forma normal	43
4.7.3 Tercera forma normal.....	43
4.7.4 Forma normal de Boyce/Codd	44
4.7.5 Cuarto forma normal.....	44
5. DESARROLLO DEL SISTEMA "ETNOBIB"	45

5.1	Análisis del problema	45
5.1.1	Planteamiento del problema.....	45
5.1.2	Análisis de necesidades.....	46
5.1.3	Lenguaje de programación	47
5.2	Conceptualización del sistema	48
5.3	Adquisición del conocimiento	49
5.3.1	Árbol de decisión.....	51
5.4	Diagrama de flujo de datos	52
5.5	Diseño del sistema	60
5.5.1	Base de datos.....	60
5.5.2	Diagrama Entidad/Relación	61
5.5.3	Definición de atributos y tablas.....	62
5.5.4	Diagrama estructurado	76
5.5.5	Máquina de inferencias.....	79
5.5.6	Base de reglas.....	79
5.6	Descripción de las interfaces del sistema	84
5.6.1	Consultas por criterios bibliográficos	84
5.6.2	Consulta experta	85
5.6.3	Resultado de consulta.....	86
5.6.4	Reportes.....	87
5.6.5	Ventana de reportes.....	88
5.6.6	Agregar.....	89
5.6.7	Modificar	90
5.6.8	Eliminar.....	90
5.6.9	Catálogos	91
5.6.10	Reportes de catálogos.....	92
5.7	Pruebas	93
5.7.1	Pruebas de unidad.....	93
5.7.2	Pruebas de integración	93
5.7.3	Pruebas de aceptación.....	93
5.7.4	Validación y verificación del conocimiento.....	93
CONCLUSIONES		95
BIBLIOGRAFÍA.....		97
APÉNDICE.....		98

INTRODUCCIÓN

La presente tesis surge de la necesidad del INAH (Instituto Nacional de Antropología e Historia) de contar con una base de datos que contenga información bibliográfica de las regiones indígenas en México, para poder ser distribuida y consultada en gran parte de la República Mexicana. El objetivo es que toda persona interesada en el tema pueda tener referencias sobre libros, publicaciones periódicas, artículos de libro o tesis.

En colaboración con el Centro de Instrumentos de la Universidad Nacional Autónoma de México, se propone el diseño y desarrollo del "Sistema inteligente para la gestión de información bibliográfica sobre etnografía en México" (ETNOBIB).

Este sistema de información es una herramienta que recopila, clasifica, documenta y mantiene la mayor cantidad de bibliografía sobre etnografía en México. Otra función elemental del sistema ETNOBIB es que tiene la capacidad de realizar búsquedas de fichas bibliográficas por diferentes tipos y cantidades de criterios.

El sistema está formado de seis principales elementos:

- Módulos de captura
- Módulos de actualización
- Módulos de generación de reportes
- Módulos de consultas
- Módulo de seguridad
- Sistema experto

Los módulos de captura y actualización se componen de varios submódulos. El módulo de captura se divide en cuatro submódulos que tienen la función de agregar fichas bibliográficas a la base de datos. El módulo de actualización se divide en ocho submódulos, cuatro para modificar los distintos tipos de documentos que contempla el sistema, y otros cuatro para poder eliminarlos de la base de datos.

El módulo de generación de reportes permite al usuario obtener una impresión en hoja de papel de la consulta realizada a la base de datos. Además, también es posible obtener informes sobre los datos etnográficos y temáticos que se emplean para proporcionar más información a una ficha bibliográfica.

En lo que refiere al módulo de consultas, éste se subdivide en dos submódulos, uno para realizar consultas meramente bibliográficas, y otro para realizar las consultas bibliográficas, etnográficas y/o temáticas.

El módulo de seguridad sirve para restringir el acceso a la base de datos. Existen cuatro distintos tipos de clave: para agregar, para modificar, para eliminar y otra para realizar cualquiera de las acciones anteriores.

El sistema experto tiene como principal función proporcionar resultados de consultas aún cuando no existan fichas bibliográficas con los criterios solicitados. Dichos resultados pretenden brindar al usuario información que mantenga relación con la ficha precisa que desearía encontrar el usuario.

El desarrollo de ETNOBIB involucró la aplicación de tres áreas que pertenecen a la ingeniería en computación: Inteligencia artificial, Ingeniería de software y Bases de datos. Los conceptos teóricos empleados para la realización del sistema se explican en los capítulos siguientes, así como el proceso de análisis, diseño y desarrollo del sistema ETNOBIB.

1.2 ANTECEDENTES

Si bien la inteligencia artificial es un campo joven, es heredera de diversas ideas, puntos de vista y técnicas de otras disciplinas.

Durante más de 2000 años de tradición, con la filosofía han surgido diversas teorías del razonamiento y del aprendizaje, simultáneamente con el punto de vista de que la mente se reduce al funcionamiento de un sistema físico. Los filósofos (desde el año 400 a.C.) permitieron poder pensar en la inteligencia artificial, al concebir a la mente, como una máquina que funciona a partir del conocimiento codificado en un lenguaje interno, al considerar que el pensamiento servía para determinar cuál era la acción correcta que había que emprender.

Durante más de 400 años de matemática, han surgido teorías formales relacionadas con la lógica, probabilidad, toma de decisiones y la computación. Las matemáticas proveyeron las herramientas para manipular las aseveraciones de certeza lógica, así como las inciertas, de tipo probabilista. Asimismo, prepararon el terreno para el manejo del razonamiento con algoritmos.

La psicología ofrece herramientas que permiten la investigación de la mente humana, así como un lenguaje científico para expresar las teorías que se van obteniendo. Los psicólogos reforzaron la idea de que los humanos y otros animales podían ser considerados como máquinas para el procesamiento de información.

La ingeniería de cómputo ofreció el dispositivo que permite hacer realidad las aplicaciones de inteligencia artificial. Los programas de inteligencia artificial por lo general son extensos y no funcionarían sin los grandes avances en velocidad y memoria aportados por la industria de cómputo.

Por último, la lingüística ofrece teorías sobre la estructura y significado del lenguaje. Los lingüistas demostraron que el uso de un lenguaje ajusta dentro de este modelo.

1.3 ¿QUÉ ES LA INTELIGENCIA ARTIFICIAL?

La inteligencia artificial (IA) es una subdivisión de las ciencias de la computación dedicada a crear *hardware* y programas de computadora (*software*) con el objetivo de proporcionar resultados como si fueran producidos por personas.

El concepto de inteligencia artificial se agrupa en cuatro categorías: sistemas que piensan como humanos, sistemas que actúan como humanos, sistemas que piensan racionalmente y sistemas que actúan racionalmente. El enfoque centrado en el comportamiento humano constituye una ciencia empírica, que entraña el empleo de

hipótesis y de la confirmación mediante experimentos. El enfoque racionalista combina matemáticas e ingeniería. A continuación se comentará con mayor detalle cada uno de estos enfoques:

- **Actuar como humano**

Mediante la prueba de Turing se intenta ofrecer una satisfactoria definición operativa de lo que es la inteligencia. Turing definió una conducta inteligente como la capacidad de lograr eficiencia a nivel humano en todas las actividades de tipo cognoscitivo, suficiente para engañar a un evaluador. Brevemente, la prueba consiste en que un humano interrogase a una computadora por medio de un teletipo; la prueba se consideraba aprobada si el evaluador era incapaz de determinar si una computadora o un humano era quien había respondido las preguntas en el otro extremo de la terminal.

El trabajo que implica programar una computadora para pasar la prueba es considerable. La computadora debe ser capaz de:

- Procesar un lenguaje natural para establecer una comunicación satisfactoria.
- Representar el conocimiento para guardar toda la información que se le haya proporcionado antes o durante el interrogatorio.
- Razonar automáticamente con el fin de utilizar la información guardada al responder preguntas y obtener nuevas conclusiones.
- Autoaprendizaje de la máquina, para adaptarse a nuevas circunstancias y para detectar y extrapolar esquemas determinados.

- **Pensar como humano**

Para afirmar que un programa determinado utiliza algún tipo de razonamiento humano, es necesario definir antes cómo piensan los seres humanos. Hay dos formas de hacer esto: mediante la introspección (captar nuestros propios pensamientos conforme estos se van dando) o mediante la realización de experimentos psicológicos. Después se procede a expresar tal teoría en un programa de computadora. Si los datos de entrada/salida del programa y la duración del tiempo de su comportamiento corresponden a los de la conducta humana, existe evidencia de que algunos de los mecanismos del programa también funcionan en los seres humanos. Lo interesante es seguir la pista de los pasos de razonamiento y compararla con la ruta seguida por sujetos humanos a los que se propuso los mismos problemas.

- **Pensar racionalmente**

Aristóteles fue uno de los primeros que intentó codificar la “manera correcta de pensar”, es decir, procesos de pensamiento irrefutables. Sus silogismos son esquemas de estructuras de argumentación mediante las que siempre se llega a conclusiones correctas si se parte de premisas correctas.

El desarrollo de la lógica formal a fines del siglo XIX permitió contar con una notación precisa para representar aseveraciones relacionadas con todo lo que existe en el mundo, así como sus relaciones mutuas. En 1965 existían programas que, contando con tiempo y memoria suficientes, podían describir un problema en notación lógica y encontrarle solución, siempre y cuando esta existiese. En la inteligencia artificial, la tradición logicista se esfuerza por elaborar programas como el anterior para crear sistemas inteligentes.

- **Actuar racionalmente**

Actuar racionalmente implica actuar de manera tal que se logren los objetivos deseados, con base en ciertos supuestos. Un agente es algo capaz de percibir y actuar. De acuerdo con este enfoque, se considera la inteligencia artificial como el estudio y construcción de agentes racionales. Es necesario contar con la capacidad para representar el conocimiento y razonar con base en él, pues de esta manera se podrán tomar decisiones correctas en una amplia gama de situaciones.

Las computadoras agilizan y simplifican algunos aspectos del proceso del pensamiento. La inteligencia artificial va un paso más allá, nos permite automatizar o agilizar tareas complejas que eran hechas manualmente. La combinación de la inteligencia artificial y sistemas de computadora basados en información incrementan las capacidades y aplicaciones de la computadora. Los programas de inteligencia artificial tienen la capacidad de comprender símbolos, términos, conceptos, ideas y relaciones haciendo parecer que tienen, en cierto grado, sentido común.

1.4 AGENTES INTELIGENTES

Un agente es todo aquello que puede considerarse que percibe su ambiente mediante sensores y que responde o actúa en tal ambiente por medio de efectores (ver figura 1.4). En el caso de los agentes robóticos, los sensores son sustituidos por cámaras y telémetros infrarrojos, y los efectores son remplazados mediante motores. En el caso de un agente de software, sus percepciones y acciones vienen a ser las cadenas de bits codificados.

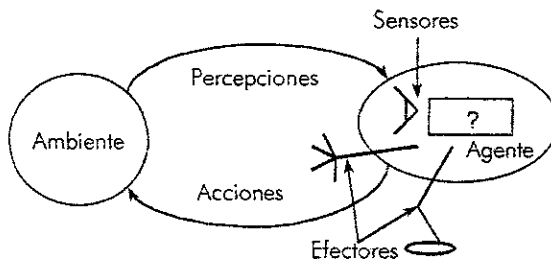


Fig. 1.4: Esquema de un agente

Un agente racional es aquel que hace lo correcto. Lo correcto es aquello que permite al agente obtener el mejor desempeño. El término medición del desempeño se enfoca a dos criterios: 1) el cómo, sirve para definir qué tan exitoso ha sido un agente. La necesidad de contar con una medición objetiva del desempeño debe ser propuesta por una autoridad (experto humano); 2) el cuándo, lo importante es medir el desempeño del agente a largo plazo. Esto podría ser por varias semanas, varios meses o inclusive varios años.

El carácter de racionalidad de lo que se hace en un momento dado dependerá de cuatro factores:

- De la medida con la que se evalúa el grado de éxito logrado.
- De todo lo que hasta ese momento hayo percibido el agente, llamado secuencia de percepciones.
- Del conocimiento que posea el agente acerca del medio.
- De las acciones que el agente pueda emprender.

Lo anterior llevo a definir lo que es un "agente racional ideal": En todos los casos de posibles secuencias de percepciones, un agente racional deberá emprender todas aquellas acciones que favorezcan obtener el máximo de su medida de rendimiento,

basándose en las evidencias aportadas por la secuencia de percepciones y en todo conocimiento incorporado en tal agente.

1.4.1 Estructura

El concepto de agente permite pensar en él como herramienta para el análisis de sistemas. El cometido de la inteligencia artificial es el diseño de un programa de agente: una función que permita implementar el mapeo del agente para pasar de percepciones a acciones. Este programa se ejecutará en algún tipo de dispositivo de cómputo, al que se denominará arquitectura. El programa elegido debe ser aquel que la arquitectura acepte y pueda ejecutar.

$$\text{Agente} = \text{arquitectura} + \text{programa}$$

En general, la arquitectura pone al alcance del programa las percepciones obtenidas mediante los sensores, lo ejecuta y alimenta al efector con las acciones elegidas por el programa conforme estas se van generando.

Lo que realmente interesa es la complejidad de la relación que existe entre la conducta del agente, la secuencia de percepciones que produce el ambiente y las metas que se espera alcance el agente.

1.4.2 Programas de agentes

El esqueleto de todo agente inteligente es la aceptación de percepciones originadas en un ambiente y la generación de acciones respectivas. Los programas de agentes emplean estructuras de datos internos que se irán actualizando con la llegada de nuevas percepciones. Tales estructuras de datos se operarán mediante los procedimientos de toma de decisiones de un agente para generar la elección de una acción, elección que se transferirá a la arquitectura para proceder a su ejecución. Es decisión del agente construir la secuencia de percepciones en la memoria.

1.5 DIFERENCIA ENTRE PROGRAMACIÓN CONVENCIONAL Y PROGRAMACIÓN EN IA

Programación convencional	Programación en IA
Procesan datos y dan información.	Procesan conocimientos y dan conclusiones.
Usan lenguajes procedurales.	Usan lenguajes simbólicos.
Usan algoritmos.	Usan reglas heurísticas.
Usan ciclos, secuencias y decisiones.	Usan reglas de inferencias.
Accesan bases de datos.	Accesan bases de conocimientos.
Están centrados en el analista y programador.	Están centrado en el experto y el usuario.
Manejan datos determinísticos.	Dan conclusiones con información incierta o incompleta.
Se usan principalmente en áreas de tipo administrativo.	Se pueden usar en todas las áreas funcionales de la empresa.
Delimitan el alcance de la función de informática a los sistemas de información y tecnología computacional existente.	Extienden el alcance de la función de informática y permiten definir nuevas estrategias.

De acuerdo a lo anterior, un programa tradicional o convencional está basado en datos numéricos, se ejecuta en forma secuencial y necesita de conocimientos precisos que rara vez se modifican. Este tipo de programas no pueden explicar ni justificar los pasos que siguen, ni la solución a la que llegan, por lo tanto, no poseen la capacidad de realizar rápidamente pequeños prototipos.

1.6 TÉCNICAS

Los programas de inteligencia artificial usan algoritmos convencionales para tareas específicas. Sin embargo, la inteligencia artificial ha desarrollado diferentes técnicas o métodos para la resolución de problemas.

Las técnicas básicas de inteligencia artificial más conocidas son:

- Sistemas expertos (emulan razonamiento).
- Lógica difusa (emula razonamiento aproximado o incertidumbre).
- Redes neuronales (emulan razonamiento y aprendizaje).
- Algoritmos genéticos (emulan razonamiento y aprendizaje).
- Razonamiento basado en casos (emula razonamiento y aprendizaje).

Algunas técnicas integrales (sistemas híbridos) de inteligencia artificial son:

- Robótica
- Multimedia (interfaces inteligentes)
- Realidad virtual

La mejor opción para desarrollar programas de computadora, a los que se les puedan aplicar técnicas de inteligencia artificial, consiste en determinar la estructura del problema y las herramientas con que se cuenta, como: métodos de representación del conocimiento, estructuras de control, lenguajes de programación, editores, compiladores y *hardware*.

1.7 APLICACIONES

Las aplicaciones usadas en donde las técnicas de inteligencia artificial han demostrado con éxito que pueden resolver complicados problemas de forma masiva, se han desarrollado en sistemas que:

- 1) Permiten al usuario preguntar a una base de datos de mejor forma que un lenguaje de programación.
- 2) Reconocen objetos de una escena por medio de aparatos de visión.
- 3) Generan palabras reconocibles como humanas desde textos computarizados.
- 4) Reconocen e interpretan un vocabulario de palabras humanas.
- 5) Resuelven problemas en una variedad de campos, usando conocimientos expertos codificados.

Los países que han dado mayor apoyo a la inteligencia artificial son Estados Unidos, Japón y el Reino Unido; lo han llevado a cabo a través de grandes compañías, así como en universidades, para resolver problemas con un considerable ahorro económico. Las aplicaciones primarias de la inteligencia artificial se clasifican en cuatro campos: sistemas expertos, lenguaje natural, robótica y visión, sistemas sensores y programación automática.

2. SISTEMAS EXPERTOS

2.1 INTRODUCCIÓN

Los sistemas expertos son programas de computadora que utilizan procesos de razonamiento similares a los humanos (en vez de técnicas de computación) para resolver problemas en campos específicos del saber. Estos procesos programados que simulan el razonamiento humano, a su vez, están basados en conocimientos humanos experimentales o habilidades que se codifican en un programa de computadora, en una estructura o conjunto de representaciones denominada base de conocimientos. En la figura 2.1 se puede observar el funcionamiento de un sistema experto.

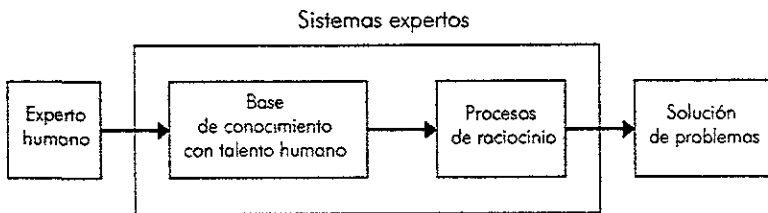


Fig. 2.1: Los sistemas expertos resuelven problemas razonando sobre los conocimientos suministrados por un experto humano.

Los sistemas expertos se basan no sólo en conocimientos sobre hechos reales, como ocurre con los programas convencionales, sino también en observaciones y conocimientos no confirmados basados en experiencia e intuición, llamado conocimiento heurístico. Tanto los hechos como el conocimiento heurístico, pertenecen originalmente a especialistas humanos en campos específicos del saber. Los hechos y el conocimiento heurístico se complementan con métodos de análisis, manejo y aplicación de los conocimientos codificados de modo que el programa pueda hacer inferencias y explicar sus acciones.

Un sistema experto puede razonar sobre los conocimientos que contiene sobre un campo limitado del saber, aproximándose a las prestaciones humanas; no pueden dar soluciones a aquellos que el ser humano todavía no ha logrado resolver.

El desarrollo histórico de la inteligencia artificial ha dado lugar al concepto de sistema experto como resultado de la convergencia de dos líneas clásicas de resolución de problemas:

- 1) La línea heurística, basada en la acumulación de una serie de restricciones limitativas de los procesos de resolución de problemas basados en la búsqueda en espacios de estados o en espacios de problemas.
- 2) La línea de deducción automática, basada en la mecanización de las teorías de interpretación de fórmulas lógicas.

En los años 70's se plantea como paradigma general para la resolución de problemas los sistemas de reglas de producciones. Se propone como representación del conocimiento un conjunto de reglas constituidas por pares "situación-acción", en cuyo antecedente se formulan las precondiciones de aplicación de cada regla y en cuyo consecuente se formulan, como bloques procedurales, las acciones de modificación de una memoria de trabajo, representativa del estado del universo en que se resuelve el problema.

2.2 TIPOLOGÍA DE PROBLEMAS

Para definir una aplicación adecuada para los sistemas expertos, cabría plantear los siguientes rasgos del área del problema como idóneos:

- 1) El procedimiento de resolución empleado sin el sistema experto actualmente debe tener una parte importante de razonamiento, siendo claramente menos relevantes las operaciones de cálculo.
- 2) El proceso de resolución utilizado por los expertos humanos debe ser suficientemente conocido. Su nivel de complejidad debe ser tal que se resuelvan los problemas en plazos razonables, de esta forma el tamaño de dominio de discurso será limitado, y por tanto, manejable al sistema a desarrollar.
- 3) Si se plantea algorítmicamente, el proceso de resolución planteado debe tener un volumen importante de operaciones, de forma que pueda apreciarse la ventaja de una base de conocimiento heurística como elemento que reduzca la cantidad de cálculos algorítmicos.
- 4) El conocimiento debe poder introducirse en forma incremental.
- 5) Su implantación debe aportar un valor añadido suficiente respecto a la forma actual de operación.

Dentro de estos criterios generales, cabe identificar dos tipos de aplicaciones:

- a) Aquellos cuyo objetivo es resolver problemas de clasificación, es decir, decisión de asignación del caso estudiado a un grupo de soluciones o grupo de decisiones recomendables, siendo estos grupos definidos previamente.
- b) Aquellos cuyo objetivo es proponer respuestas previamente no clasificadas, sino que se construyen directamente por la unión de componentes básicos de la base de conocimiento.

2.3 INGENIERÍA DEL CONOCIMIENTO EN SISTEMAS EXPERTOS

La ingeniería del conocimiento es la herramienta fundamental para el desarrollo de la inteligencia artificial. Es un conjunto de técnicas y criterios encaminados a producir la incorporación del conocimiento procesable por las distintas formas de representación y constituye un concepto paralelo de la metodología de la programación en las aplicaciones clásicas.

En el campo de la ingeniería del conocimiento existen tres conceptos básicos:

- 1) *Datos*: término que se refiere a un conjunto de caracteres alfanuméricos que por si solos no tienen significado alguno para el usuario.
- 2) *Información*: es una organización de datos la cual tiene un significado para el usuario.
- 3) *Conocimiento*: es la organización de la información para la solución de un problema, lo cual a su vez genera nuevo conocimiento.

El ingeniero del conocimiento es aquel que realiza investigaciones en un dominio en particular y crea una representación formal de los objetos y relaciones del dominio. Normalmente entrevista a los verdaderos expertos de un campo para que estos le enseñen lo necesario sobre el dominio de interés y le deslinden las fronteras del conocimiento respectivo. También debe poseer suficiente dominio de un lenguaje para representar el conocimiento, con el fin de que pueda codificar adecuadamente tales hechos. Además debe contar con un conocimiento amplio de la implantación del procedimiento de inferencia, que garantice la respuesta de las consultas en un tiempo razonable.

La ingeniería del conocimiento tiene tres fases principales: adquisición del conocimiento, representación del conocimiento y razonamiento.

2.3.1 Adquisición del conocimiento

Es el proceso de captura, selección, estructuración, organización y validación del conocimiento a partir de una o más fuentes. A este proceso se le considera el "cuello de botella" en el desarrollo de un sistema inteligente, ya que por lo general involucra del 60 al 70% del tiempo total de desarrollo.

Durante la adquisición del conocimiento es necesario identificar:

- Fuentes del conocimiento
 - a) Públicas – libros, revistas, documentales, videos, periódicos, mapas, fotografías, tesis, sonidos, diagramas, etc.
 - b) Privadas – experto humano.

- Tipos de conocimiento
 - a) Declarativo – cuando se declaran hechos, conceptos o propiedades de un dominio particular. Expresa las relaciones lógicas o empíricas entre dos o más elementos del dominio.
 - b) Procedural – cuando se requiere de una secuencia de pasos a seguir para obtener un resultado específico. Este conocimiento es información respecto a cómo aplicar conocimiento declarativo en la solución de problemas.
- Categorías de conocimiento
 - a) Superficial – combina conocimiento declarativo y procedural en las heurísticas que un experto humano emplea para resolver problemas en un dominio específico.
 - b) Profundo – consiste de conocimiento fundamental de un dominio; incluyendo definiciones, axiomas, leyes generales, principios y relaciones causales. Requiere de análisis científico y técnico.
- Métodos de adquisición del conocimiento
 - a) Manuales – se puede adquirir el conocimiento del experto humano mediante entrevistas, observación, exámenes, protocolo de análisis, informes, etc.
 - b) Automáticos – es una entrevista estructurada en el que sistemáticamente se van solicitando al experto los conocimientos que se necesitan. Se emplea cuando el experto no está disponible el tiempo necesario.
- Mapas del conocimiento

Es un diagrama que permite plasmar las diferentes relaciones lógicas que emplea un experto humano para resolver un problema. Existen las tablas de decisión, listas, árboles de decisión, diagramas de precedencia, etc. La amplitud o tamaño de cada mapa de conocimiento depende del nivel de abstracción o de detalle que se quiera dar al análisis y solución del problema. La figura 2.3.1 muestra la estructura de un árbol de decisión:

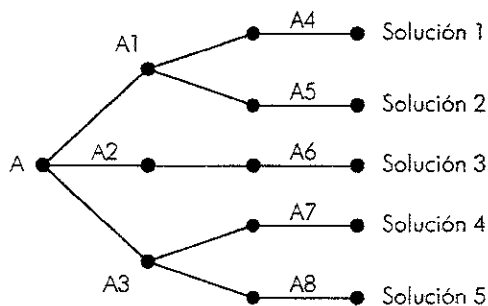


Fig. 2.3.1: Estructura de un árbol de decisión

El ingeniero del conocimiento deber ser una persona que sepa de informática, inteligencia artificial, psicología y además que tenga cierto conocimiento del área de aplicación del experto, así podrá convertir los conocimientos adquiridos en las entrevistas en información para la computadora.

El proceso de la adquisición del conocimiento es arduo y extenso porque, independientemente de la cooperación del experto para comunicar su saber, el ingeniero del conocimiento debe adquirir el saber de todas las formas posibles.

El ingeniero del conocimiento ha de saber escuchar y preguntar, e inspirar confianza y extraer los conocimientos al experto independientemente de las diferencias de carácter y personalidad. Debe de comprender que, puesto que casi todo el saber del especialista es intuitivo e inconsciente, se puede aprender de él, tanto por preguntas directas como por observación y convivencia, es decir, de modo aleatorio.

Verbalizar conocimientos heurísticos e intuitivos con precisión y en una forma que otra persona pueda usar, es una tarea realmente difícil. La dificultad se hace aún mayor dado el abismo existente entre el modo en que el experto humano expresa sus conocimientos y lo que la computadora puede realmente utilizar.

Para obtener conocimiento de un especialista en forma utilizable, hay que tener una idea clara de lo que se busca. Saber lo que se busca es mucho más importante que los conocimientos sobre un determinado campo de aplicación o sobre los elementos más importantes de dicho campo. Significa también la capacidad para comprender el aspecto que dicho conocimiento puede tener a fin de poder identificarlo, organizarlo e imponer una imagen o estructura general del conocimiento en el saber específico que el experto está comunicando.

2.3.2 Representación del conocimiento

Los esfuerzos de la inteligencia artificial apuntaban a descubrir un pequeño conjunto de técnicas de razonamiento potentes que pudieran resolver muchos problemas. Algunos investigadores han encontrado maneras eficientes y eficaces de representar el conocimiento para reducir la búsqueda requerida por un programa. Los métodos de representación del conocimiento son combinaciones de estructuras de datos que junto con procedimientos de interpretación, ayudan a hacer inferencias sobre los datos almacenados. La representación del conocimiento brinda a la computadora una descripción de cómo interactúan los distintos grupos de información, así como para procesarla y/o interpretarla.

El objetivo es elegir una representación que facilite el trabajo en un área particular. Esto se puede extender para combinar diferentes representaciones dentro de un solo sistema, o desarrollar una nueva representación o una variación de alguna existente si se ajusta mejor al dominio del problema.

Algunas formas de representación del conocimiento son: reglas de producción, *frames*, redes semánticas y cálculo de predicados. A continuación se explica más a detalle los *frames* y las reglas de producción.

- *Frames*

Es una estructura de datos que incluye todo el conocimiento acerca de un objeto particular. Dicho conocimiento está organizado en una estructura jerárquica. Cada *frame* constituye un objeto, el cual puede representar a un objeto físico, un evento, una localización, una situación, u otro elemento.

Para representar un *frame*, el conocimiento es particionado en *slots*. Cada *slot* describe los atributos y características del objeto, y puede incluir tanto conocimiento declarativo como procedural. Para poder describir completamente a un objeto debemos conocer todos sus atributos relevantes, y para cada atributo se debe establecer un valor. Cada *slot* está compuesto de facetas. A través de una faceta se puede obtener el valor de un atributo del *frame*. Las facetas pueden ser:

- *Valor* – describe el atributo.
- *Default* – es usado si el *slot* está vacío, sin descripción.
- *Rango* – indica qué clase de información puede aparecer en el *slot*.
- *If-added* – contiene conocimiento procedural o de enlace. Especifica una acción que debe ser tomada cuando un valor en el *slot* es agregado o modificado.
- *If-needed* – se usa cuando el valor del *slot* no está dado. Cuando se activa este procedimiento, el valor del *slot* debe ser calculado, extraído de una base de datos, o solicitado al usuario.
- *Otros* – un mismo *slot* puede contener *frames*, reglas, red semántica o cualquier tipo de información.

En la figura 2.3.2 se muestra un ejemplo del *frame*: Automóvil.

Facetas						
Slot	Valor	Default	Rango	If-added	If-needed	Otros
Color	Rojo		Rojo, blanco, azul			
Marca	Ford		Ford, Honda, Nissan			
Precio	120,000		Pesos			
Equipamiento		No	Sí, no	Almacenar		
Transmisión	Manual		Manual, automático			
n° de puenas		4	2, 4, 5	Almacenar		
Seguro			Total, parcial		R1	

Fig. 2.3.2: Ejemplo de un *frame*.

- Reglas de producción

A menudo es conveniente representar la naturaleza dinámica de una aplicación de inteligencia artificial por un conjunto de reglas hechas de condiciones y acciones. Esta es probablemente la forma más popular de representación del conocimiento utilizada en los sistemas expertos. Esto se debe a la cómoda estructura que las mismas poseen y a la forma tan natural en que pueden ser expresados los conocimientos.

Las reglas de producción son un tipo de regla "Si... entonces...", basada en condiciones y acciones. Las descripciones de una aplicación dada o contexto de un problema son unidas en una colección de condiciones, en unas reglas que hacen que las acciones de la regla a ejecutar den lugar a nuevas descripciones que producen más acciones, y así hasta que el sistema encuentre la solución o se detenga.

Un sistema consistente en un conjunto de reglas de producción se denomina "sistema de producción". Las reglas de producción son los operadores en el sistema, es decir, las que se usan para manipular las bases de datos.

Cada regla consta de una parte "C" llamada antecedente y de una parte "A" llamada consecuente, y es de la forma:

Si C entonces A

En una regla de producción el antecedente está compuesto por una o varias proposiciones, las cuales se combinan mediante conectores para formar una condición o premisa más compleja. Los conectores lógicos utilizados son los clásicos de la lógica del cálculo proposicional:

- 1) *Conjunción* – representa la condición del cumplimiento conjunto de dos proposiciones. Es decir, una conjunción simple se cumple sólo cuando las dos proposiciones componentes son verdaderas.
- 2) *Disyunción* – representa el caso de una alternativa entre dos proposiciones. Para que una disyunción inclusiva se cumpla basta que una de las dos proposiciones componentes sea verdadera.
- 3) *Negación* – significa la satisfacción o el cumplimiento de lo contrario que ha sido expresado en la proposición primaria. Esto es, si es verdadero, la negación es falsa, y viceversa. La negación puede ser aplicada a proposiciones simples como compuestas.

De forma general, el consecuente en una regla de producción está compuesto por una proposición simple. Puede significar cosas distintas, por ejemplo puede significar la orden de añadir algo a la base de datos acerca del problema que se esté considerando, sugerir que el usuario asuma una tarea determinada, etc

2.3.3 Razonamiento

Toda aplicación en inteligencia artificial requiere de conocimientos y métodos de inferencia para obtener soluciones. Los esquemas de representación del conocimiento permiten dotar a los sistemas expertos de conocimientos, mientras que los métodos de inferencia permiten a éstos manipular de forma apropiada y eficiente dicho conocimiento para alcanzar las metas propuestas.

Así pues, el razonamiento (máquina o motor de inferencia) es el mecanismo de selección de aplicación de reglas; es el cerebro del sistema experto. Principalmente hay dos enfoques:

- Enfoque guiado por datos

También es conocido como enfoque sintético (*bottom-up*), encadenamiento hacia adelante (*forward chaining*) o razonamiento antecedente. En este método las reglas son aplicables solamente si su parte condición es satisfecha por la base de datos. Al utilizar este método, se comienza entrando datos del problema en curso a la base de datos.

El siguiente procedimiento recursivo, escrito en pseudocódigo, bosqueja la estructura de un módulo sencillo de aplicación de reglas:

```

procedimiento generar;
comenzar
  identificar el conjunto S de reglas aplicables;
  mientras S no sea vacío
    comenzar
      seleccionar una regla R de S;
      aplicar R;
      si se soluciona el problema aplicando R
        entonces indicar ÉXITO
      en otro caso llamar "generar" recursivamente
    eliminar R de S y anular el efecto de aplicar R
  acabar
acabar;

```

El procedimiento "generar" produce nuevos hechos a partir de la base de datos utilizando reglas. Después se añaden estos hechos a la base de datos.

Entre las ventajas que tiene este enfoque se incluyen su simplicidad y el hecho de que puede utilizarse para proporcionar todas las soluciones a un problema dado.

Una desventaja del enfoque guiado por datos es que el comportamiento del sistema, al intentar solucionar un problema, puede ser ineficaz y puede parecer

también desatinado porque algunas reglas ejecutadas podrían no estar relacionadas con el problema en cuestión.

- Enfoque guiado por objetivos

Se le conoce también como enfoque analítico, encadenamiento hacia atrás o razonamiento consciente. En este tipo de enfoque, el sistema centra su atención únicamente en las reglas que sean relevantes para el problema en curso. La función que se muestra a continuación (escrita en pseudocódigo) bosqueja la estructura de un módulo simple de aplicación de reglas:

función validar

var resultado

comenzar

resultado = falso;

rastrear la base de reglas para identificar el conjunto de reglas S aplicables que tengan X en el miembro derecho;

si S es vacío

entonces pedir al usuario que añada algunas reglas a S;

mientras (resultado = falso) y (S no es vacío)

comenzar (ciclo)

seleccionar y eliminar una regla R de S;

C = la parte de condición R;

si C es verdadero en la base de datos

entonces resultado = verdadero

en otro caso si C es falso en la base de datos

entonces no hacer nada

en otro caso si validar C es verdadero

entonces resultado = verdadero

acabar;

validar = resultado

acabar;

La principal ventaja que tiene es que no solicita datos y no aplica reglas que no estén relacionadas con el problema en cuestión. Este enfoque es el que se ha de elegir si se utiliza un sistema de producción para comprobar una hipótesis concreta, sin embargo, puede utilizarse también para comprobar cada hipótesis posible, proporcionando por tanto todas las respuestas que se pueden hacer utilizando un enfoque guiado por datos.

- Métodos mixtos

Los enfoques llamados guiados por datos y guiados por objetivos pueden combinarse de varias maneras. Lo específico del conocimiento y la clase de problemas a resolver puede hacer conveniente el diseño expreso de la estrategia de

control más adecuada; por ello, para una misma base de conocimiento cabe plantear varios motores de inferencia adaptados a diversos tipos de problemas.

2.4 ESTRUCTURA

La estructura de un sistema experto se forma de tres elementos principales:

2.4.1 Base de conocimiento

Es un conjunto de representaciones de ciertos hechos acerca del mundo. Consiste en una base de datos que se utiliza para almacenar los datos (hechos) del problema que se esté considerando; también contiene una base de reglas, las cuales representan el conocimiento general acerca del dominio del problema.

Una buena base de conocimiento debe ser clara. Es necesario que estén definidas todas las relaciones de verdadera importancia, y suprimir todo detalle irrelevante. Esto permitirá al creador de la base de conocimientos ocuparse sólo del contenido y deberá obtener siempre las mismas respuestas, independientemente de cómo esté codificado el lenguaje.

2.4.2 Máquina de inferencias

Este módulo es capaz de producir las respuestas del sistema a partir de los datos de planteamiento de las mismas y de la base de conocimiento; busca, selecciona y aplica reglas que puedan producir cambios y/o adiciones a la base de datos.

2.4.3 Interfaz de usuario

Conjunto de pantallas necesarias para la visualización y navegación del sistema

La siguiente figura 2.4 esquematiza la estructura:

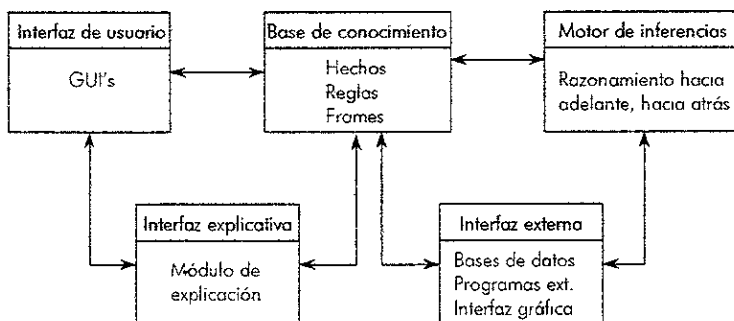


Fig. 2.4: Estructura de un sistema experto

Los módulos de interfaz explicativa e interfaz externa pueden no estar presentes en algunos sistemas expertos. El módulo de explicaciones proporciona al usuario descripciones acerca del proceso de inferencia, por ejemplo: ¿cómo se logró una conclusión? o ¿por qué se requiere cierta información?. La interfaz externa existe cuando el sistema experto extrae información de alguna base de datos externa, o bien, se complementa con algún otro sistema.

Esta estructura ofrece un mismo conocimiento que puede utilizarse para obtener respuestas a distintos problemas, lo que no ocurre con la estructura clásica de las aplicaciones algorítmicas en las que, tanto conocimiento como procedimiento, están integrados en un único problema orientado a una clase más restringida de problemas.

Los conocimientos del experto humano se guardan en la base de conocimiento, el mecanismo inferencial hace lo que su nombre implica, inferencias, es decir, es el proceso por el que se llega a una conclusión en base a una serie inicial de proposiciones. Ver figura 2.4.1.

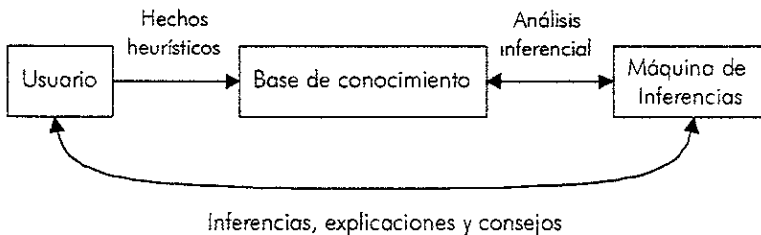


Fig. 2.4.1: Los sistemas expertos infieren y aconsejan en base a su interpretación de los conocimientos de los expertos humanos.

2.5 ¿CUÁNDO USAR UN SISTEMA EXPERTO?

Un problema se presta a ser considerado como un sistema experto cuando:

- La solución del problema tiene una rentabilidad tan alta que justifica el desarrollo de un sistema.
- El problema puede resolverse sólo por un conocimiento experto que puede dar una forma a los conocimientos necesarios para resolver el problema y la intervención de este experto dará al sistema la experiencia que necesita.
- El problema puede resolverse solamente por un conocimiento experto en vez de usar algoritmos particulares.

- Se tiene acceso a un experto que puede dar forma a los conocimientos necesarios para resolver el problema. La intervención de este experto dará al sistema la experiencia que necesita.
- El problema cambia rápidamente, o bien el conocimiento es el que cambia rápidamente, o sus soluciones son las que cambian constantemente.
- El problema no puede tener una solución única. Los sistemas expertos funcionan mejor con los problemas que tienen un cierto número de soluciones interceptables.

El desarrollo de un sistema experto no se considera que está acabado una vez que funciona este, sino que se continúa desarrollando y actualizando tanto el conocimiento del sistema como los métodos de procesamiento, quedando reflejados los progresos o modificaciones en el campo, área o sistema.

Los sistemas expertos son programas que contienen la erudición de un especialista humano versado en un determinado campo de aplicación. En este sentido, los expertos humanos escasean y su contratación supone una enorme inversión económica.

La contención de un auténtico especialista humano es un gran problema, entonces la alternativa es el desarrollo de un sistema experto, esto es, un modelo computarizado de las capacidades de razonamiento y habilidades en resolución de problemas del especialista humano, que puede resolver cuestiones mediante procedimientos similares a los que utilizamos los seres humanos.

2.6 VENTAJAS

Los sistemas expertos pueden proporcionar los siguientes beneficios:

- Incrementar el rendimiento y la productividad. Pueden trabajar más rápido que los seres humanos, reduciendo costos y cantidad de personal.
- Incrementar la calidad. Pueden incrementar la calidad proporcionando un consejo consistente y reduciendo el margen de error.
- Reducción de tiempos. Muchos son usados para diagnosticar malos funcionamientos y prescribir reparaciones, además de reducir tiempos significativamente.
- Captura de experiencia escasa. La escasez de experiencia viene a ser evidente en situaciones donde no existen muchos expertos para una determinada tarea.
- Flexibilidad. Ofrecen flexibilidad en el campo de la manufactura o en el de la producción de servicios.
- Eliminación de la necesidad de adquirir equipos costosos. En muchas situaciones, un experto humano podría realizar complejas tareas de monitoreo y control. En el caso de que el experto humano no estuviera disponible, se requeriría la adquisición de equipo caro que realizaría las mismas funciones. Los sistemas expertos pueden realizar las mismas tareas con instrumentos de más bajo costo porque puede

realizar sondeos complejos y rápidos de la información proporcionada por los instrumentos.

- Operación en ambientes peligrosos. Muchas tareas requieren expertos humanos que deben operar en ambientes peligrosos. Los sistemas expertos podrían evitar a los humanos tales ambientes, como involucrarse en conflictos militares o trabajar en condiciones de extrema humedad, calor, frío o ambientes tóxicos.
- Accesibilidad al conocimiento y ayuda a nivel de escritorio. La gente puede preguntar a los sistemas y recibir consejos. Una tarea de aplicabilidad son tutoriales que actúan a la vez como soporte técnico.
- Confiabilidad. Consistentemente ponen atención a todos los detalles y no pasan por alto información relevante.
- Incrementar las capacidades de otros equipos de cómputo. La integración de sistemas expertos con otros sistemas hace a estos más efectivos y los hace proporcionar resultados de mayor calidad.
- Integración de varios expertos. En ciertos casos integran las opiniones de varios expertos y esto podría incrementar la calidad de las conclusiones o resultados.
- Habilidad para trabajar con información incompleta o desconocida. A diferencia de los sistemas tradicionales de cómputo, los sistemas expertos pueden trabajar con información incompleta o ambigua; con probabilidad y estadística.
- Proporciona entrenamiento. Principiantes que trabajan con sistemas expertos se hacen cada vez más experimentados.
- Mejoramiento de la solución de problemas. Incluyen los elementos de juicio de las personas más expertas dentro del análisis.
- Habilidad para resolver problemas complejos. Algunos sistemas expertos ya son capaces de resolver problemas donde el ámbito de conocimiento excede por mucho el de un individuo. Sin embargo, estos problemas tienen un dominio bastante restringido.
- Transferencia del conocimiento a lugares remotos. Uno de los grandes potenciales de los sistemas expertos es que es fácil transferir el conocimiento a través de las fronteras internacionales. Esto podría ser muy importante para ayudar al desarrollo de países que no pueden pagar por el conocimiento poseído por expertos humanos.

2.7 PROBLEMAS Y LIMITACIONES

Las técnicas y métodos disponibles para el desarrollo de sistemas expertos no son del todo consistentes ni efectivos, ni siquiera para aquellos problemas de las categorías genéricas. Para aplicaciones de modesta complejidad, algunos códigos de sistemas expertos, especialmente aquellos construidos con lenguajes de programación procedural, son generalmente difíciles de entender, depurar, extender y mantener.

A continuación se mencionan algunos factores y problemas que han frenado el desarrollo comercial de los sistemas expertos:

- No todas las veces el conocimiento está disponible fácilmente.
- Es difícil de extraer la experiencia de los expertos humanos.
- Los sistemas expertos trabajan adecuadamente solo en un dominio restringido.
- El método de solución de cada experto humano varía.
- El conocimiento frecuentemente está incompleto y a veces su fundamento es poco científico.
- Probar y refinar el conocimiento es complicado.
- Muchos expertos no tienen medios independientes de verificar si sus conclusiones son razonables.
- El vocabulario que los expertos usan para expresar hechos y relaciones es frecuentemente limitado y no es comprensible para otros.
- Se necesita ayuda de ingenieros del conocimiento, quienes son escasos y sus servicios son costosos. Esto es un hecho que podría significar que el precio del desarrollo de un sistema experto se incremente de manera considerable.
- Una de las principales barreras es la falta de confianza de los usuarios finales en los sistemas expertos.

Muchas de estas limitaciones desaparecerán con los avances tecnológicos a través del tiempo.

3. INGENIERÍA DE SOFTWARE

3.1 INTRODUCCIÓN

Durante las primeras tres décadas de la era de la computación, el primer reto fue desarrollar *hardware* que redujera el costo del proceso y almacenado de datos. Durante los años 80's, los avances en la microelectrónica resultaron en *hardware* poderoso que redujo los costos. En nuestros días el problema es diferente. El reto es mejorar la calidad de solución a problemas por medio de la implementación de *software*.

La necesidad de enfoques sistemáticos para el desarrollo y mantenimiento de estos productos de *software* se patentizó en la década de 1960. Durante ésta, aparecieron las computadoras de la tercera generación y se desarrollaron técnicas de programación como la multiprogramación y el tiempo compartido. Estas nuevas capacidades aportaron la tecnología necesaria para el establecimiento de sistemas computacionales interactivos, multiusuario, en línea y en tiempo real.

Conforme las computadoras crecieron haciéndose más complejas, resultó obvio que la demanda por los productos de *software* creció en mayor cantidad que la capacidad de producir y mantener dichos productos. Desde 1968, se ha diversificado la utilización de las computadoras y se ha hecho más compleja y crítica para la sociedad moderna. Como resultado de esto, el campo de ingeniería de productos de *software* ha evolucionado para convertirse en una disciplina tecnológica de considerable importancia.

La ingeniería de *software* comparte, junto con las otras ramas de la ingeniería, el enfoque pragmático para el desarrollo y mantenimiento de artefactos tecnológicos; existen, sin embargo, diferencias significativas entre esta ingeniería y las otras, siendo la fuente principal de dichas diferencias la falta de leyes físicas para la programación, la intangibilidad del producto y lo oculto de las interfaces entre los diversos módulos de programación.

En el desarrollo de *software* es importante reconocer los problemas y sus causas. Las soluciones deben de proveer asistencia técnica por si mismas al desarrollador, mejorar la calidad del *software*, y finalmente, permitir al mundo del *software* llevar una buena relación con el mundo del *hardware*. Las metas primordiales de esta disciplina tecnológica son mejorar la calidad de estos productos y aumentar la productividad y satisfacción profesional de los ingenieros que se dedican a esta disciplina.

3.2 DEFINICIÓN

La ingeniería de software, o ingeniería de programación, se define como la disciplina tecnológica y administrativa dedicada a la producción sistemática y al mantenimiento de los productos de programación, los cuales son desarrollados y modificados a tiempo y dentro de un presupuesto definido. La ingeniería de software difiere de la programación tradicional en que se utilizan técnicas de ingeniería para especificar, diseñar, instrumentar, validar y mantener los productos dentro del tiempo y el presupuesto establecidos para el proyecto.

La calidad de los programas es una preocupación primordial de los ingenieros de programación, las características importantes de la calidad dependerán del producto en particular. En algunos casos, la transportabilidad de los productos entre diversas máquinas podrá ser un atributo de importancia capital, mientras que en otras ocasiones el uso eficiente de la memoria puede ser lo fundamental. Por otro lado, existen algunas características de calidad que son fundamentales en todo producto de programación, entre ellas: utilidad, claridad, confiabilidad y eficiencia.

El factor más importante de la calidad de un producto es su utilidad, es decir, que el producto de programación satisfaga las necesidades del usuario. En ocasiones no ocurre esto debido a la pobre comunicación existente entre el cliente, los usuarios y los ingenieros de programación. La planeación cuidadosa, el análisis y la participación del cliente son indispensables para el desarrollo de productos útiles.

Los productos de programación deben estar escritos con claridad y ser fáciles de entender. La clave para realizar un sistema fácil de probar y mantener radica en hacerlo comprensible; los productos de programación que se presentarán a los usuarios deben tener una integridad conceptual y ser claros en el propósito del proyecto.

La confiabilidad del producto está definida como la capacidad de un programa para desempeñar una función requerida bajo ciertas condiciones durante un tiempo específico. El grado de confiabilidad deseado en un producto particular puede ser expresado en términos del costo de la falla del producto.

Finalmente, un producto de software deberá ser eficiente tanto como la aplicación particular lo amerite. Existen productos que están limitados críticamente por el tamaño de memoria y la velocidad de ejecución.

3.3 SISTEMA DE INFORMACIÓN

Una de las áreas de aplicación para las computadoras que más ha crecido tiene que ver con el almacenamiento y obtención de datos. A este tipo de sistemas se les llama "sistemas basados en información", o simplemente, "sistemas de información", porque el principal objetivo del sistema es manejar información. En el corazón de dichos sistemas está una base de datos sobre la cual se aplican transacciones para crear, extraer, actualizar o eliminar objetos.

3.4 CICLO DE VIDA DE LOS SISTEMAS

El ciclo de vida de un producto de *software* se divide en una serie de actividades sucesivas; cada fase requiere información de entrada, procesos y resultados, todos ellos bien definidos. Se necesitan recursos para terminar los procesos de cada fase, y cada una de ellas se efectúa mediante la aplicación de métodos explícitos, herramientas y técnicas.

La ingeniería de programación consiste en la aplicación de metodologías de análisis y diseño para la creación y mantenimiento de sistemas de información. Existen cuatro principales fases en el ciclo de vida de todo sistema:

- 1) Análisis
- 2) Diseño
- 3) Desarrollo
- 4) Mantenimiento

3.4.1 Análisis

El análisis de sistemas es el estudio detallado de la forma como funciona el proceso operacional de la información en una organización, el cual se está considerando para implementarse en una computadora. La falta de análisis es la causa principal de retrasos en programación, incremento de costos, poca calidad, y altos costos de mantenimiento en los desarrollos de sistemas. Para evitar estos problemas se requiere de un análisis cuidadoso, tanto en el proceso de desarrollo, como en la operación del producto. Los principales propósitos de esta fase son aclarar los objetivos, necesidades y restricciones del sistema.

El analista de sistemas funciona como una interfaz entre las personas responsables de realizar dicho proceso (los usuarios) y las responsables de implementarlo en una computadora (los programadores). Este proceso es fundamental para el éxito del sistema final. El analista se entrevista con los usuarios y estudia la documentación proporcionada por éstos últimos. La metodología y sus formas para registrar la

información deben tener el suficiente detalle para que los programadores puedan trabajar con ella.

Existen tres principales etapas que se deben realizar durante el análisis:

- Estudio preliminar

Marca el inicio sobre el conocimiento general del problema y su objetivo es obtener toda la información necesaria para estructurar y plantear el proyecto. La definición del problema requiere de un entendimiento cabal del dominio del problema y del entorno de éste. A continuación se plantean cinco puntos que pueden resultar útiles para definir el problema:

- 1) Establecer específicamente el problema por resolver.
- 2) Justificar una estrategia de solución computarizada para el problema.
- 3) Identificar las funciones por realizar, las restricciones, el subsistema de equipo electrónico, el subsistema del producto de programación, y el del personal.
- 4) Determinar los objetivos y requisitos en el nivel del sistema para el proceso de desarrollo y los productos finales.
- 5) Establecer criterios de alto nivel para la aceptación del sistema.

También es importante considerar algunos aspectos sobre la organización o empresa para completar el análisis, estos son:

- Área de desarrollo del sistema
- Sistemas actuales que utilizan.
- Plataformas de *hardware* y *software*.
- Recursos humanos.
- Recursos económicos.
- Tiempo estimado para cubrir los requerimientos.

- Planeación

La planeación del proceso de desarrollo de un sistema de información comprende varias consideraciones importantes. La primera es definir un modelo para el ciclo de vida del producto. Este ciclo incluye todas las actividades requeridas para definirlo, desarrollarlo, probarlo, entregarlo, operarlo y mantenerlo. También es esencial definir un modelo de ciclo de vida para cada proyecto de programación, puesto que permite clasificar y controlar las diferentes actividades necesarias para el desarrollo y mantenimiento del producto. Un modelo del ciclo de vida entendido y aceptado por las partes interesadas en el proyecto mejora la comunicación, permitiendo así una mejor administración, asignación de recursos, control de costos y calidad del producto.

Otros aspectos:

- Estudios de costos y recursos – se estima el costo del proyecto independientemente de los recursos de *hardware* y *software* que se requieran y del personal necesario para la realización del proyecto.
 - Tiempo para desarrollar el sistema.
 - Estructura organizacional del proyecto.
 - Herramientas y técnicas que se emplearán.
 - Condiciones comerciales de operación – se refiere a la forma de pago, descuentos, garantía, soporte técnico, tiempo de entrega, coordinación del proyecto, etc.
 - Alcances del sistema.
- **Análisis estructurado**

El análisis estructurado de sistemas es un enfoque dirigido a los datos y orientado a la ingeniería para el análisis de sistemas. Es estructurado en el sentido de que su naturaleza es descendente, está diseñado para integrarse con las técnicas de diseño estructurado de programas y está unificado para resolver el problema de análisis de sistemas. La técnica central es el Diagrama de Flujo de Datos (DFD).

El DFD es una representación gráfica que tiene la finalidad de mostrar el movimiento de los datos a través de un sistema. Un DFD es una estructura bidimensional compuesta por cuatro tipos de elementos básicos (ver figura 3.4.1): procesos (representadas por rectángulos redondeados), almacenes de datos (rectángulos de aristas rectas orientados en forma horizontal), entidades externas (cuadrados) y flechas de flujo de datos. Los procesos transforman datos; los almacenes de datos contienen datos; las flechas de flujo de datos representan el movimiento de dichos datos; y las entidades externas son las interfaces de datos con personas o sistemas fuera del sistema actual.

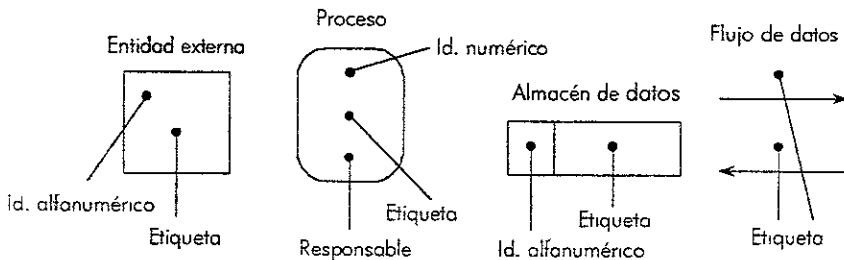


Fig. 3.4.1: Elementos de un DFD

Una característica del concepto de los DFD es que se trazan en forma descendente. La intención es que los procesos se exploten repetidamente, bajando de nivel, creando así una jerarquía de niveles de proceso, hasta que cada una de ellas de nivel más bajo represente una parte de la función que pueda implementarse en un módulo de código estructurado.

Es posible determinar exactamente los campos que pertenecen a un determinado almacén a partir de un estudio de los flujos de datos que entran y salen de él. Los campos deben pasarse en forma tal que estén agrupados en conjuntos simplificados (que contendrían datos no redundantes si se implementaran directamente como archivos físicos). Una técnica para lograr esto es la normalización de datos.

3.4.2 Diseño

El diseño está situado en el núcleo del proceso de ingeniería de programación por lo cual es la parte más importante. Una vez que los requerimientos han sido analizados y especificados, el diseño de software es la primera de tres actividades técnicas (diseño, programación y prueba) que son requeridas para construir y verificar software. Cada actividad transforma información en una forma que resulta en software válido.

Los requerimientos manifestados por información, modelos funcionales y de conocimiento, alimentan el proceso de diseño. Esta etapa produce un diseño de datos, un diseño de arquitectura y un diseño procedural. El diseño de datos transforma la información obtenida del análisis creando estructuras de datos que se requerirán para la implementación de software. El diseño de arquitectura define las relaciones entre los componentes estructurales del programa. El diseño procedural transforma los componentes estructurales en una descripción procedural del software.

Es aquí donde se toman decisiones que afectarán el éxito de la implementación del software. Las decisiones tomadas durante el diseño hacen de él un paso primordial para su desarrollo.

La importancia del diseño de software puede ser resumida en una sola palabra, calidad. El diseño es el único camino por donde es posible traducir los requerimientos del cliente en un producto de computadora. Sin el diseño, se pone en riesgo la construcción de un sistema estable en el cual no sea posible hacer modificaciones, donde las pruebas sean más difíciles y donde la calidad no puede ser construida a pesar de todo el proceso de ingeniería.

Los principales conceptos del diseño son:

a) Abstracción

La abstracción es la herramienta intelectual que permite trabajar con los conceptos independientemente de las instancias particulares de estos; durante la definición de

los requerimientos y el diseño, la abstracción permite la separación de los aspectos conceptuales de un sistema de lo que será más tarde instrumentado.

b) Modularidad

La arquitectura del *software* engloba modularidad, esto es, el *software* es dividido en componentes que son nombrados y desarrollados de forma separada, llamados módulos, que son integrados para satisfacer los requerimientos del problema.

c) Arquitectura de *software*

Se refiere a dos características importantes de un programa de computadora: 1) la estructura jerárquica de componentes procedurales (módulos), y 2) la estructura de datos.

d) Control jerárquico

También llamado estructura del programa, representa la organización de los componentes de un programa e implica un control de la jerarquía. El uso de una estructuración permite que un sistema sea definido en términos de unidades más pequeñas y manejables, con una clara definición de las relaciones entre las diferentes partes del sistema.

e) Estructura de datos

Es una representación de las relaciones lógicas entre elementos individuales de datos. La estructura de datos es tan importante como lo es la estructura del programa para la representación de la arquitectura de *software*.

f) Verificación

Es un concepto fundamental en el diseño de la programación, ya que el diseño es el puente entre los requerimientos del cliente y la implementación que satisface esos requerimientos. Un diseño es verificable si puede demostrarse que el diseño generará el producto que satisface los requerimientos del cliente.

Existen cuatro principales etapas durante el diseño: modelado de datos, normalización, diagrama estructurado y prototipo.

- Modelado de datos

Es la representación gráfica de las relaciones que guardan las entidades de datos del sistema, indicando cuál es la jerarquía y dependencia entre ellas, así como los esquemas de normalización. En el capítulo de Bases de Datos, subtema Modelo Entidad/Relación, se explica con más detalle dicho concepto.

- **Normalización**

Es un proceso metodológico que se aplica para eliminar redundancia en la información, así como para realizar la migración de llaves entre las entidades de datos. Una explicación más detallada se encuentra en el capítulo Bases de Datos, subtema Normalización.

- **Diagrama estructurado**

Es una representación gráfica de la “navegación” del sistema de información, a su vez representa los módulos que lo componen; es un diagrama jerárquico que muestra los niveles del sistema. También se le conoce como “mapa del programador” por la información que proporciona a la persona que programa.

El diseño estructural se emplea durante el diseño arquitectónico y comprende la identificación de los componentes de la programación, el desacoplamiento y la descomposición en módulos de procesamiento y estructuras de datos conceptuales, y la especificación de las interconexiones entre componentes.

- **Prototipo**

Es la simulación de la ejecución del sistema de información con la finalidad de que el usuario pueda comprobar que los requerimientos que planteó inicialmente están siendo considerados. El prototipo es empleado para validar la estructura y navegación del sistema.

3.4.3 Desarrollo

Consiste en un conjunto de técnicas y métodos de programación para desarrollar el sistema. El proceso de codificación traduce los detalles de la representación del diseño en líneas de código de un lenguaje de programación. Las características del lenguaje de programación tienen un impacto en la calidad y la eficiencia de la codificación. Los lenguajes de programación modernos proporcionan muchas características para mejorar la calidad del código fuente, como elementos estructurados, tipos de datos predefinidos, verificación de tipos, mecanismos para manejo de excepciones, y módulos con compilación separada. Existe una gran variedad de lenguajes en el mercado; el mejor lenguaje y el más apropiado para desarrollar software es aquel que se acople mejor a los requerimientos.

El objetivo principal de la instrumentación es el escribir código fuente y la documentación interna de modo que la concordancia del código con sus especificaciones sea fácil de verificar, y que se faciliten la depuración, pruebas y modificaciones. La calidad del código fuente se mejora mediante técnicas de codificación estructurada, buen estilo de codificación, documentos adecuados de

apoyo, buenos comentarios internos, y por las características que proporcionan los lenguajes de programación modernos.

Durante las primeras fases de definición y desarrollo, se intenta construir software desde un concepto abstracto hasta una implementación tangible. Ahora viene el proceso donde se realizan una serie de pruebas que tratan de demostrar el software que se ha construido.

A continuación se muestran una serie de reglas que sirven como objetivos de una prueba:

- 1) Probar el proceso de ejecución del programa con el objeto de encontrar algún error.
- 2) Una buena prueba es aquella que tiene una alta probabilidad de encontrar un error que aún no se ha descubierto.
- 3) Una prueba exitosa es aquella que descubre un error que está oculto.

- Verificación y validación

Los objetivos de estas dos actividades son valorar y mejorar la calidad de los productos del trabajo generados durante el desarrollo y modificación del software. Los atributos de la calidad deberán ser la corrección, perfección, consistencia, confiabilidad, utilidad, eficacia y apego a los estándares.

La verificación y validación implican la valoración de los productos de trabajo para determinar el apego a las especificaciones. Éstas incluyen las especificaciones de requisitos, documentación del diseño, principios generales de estilo, estándares del lenguaje de programación, estándares del proyecto, y estándares organizacionales y expectativas del usuario. Se deben examinar los requisitos para asegurarse que concuerden con las necesidades del usuario, así como con las restricciones del ambiente y los estándares de notación.

La alta calidad no se puede lograr solo mediante la prueba del código fuente. En el supuesto de que los errores del código fuente fueran la única medida de la calidad, las pruebas, por sí solas, no podrían garantizar la ausencia de errores de un programa. La calidad de los productos de trabajo generados durante el análisis y diseño se puede estimar y mejorar utilizando procedimientos sistemáticos de control de calidad, mediante recorridos e inspecciones y por medio de verificaciones automatizadas para supervisar que sea consistente y que esté completo.

3.4.5 Mantenimiento

El término "mantenimiento de software" se usa para describir las actividades de la ingeniería de software que ocurren después de entregar un producto al cliente. Esta fase representa el periodo en el que un producto desempeña un trabajo útil. Por lo general, el ciclo de desarrollo para un producto abarca entre 1 ó 2 años, mientras que la fase de mantenimiento dura de 5 a 10 años.

Las actividades de mantenimiento implican mejorar los productos de software, adaptarlos a nuevos ambientes, y corregir problemas. La mejora de los productos de software puede dar como resultado proporcionar nuevas capacidades funcionales, mejorar los despliegues al usuario y los modos de interacción, o revalorar las características del desempeño del sistema.

Se debe observar que el mantenimiento del software es un microcosmos del ciclo de desarrollo del software. El mejoramiento y la adaptación del software reinician el desarrollo en la fase de análisis, mientras que la corrección de un problema de software puede reiniciar el ciclo de desarrollo en la fase de análisis, en la fase de diseño, o en la de implementación.

Las actividades de análisis durante el mantenimiento del software implican la comprensión del alcance y efecto de una modificación deseada, además de las restricciones para hacer la modificación. El diseño durante el mantenimiento supone rediseñar el producto para incorporar los cambios deseados. Entonces estos deben implantarse, la documentación interna del código se debe actualizar, y se deben proyectar nuevos casos de prueba para evaluar la adecuación de la modificación.

Todas estas tareas se deben efectuar mediante un enfoque sistemático ordenado que rastree y analice los requisitos de las modificaciones, y con un cuidadoso rediseño, reimplantación, revalidación y redocumentación de los cambios. De otro modo, el producto de software se degradará con rapidez como resultado del proceso de mantenimiento.

4. BASES DE DATOS

4.1 INTRODUCCIÓN

El procesamiento de la información es esencial para la administración de los gobiernos, los negocios y la educación. En nuestra sociedad es vital para una organización o empresa proporcionar información correcta y puntual para apoyar la toma de decisiones y otras actividades gerenciales. Como resultado del crecimiento económico y avances tecnológicos, muchas organizaciones han crecido tanto en tamaño como en sofisticación de sus funciones administrativas. Mientras el volumen de procesamiento de datos crece a una rapidez sin precedentes, también crece la demanda de medios eficientes para manejarlos.

La invención de las computadoras revolucionó los métodos tradicionales del procesamiento de la información. En los sistemas de información convencionales, las aplicaciones individuales se desarrollaban independientemente, y cada programa de aplicación procesaba sus propios archivos privados. Como resultado, algunas actividades se duplicaban y la información redundante se almacenaba para usarla en distintas operaciones.

Al final de los años 60's surgió el sistema de bases de datos para superar los problemas asociados con los sistemas de información tradicionales. Archivos individuales se integraban en una sola base de datos para ser compartidos por todos los usuarios de una empresa. Dada la centralización de los datos por medio de un sistema de base de datos, los requerimientos de todos los usuarios se pueden coordinar de una manera efectiva para alcanzar la mejor utilidad general para la organización.

4.2 SISTEMA DE BASES DE DATOS

Un sistema de bases de datos es un sistema computarizado de información para el manejo de datos por medio de paquetes de *software* llamados Sistemas de Manejo de Bases de Datos (DBMS). Un sistema de bases de datos incluye cuatro componentes principales: datos, *hardware*, *software* y usuarios.

El DBMS es la porción más importante del *software* de un sistema de base de datos. Es una colección de numerosas rutinas de *software* interrelacionadas, cada una de las cuales es responsable de alguna tarea específica. Sus funciones principales son:

- Crear y organizar la base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos, de tal manera que los datos se puedan acceder rápidamente en cualquier parte de la base.

- Manejar los datos de acuerdo con las peticiones de los usuarios.
- Mantener la integridad y seguridad de los datos.
- Registrar el uso de las bases de datos.

El DBMS interpreta y procesa las peticiones del usuario para recobrar información de la base, es decir, sirve de interfaz entre los usuarios y la base de datos. Las “preguntas” a la base pueden tener distintas formas, pueden teclearse directamente desde la terminal, o codificarse con programas en lenguajes de alto nivel.

Un sistema de base de datos también proporciona a sus beneficiarios un control centralizado de sus datos de operación, los cuales representan uno de sus activos más valiosos. Las ventajas de tener un control centralizado de los datos son:

- Reducción de redundancia de los datos.
- Puede evitarse la inconsistencia, proporcionando así información correcta y fidedigna.
- Los datos pueden compartirse con otras aplicaciones existentes.
- Pueden hacerse cumplir las normas establecidas aplicadas a la representación de los datos.
- Pueden aplicarse restricciones de seguridad sobre los datos de operación.
- Puede conservarse la integridad, es decir, que los datos de la base sean exactos.

4.3 DEFINICIÓN

Una base de datos es una colección de archivos interrelacionados creados con un DBMS. El contenido de una base se obtiene combinando datos de todas las diferentes fuentes en una organización, de tal manera que los datos estén disponibles para todos los usuarios, y los datos redundantes puedan eliminarse, o al menos minimizarse.

4.4 ARQUITECTURA

Un sistema robusto de bases de datos puede proporcionar beneficios que van más allá de las simples facilidades convenientes para el usuario para realizar consultas. El DBMS tiene como objetivos lograr la independencia, economía, integridad, seguridad y correlación de los datos. Para lograr esto se establece una arquitectura, la cual se divide en tres niveles generales. En la figura 4.4 se muestra la arquitectura.

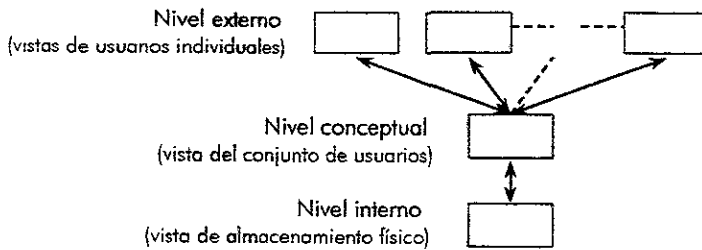


Fig. 4.4: Arquitectura de una base de datos

4.4.1 Nivel interno

Es una representación de muy bajo nivel de la base de datos en su totalidad. Es el nivel más cercano al almacenamiento físico, es decir, el que concierne a la manera como los datos se almacenan en realidad.

La vista interna se describe por el DBMS como un esquema interno, el cual no sólo define los diversos tipos de registros almacenados, sino que también especifica cuáles índices existen, de qué manera se representan los campos almacenados, en qué secuencia física se hallan los registros almacenados, etc.

4.4.2 Nivel conceptual

Es una representación del contenido total de información de la base de datos, en forma relativamente abstracta (orientado al usuario y no a la máquina) en comparación con la forma en la cual los datos se almacenan físicamente. Se pretende que sea una vista de los datos como son en realidad, y no como los usuarios están obligados a verlos por ciertas restricciones, por ejemplo, lenguaje particular o *hardware*.

El administrador de la base de datos (DBA) define este nivel mediante un esquema que comprende las siguientes definiciones:

- a) Definición de los datos – se describen el tipo de datos y la longitud de campo de todos los elementos.
- b) Relaciones entre datos – se definen relaciones entre datos para enlazar tipos de registros relacionados.

4.4.3 Nivel externo

Es el que se relaciona con el usuario, es decir, el que atañe a la manera como cada usuario ve los datos. Los usuarios son programadores de aplicaciones o usuarios de terminales en línea. Cada usuario tiene un lenguaje a su disposición. Para el programador de aplicaciones, se trata de un lenguaje convencional de programación.

Para el usuario de una terminal se trata de un lenguaje de consulta o un lenguaje de propósito especial hecho a la medida de sus necesidades.

El lenguaje de usuario incluye un sublenguaje de Datos (DSL), o sea, un subconjunto del lenguaje total que concierne a los objetos y a las operaciones de la base de datos. El DSL es una combinación de dos lenguajes: un Lenguaje de Definición de Datos (DDL), que permite la definición o descripción de los objetos de la base de datos, y un Lenguaje de Manipulación de Datos (DML), que apoya el manejo o procesamiento de esos objetos.

4.5 ESTRUCTURAS

Las estructuras tienen la finalidad de presentar gráficamente al usuario la base de datos, esto es, la vista externa. El rango de estructuras de datos soportadas a nivel usuario (externo o conceptual) afecta de manera decisiva a muchos componentes del sistema, por esta razón las bases de datos se clasifican de acuerdo con el enfoque que adoptan. Los tres enfoques difieren en la manera en que permiten al usuario ver y manipular las asociaciones. A continuación se presentan:

4.5.1 Enfoque relacional

En el sistema DBMS relacional, una estructura lógica se representa por medio de tablas bidimensionales llamadas relaciones. Todas las entidades están representadas como tablas separadas sin estar colocadas en alguna jerarquía fija. Para explicarlo se utilizará el siguiente ejemplo:

S

S#	NOMS	ESTADO	CIUDAD
S1	Salazar	20	Londres
S2	Jaramillo	10	París
S3	Bernal	30	París

P

P#	NOMP	COLOR	PESO	CIUDAD
P1	Tuerca	Rojo	12	Londres
P2	Perno	Verde	17	París
P3	Tornillo	Azul	17	Roma
P4	Tornillo	Rojo	14	Londres

SP

S#	P#	CTD
S1	P1	300
S1	P2	200
S1	P3	400
S2	P1	300
S2	P2	400
S3	P2	200

Los datos se organizan en *tablas* o *entidades*, en donde cada una de ellas se asemeja mucho a un archivo secuencial convencional. Cada tabla representa un objeto particular y específico, por ejemplo, la tabla S se refiere solamente a proveedores, la tabla P a partes y la tabla SP a las remesas; cada una de ellas tiene atributos (nombre, estado, color, etc.) que caracterizan a cada tabla. Los renglones de las tablas reciben el nombre de *registros* o *tuplas*. Las columnas se llaman *campos* o *atributos*.

Un campo es la unidad más pequeña a la cual uno puede referirse en un programa de computadora.

Un registro es un conjunto de campos con relación entre sí.

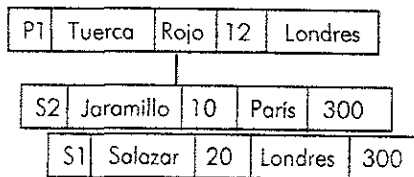
Un archivo es una colección de registros del mismo tipo.

Cada columna representa un campo, y cada renglón un registro consistente en cierto número de atributos que describen una tabla.

La ventaja principal del enfoque relacional está en la simplicidad de su representación en la estructura lógica de la base de datos y en la flexibilidad para establecer relaciones de datos por medio de campos de conexión.

4.5.2 Enfoque jerárquico

En esta vista los datos se representan por una sencilla estructura de árbol. Cada árbol se compone de una ocurrencia de registro de cierta información, con un conjunto de ocurrencias de registro de otra información. Haciendo referencia al ejemplo anterior, se observaría lo siguiente:



En general, la raíz del árbol puede tener cualquier número de dependientes; cada uno de estos puede tener cualquier número de dependientes de nivel inferior, y así sucesivamente hasta cualquier número de niveles.

Sin embargo, esta estructura es un objeto más complejo que las tablas del enfoque anterior porque contiene varios tipos de registros, no sólo uno (en el ejemplo, hay uno para las partes y otro para los proveedores) y también porque contiene ligas que conectan ocurrencias de registros. De igual forma, resulta más complicado el empleo del DML, por lo que el usuario está obligado a dedicar tiempo y esfuerzo a resolver problemas introducidos por la estructura jerárquica. Resulta claro que la situación empeorará a medida que se introduzcan más tipos de registros en la estructura y la jerarquía se vuelva más compleja. Esto significa que los programas son más complicados de lo necesario, con la consecuencia de que su escritura, depuración y mantenimiento necesitarán más tiempo de programador del debido.

4.5.3 Enfoque de red

Esta estructura representa los datos, al igual que el enfoque jerárquico, con registros y ligas. Una red es una estructura más general que una jerarquía porque una ocurrencia de registro específica puede tener cualquier número de superiores inmediatos, así como cualquier número de dependientes inmediatos. Por estas razones, la estructura interna de este archivo es más compleja que en el caso jerárquico. El manejo del DML requiere de más operadores para realizar consultas.

La desventaja principal del enfoque de red es su excesiva complejidad, tanto en la estructura de datos en sí, como en el DML asociado.

4.6 MODELO ENTIDAD/RELACIÓN

Basado en el enfoque relacional, el modelo entidad/relación es una representación gráfica de la relación que guardan las entidades de datos entre sí, indicando cual es la jerarquía y dependencia entre ellas, así como los esquemas de normalización (selección de llaves, migración de llaves, etc.).

La relación es una conexión entre dos entidades que representa la cardinalidad, la cual puede ser: 1:1 (uno a uno), 1:M (uno a muchos) o M:N (muchos a muchos).

4.6.1 Relación uno a muchos

Se dice que una relación entre entidades es 1:M si la ocurrencia de una entidad está relacionada con ocurrencias múltiples de la otra entidad (ver figura 4.6.1). La mayoría de las relaciones que se presentan en un esquema son de este tipo. Por ejemplo,

suponiendo una familia de cinco elementos, un padre tiene muchos hijos y cada hijo tiene un padre.

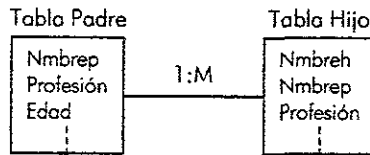


Fig. 4.6.1: Relación uno a muchos (1:M)

Para explicar cómo es que se realiza dicha relación, primera se definirá lo que es una *llave primaria*. Dentro de una entidad específica puede existir un atributo cuyos valores son únicos dentro de dicha tabla y, por tanto, se pueden usar para identificar los registros de esa tabla. Para este caso, se dice que dicho campo o atributo es la *llave primaria* de la tabla (Nmbrep y Nmbreh para las tablas Padre e Hijo respectivamente).

No toda entidad tendrá una llave primaria de un solo campo, sin embargo, cada entidad tendrá alguna combinación de atributos que, tomados en conjunto, tienen la propiedad de identificación única. A ese conjunto de atributos también se le llama *llave*.

La relación uno a muchos puede implantarse físicamente por medio de señadores, índices o combinación de ambos. Si se enlazan dos archivos con índices, el campo común debe estar contenido en ambos tipos de registro Padre e Hijo. El campo común en el registro padre debe ser una llave primaria mientras que en el campo hijo es una *llave foránea* (Nmbrep). Así, la relación entre la entidad padre e hijo es 1:M.

4.6.2 Relación uno a uno

La relación 1:1 es un caso particular de la relación 1:M. Con una relación 1:1, la ocurrencia de una entidad se puede enlazar a sólo una ocurrencia de otra. Este tipo de relación es inusual.

4.6.3 Relación muchos a muchos

La relación M:N sucede cuando se puede asociar una ocurrencia en una entidad con muchas ocurrencias en la otra entidad, o viceversa. Por ejemplo, la relación entre profesor y estudiante es M:N porque un profesor enseña a muchos estudiantes y un estudiante puede tener distintos profesores. Esta relación no se puede implantar directamente porque no se puede determinar cuáles son las ocurrencias específicas que suceden entre las tablas. Para lograr esto, se reducen a relaciones uno a muchos insertando una tabla de tipo conexión que contiene, al menos, las llaves primarias (Clave y Cuenta) de las tablas originalmente involucradas. Ver la figura 4.6.3.

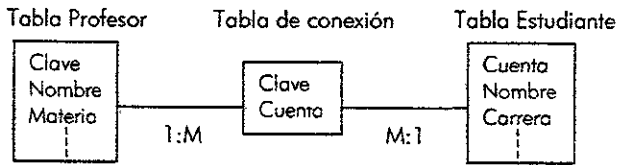


Fig. 4.6.3: Relación muchos a muchos (M:N)

4.7 NORMALIZACIÓN

Para el diseñador de bases de datos, el derivar entidades o registros de tipo conceptual de un grupo de datos se puede hacer intuitivamente. Sin embargo, la intuición no siempre ayuda a resolver el problema, especialmente cuando el diseño es muy complejo. La teoría de la normalización es una ayuda que proporciona un procedimiento riguroso para establecer el diseño de una base de datos.

La normalización es una metodología que sirve para establecer adecuadamente la estructura lógica de los datos y para decidir las tablas que se necesitan y los atributos que deben tener cada una de ellas. La meta final del proceso es la agrupación de todos los atributos de una base de datos en relaciones apropiadas para que la base se pueda almacenar con el mínimo de datos redundantes. El propósito de este proceso es quitar las cualidades indeseables de una relación que puedan causar anomalías en el almacenamiento cuando se efectúen operaciones de inserción, eliminación o actualización de la base de datos.

Las relaciones normalizadas se agrupan en varias categorías llamadas *formas normales*, siendo cada nivel una descomposición más completa de una relación que la del nivel anterior. Existen varias de ellas: 1FN (primera forma normal), 2FN (segunda forma normal), 3FN (tercera forma normal), FNBC (forma normal de Boyce-Codd), 4FN (cuarta forma normal) y 5FN (quinta forma normal). La aplicación de esta última forma es muy inusual.

Antes de explicar las formas normales se definirá los siguientes conceptos:

- 1) *Dependencia funcional* – la dependencia funcional entre los atributos A y B en una relación se define como sigue: el atributo A es funcionalmente dependiente del atributo B si el valor de A está determinado por el valor de B.
- 2) *Determinante* – para el anterior caso, se dice que B es el determinante.
- 3) *Llave aspirante* – es un atributo o grupo de atributos cuyo contenido puede representar de manera única a cada registro de una relación.
- 4) *Dependencia multivalores (DMV)* – se dice que el atributo A de una relación es dependiente de multivalores del atributo B si un rango específico de valores del atributo A está determinado por un valor particular de B.

5) *Archivo plano* – es una relación donde sólo un valor está contenido en cada campo.

4.7.1 Primera forma normal

Los datos en la primera forma normal tienen la propiedad de que cada anotación de datos, o valor de campo, debe ser indivisible (atómico). De otra forma, todos los campos en cada registro contienen un solo valor tomado de sus dominios respectivos. El dominio de un campo es el conjunto de valores permitidos para dicho campo.

La representación de los datos en la primera forma normal no es por sí misma útil como un arreglo para el control de la redundancia, sino sólo un punto de arranque para continuar trabajando.

4.7.2 Segunda forma normal

Una relación pertenece a 2FN si está en 1FN y cada atributo no llave de la relación es total y funcionalmente dependiente de su llave primaria.

La metodología en este punto se orienta al problema de qué buscar en la estructura de datos y qué modificar para disminuir la redundancia. Los datos se dividen en tablas, cada una de las cuales tiene la propiedad de que su llave entera se necesita para definir cada uno de sus campos no llave.

En este proceso varios campos se duplican, sin embargo es necesario hacerlo entre aquellos campos que participan como campos llave, generando un decremento neto en la redundancia con relación a la primera forma normal. Es importante hacer notar que cada una de las tablas que resultaron de la división de datos representan áreas de conocimiento identificables de manera individual (aunque no en el grado final).

4.7.3 Tercera forma normal

Una relación es 3FN si es 2FN y ningún atributo no llave en la relación es funcionalmente dependiente de algún otro atributo no llave.

En esta forma la situación es tal que un campo no llave define a otros del mismo tipo, indicación de que se están mezclando tipos fundamentalmente distintos de información en la misma tabla, provocando redundancia.

Se requiere de la creación de nuevas tablas y puede decirse que en cada tabla no existe situación alguna en la que un campo no llave defina a otro del mismo tipo. Se dice que una tabla R está en 3FN si y sólo si cada registro de R se compone de un valor de llave primaria que identifica alguna entidad, junto con un conjunto de valores de atributos mutuamente independientes que describen esa entidad de alguna manera.

4.7.4 Forma normal de Boyce/Codd

Se dice que una relación se encuentra en FNBC si cada determinante en la relación es una llave aspirante.

Es una revisión mejorada de la tercera forma normal. En la mayoría de los casos, cuando una relación es 3FN también es FNBC. La definición de esta forma normal es más restrictiva que la 3FN original. En otras palabras, cuando una relación es 3FN, no necesariamente es FNBC.

4.7.5 Cuarta forma normal

Una relación es 4FN si es FNBC y no contiene dependencias multivalores. En circunstancias normales no se requiere más normalización después de 3FN porque ya se han tomado en cuenta los atributos DMV durante la etapa inicial de la normalización para derivar un archivo plano 1FN.

Los datos normalizados, sujetos a posibles modificaciones estructurales por razones de desempeño, son el diseño final para las bases de datos relacionales. El proceso de normalización asegura que los campos de unión aparecerán en todas las tablas en que deben hacerlo con base en las relaciones entre los campos.

Varias razones hacen que el diseño cuidadoso de la base de datos sea esencial: la redundancia de datos, el desempeño en la aplicación, la independencia y seguridad de los datos, y la facilidad de programación.

- *Redundancia de datos* – el objetivo es evitar la repetición de datos tanto dentro de un archivo como entre archivos.
- *Desempeño* – se considera como la velocidad operacional de aplicaciones y sistemas, sin embargo puede verse afectado o influenciado por un cierto número de factores en el medio ambiente de la base de datos. Varios de estos factores pueden ser la velocidad de procesamiento de la Unidad Central de Proceso (CPU), la tasa de transferencia de datos a disco, la velocidad de canal, el sistema de *hardware*, etc.
- *Independencia de los datos* – es la capacidad para modificar su estructura sin afectar programas ya existentes.
- *Seguridad de los datos* – se refiere a las claves de acceso asociadas con un usuario específico o con datos particulares.
- *Facilidad de programación* – el objetivo es disminuir la complejidad de programación y las tasas de error esperadas de dicha programación.

5. DESARROLLO DEL SISTEMA "ETNOBIB"

5.1 ANÁLISIS DEL PROBLEMA

5.1.1 Planteamiento del problema

Los libros, revistas, artículos, folletos, etc., son una de las principales fuentes de información en nuestros días. El contar con información actualizada de forma rápida y sencilla es sumamente importante para cualquier persona o institución. La computadora es una herramienta que ha venido a ayudar y agilizar muchas de las tareas que anteriormente tenían que ser realizadas manualmente.

Actualmente la computadora ha venido a ayudar a la tarea de la búsqueda bibliográfica en bibliotecas y en organizaciones que así lo requieren. La antropología, en especial la etnografía en México, es una de las áreas que no ha recibido esta ayuda, por lo que la necesidad de tener un *software* de información bibliográfica de las regiones indígenas ha sido el motivo de elaboración de esta tesis.

Para ello se propone el diseño y desarrollo del "Sistema inteligente para la gestión de información bibliográfica sobre etnografía en México" (ETNOBIB). Este sistema de información debe ser una herramienta que sirva para recopilar, clasificar, documentar y mantener la mayor cantidad de bibliografía sobre etnografía en México. Otra elemental función del sistema es que tenga la capacidad de realizar búsquedas de fichas bibliográficas por diferentes tipos y cantidad de criterios.

Es importante mencionar que el sistema ETNOBIB tiene como finalidad el ser distribuido en todo el país a todos los profesionales interesados en la etnografía mexicana, contando con una herramienta de trabajo actualizada y completa para fines de investigación y docencia.

En nuestro país actualmente se tiene un importante porcentaje de usuarios de computadoras, sin embargo los usuarios caseros son aquellos que resultan perjudicados por las actualizaciones constantes de *hardware* y *software*, quedando a la zaga de nuevas y mejores aplicaciones. Una gran cantidad de los profesionales interesados en la etnografía no cuentan con el poder de cómputo necesario que se requiere para manejar grandes aplicaciones y/o sistemas de información, por lo que se requiere que el sistema no consuma grandes cantidades de recursos de cómputo, pero que al mismo tiempo sea eficiente y amigable con el usuario.

Para el almacenamiento de la información se necesita utilizar una base de datos. Se empleará el modelo relacional de bases de datos ya que es el más común en computadoras personales. Cuando la información se encuentra almacenada, no en una tabla, sino dispersa en tres o más, las consultas a la base de datos pueden resultar muy complejas.

El proceso de extraer datos ubicados en múltiples tablas, implica encontrar los datos requeridos en dos tablas con un campo común. Las tablas se unen utilizando el campo común para formar una nueva relación que contiene los datos deseados y el campo común de cada una de las tablas originales. Otras tablas, que contienen datos requeridos para completar la consulta original, se unen entre sí con la relación recientemente formada hasta que finalmente se obtiene una relación que contiene todos los datos deseados.

La búsqueda y unión de múltiples relaciones por este procedimiento consumen mucho tiempo y memoria, por lo que se requieren computadoras veloces con una buena cantidad de memoria RAM para tener una respuesta en un tiempo razonable.

La reducción del proceso es cada vez más importante en aquellos casos en que la información solicitada está dispersa en diferentes relaciones. Para el caso de desarrollo de este sistema se pretende tener una base de datos con aproximadamente 25,000 registros. Tomando en cuenta la cantidad de tablas (40) de la base de datos que se implementarán y la capacidad de cómputo de los usuarios finales, se determinó desarrollar un sistema para realizar las consultas bibliográficas, teniendo como principal fin administrar los recursos de cómputo y hacer más eficiente el tiempo de respuesta del sistema.

5.1.2 Análisis de necesidades

El Instituto Nacional de Antropología e Historia planteó la necesidad de que el sistema de información ETNOBIB fuera capaz de implementarse en computadoras cuyos recursos de *hardware* y *software* fueran limitados, es decir, que ETNOBIB pudiera ejecutarse, cuando menos, en máquinas con sistema operativo Windows 3.1, con memoria RAM de 8 Mb y un espacio libre en disco duro considerablemente pequeño.

Por las anteriores razones, se decidió utilizar como herramienta de desarrollo el lenguaje de programación Visual Basic de 16 bits, esto debido a que en el sistema operativo Windows 3.1 sólo es posible ejecutar aplicaciones de 16 bits.

Existen lenguajes de programación de inteligencia artificial que resultan muy eficientes para el manejo de la base de conocimiento y los métodos de inferencia, pero requieren de una buena cantidad de memoria y son restrictivos. La interfaz de un sistema experto con una base de datos puede ser mucho más fácil en un lenguaje convencional que en un lenguaje de inteligencia artificial. Visual Basic tiene como manejador nativo de base de datos a Access 2.0, por lo que resulta ser muy sencilla la interfaz y el manejo en la programación de las transacciones con la base de datos.

Access 2.0 es la herramienta que se utilizó para crear la base de datos en donde se almacenará la información requerida por los diferentes módulos del sistema. Además, dicha herramienta emplea el modelo de datos relacional, que concuerda con la estructura elegida para el desarrollo del proyecto.

Los requerimientos de *software* y *hardware* son:

- Computadora IBM PC o compatible
- Microprocesador 80386 o superior (Recomendado Pentium)
- 8MB de memoria RAM convencional (Recomendado 16MB)
- Unidad de disco compacto
- Monitor compatible Windows VGA o superior (se recomienda una resolución mayor o igual a 800x600 pixeles)
- Espacio libre en disco duro de 5MB
- Sistema operativo Windows 3.1 o superior (recomendado Windows 95)

5.1.3 Lenguaje de programación

Una vez que se ha establecido el *hardware* en el cuál se implementará el sistema, es necesario seleccionar la herramienta de desarrollo. El sistema necesita tener una interfaz amigable con el usuario y además que soporte los requerimientos de *hardware*, que para este caso resultan ser limitados, principalmente por la memoria RAM, por la velocidad del microprocesador y por el sistema operativo en donde se implementará.

Visual Basic maneja lo que se conoce como programación orientada a eventos, esto significa que el usuario puede disparar ciertos eventos (mover el ratón, hacer *click* sobre algún control, etc.) o el sistema operativo, en este caso Windows.

Las aplicaciones basadas en Windows son multitarea. Pueden compartir el espacio de la pantalla y el tiempo de procesador. También se basan en elementos gráficos, a diferencia de las aplicaciones basadas en MS-DOS, las cuales están basadas en caracteres.

Visual Basic tiene dos grandes componentes: la parte gráfica y la parte de código. Viendo esto de forma tabular tendría la siguiente forma:

Aplicación	Formas
	Propiedades
	Eventos
	Controles
	Propiedades
	Eventos

Visual Basic ofrece a los programadores objetos (formas y controles) que pueden usarse para construir aplicaciones. Una forma es la parte principal de una aplicación basada en gráficos. Es lo que el usuario ve y con lo cual interactúa para llevar a cabo alguna tarea.

Los controles son herramientas que se colocan en las formas para dar a una aplicación funcionalidad, por ejemplo: los botones de comando, las etiquetas, combos, etc. Los controles y formas tienen propiedades o atributos que se pueden cambiar en tiempo de diseño o en tiempo de ejecución. Los procedimientos de eventos son códigos internos que Visual Basic y Windows nos dan para darle funcionalidad a una aplicación.

5.2 CONCEPTUALIZACIÓN DEL SISTEMA

Objetivos del sistema:

- Diseñar una base de datos que a la larga concentre la totalidad del acervo bibliográfico sobre etnografía en México realizado durante el siglo XX.
- Diseñar, desarrollar e implementar un sistema dotado de un módulo de captura, un módulo de actualización, un módulo de generación de reportes, un módulo de consultas, un módulo de seguridad y un sistema experto.
- El sistema experto permitirá superar las limitaciones que frecuentemente enfrentan los sistemas de consulta del INAH en cuanto a los resultados que brindan.
- Producir una herramienta de trabajo para la gestión de información bibliográfica.
- Desarrollar una arquitectura híbrida integrando bases de datos con sistemas expertos.

El sistema se forma de tres componentes principales como se ilustra en la figura 5.2:

- Una base de datos que contenga toda la información bibliográfica. Debe estar estructurada de tal manera que el usuario obtenga con veracidad y en un breve lapso de tiempo la información deseada.
- Un sistema experto que realice las consultas de fichas bibliográficas personalizadas o según un criterio específico.
- Módulos para realizar la captura o edición de información, y la elaboración de reportes a fin de dar mantenimiento a la base de datos de manera fácil, rápida y eficiente.

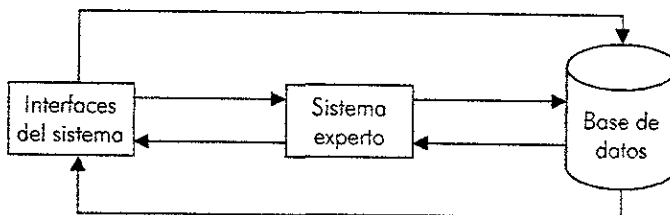


Fig. 5.2: Estructura general del sistema

Existen varios tipos de documentos que serán consultados: libros, artículos de libro, publicaciones periódicas y tesis. Cada ficha tendrá su correspondiente información bibliográfica, así como información etnográfica y temática. Para almacenar la información etnográfica y temática es necesario contar con catálogos que contengan estados, municipios, localidades, grupos etnolingüísticos, regiones y temas.

El sistema debe permitir agregar, modificar y eliminar fichas de los cuatro tipos de documento, pero no solo a estos últimos, sino también debe realizar las mismas funciones sobre los catálogos. Para el mantenimiento de la base de datos se contará con una opción de impresión de reportes de los diferentes tipos de documentos y de los catálogos, contando con diversas opciones de reporte.

El módulo de consultas tendrá una opción para realizar búsquedas con base en criterios bibliográficos comunes (por autor, título, año, etc.), limitando la búsqueda al tipo de documento deseado, es decir, libros, artículos de libro, publicaciones periódicos o tesis.

También es necesario contar con un módulo de consultas expertas, considerando criterios bibliográficos, etnográficos y temáticos. Lo anterior puede también limitarse al tipo de documento deseado, pero también se puede abrir el criterio para que el sistema busque en toda la base de datos, sin importar el tipo de documento. Este módulo será capaz de proporcionar (frecuentemente) resultados al usuario en el caso de que la consulta no haya sido exitosa debido a que no exista el documento, o bien, que la información tecleada por el usuario esté incorrecta. Es importante mencionar que el sistema experto entrará en función cuando el número de criterios a consultar sea mayor o igual a tres.

Las consultas se deberán poder imprimir con diferentes opciones: una versión corta que contenga solo información bibliográfica, una versión larga que tenga información bibliográfica, etnográfica y temática, y una serie de reportes que imprimen los comentarios, índice y bibliografía de cada ficha bibliográfica.

5.3 ADQUISICIÓN DEL CONOCIMIENTO

La fuente de donde se extrajo el conocimiento para la construcción del sistema experto fue de la experiencia y los conocimientos del experto humano. La información se obtuvo a través de entrevistas programadas, las cuales se llevaban a cabo en promedio una vez a la semana. En este caso el experto fue un etnólogo coordinador de la bibliografía. La experiencia del experto es en el campo de la antropología, arqueología y etnología, auxiliando a realizar búsquedas y clasificar los documentos.

La información proporcionada por el experto era del tipo bibliográfica, etnográfica y temática. Respecto a los datos etnográficos y geográficos, proporcionó información

sobre los municipios de los estados de la República Mexicana, así como las localidades de dichos municipios. De igual forma, presentó un listado de las regiones indígenas que tienen consideradas para emplear con el sistema ETNOBIB. También aportó el nombre de muchas de las familias lingüísticas que existen en todo el país, los grupos lingüísticos, las lenguas que hablan dichos grupos étnicos, y finalmente los sinónimos de las lenguas aceptados y reconocidos por un grupo de etnólogos.

Los temas se obtuvieron por medio de un listado que él mismo realizó. Para ello requería de la construcción de tres tablas en la base de datos, una para los conceptos generadores, otra para las sinonimias y otra para los conceptos derivados. Cada una de ellas manteniendo una relación entre sí. Como se mencionó anteriormente, los temas forman parte de los catálogos.

Además de lo anterior, se discutieron muchos aspectos referentes a la visualización del sistema, pero principalmente sobre el funcionamiento que tendrían las consultas bibliográficas - etnográficas. El interés del experto era que cada consulta realizada a la base de datos tratara de brindar, ya sea el resultado exacto de la consulta, o bien una aproximación fidedigna sobre la ficha que se consultó. De esta forma, el usuario puede obtener información completamente relacionada a su interés.

Cada regla del sistema experto es una consulta a la base de datos, donde la consulta es sencilla, es decir, involucra solo un criterio de búsqueda en una sola tabla. La consulta de un usuario estará conformada por varias consultas simples, según la cantidad de criterios que use el usuario. El resultado de cada consulta sencilla es procesado y complementado con otro resultado anterior, y así hasta acabar. Si el resultado no ha sido satisfactorio, es decir, no se ha encontrado una sola ficha que coincida con los criterios solicitados, entonces se inicia una nueva búsqueda en donde se busca que coincida la cantidad de $n-1$ criterios de n originales, permitiendo eliminar aquellos criterios que hayan causado un conflicto en la consulta y dando así una aproximación al resultado esperado. Este razonamiento se puede observar gráficamente en la figura 5.3.1; en dicha gráfica solo se esquematiza el caso en que se seleccionan tres criterios.

Otro aspecto interesante que remarcó el experto es que existe una gran diversidad de formas en que una lengua puede ser consultada. Por tal motivo, el sistema experto tiene la capacidad de buscar todos los sinónimos posibles de una lengua para proporcionar el resultado que el usuario espera. Los sinónimos se encuentran en la base de conocimientos y fueron proporcionados por el experto humano.

5.3.1 Árbol de decisión

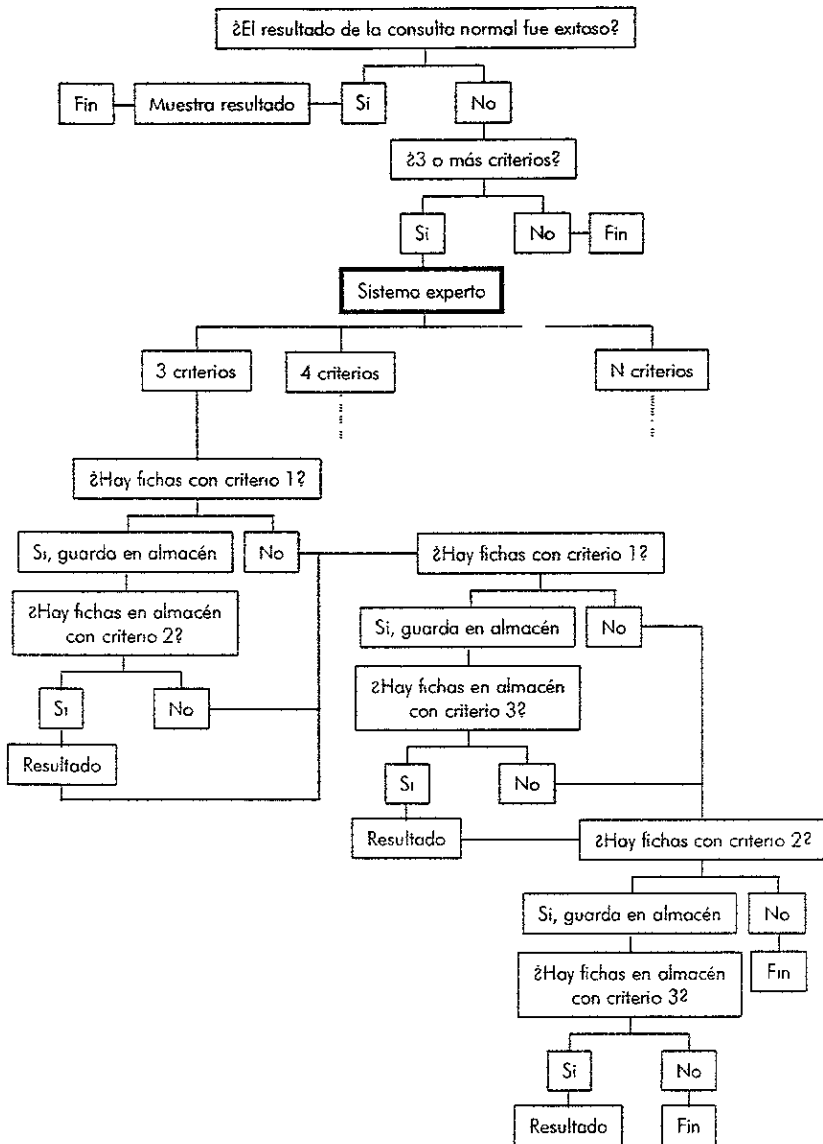


Fig. 5.3.1: Esquemización de la consulta experta

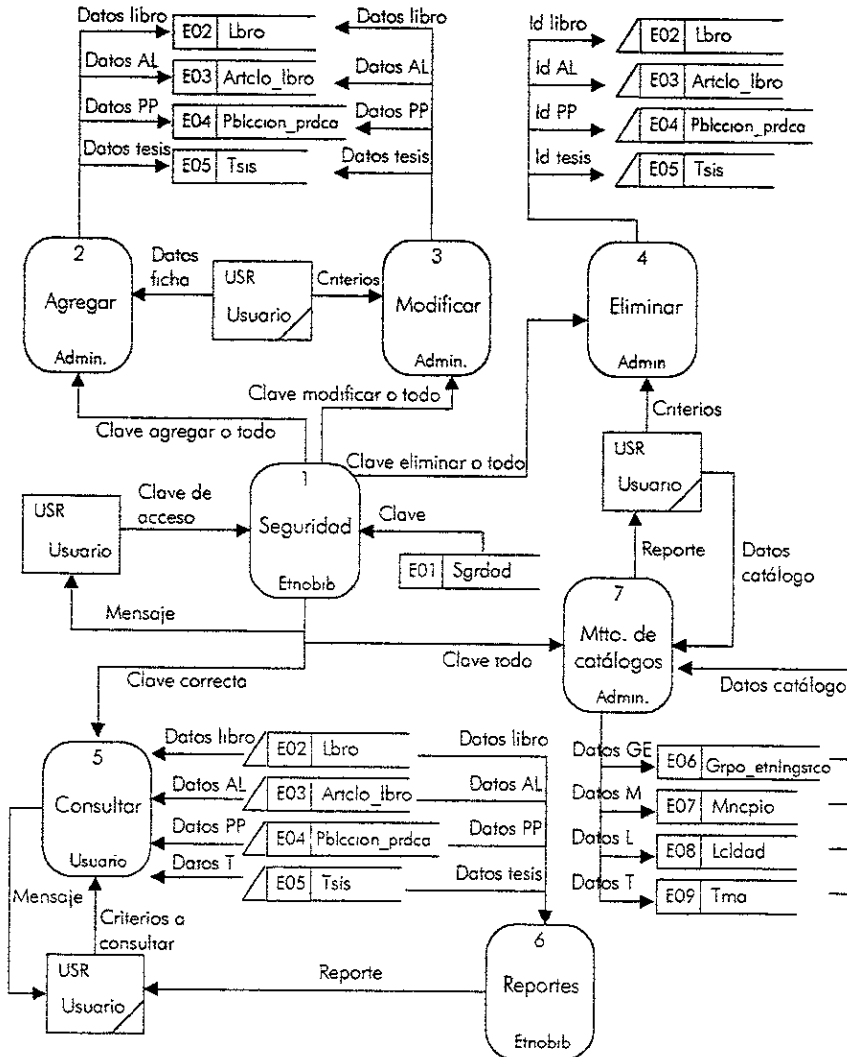
5.4 DIAGRAMA DE FLUJO DE DATOS

La preparación para el diseño de una base de datos en realidad comienza en una etapa muy anterior del ciclo de vida del procesamiento de datos. Esta etapa se explica en el capítulo de Ingeniería de software, en "Análisis estructurado".

En la figura 5.41 se muestra el DFD de primer nivel diseñado para el sistema. Este representa en forma general el funcionamiento de ETNOBIB. En las figuras 5.4.2, 5.4.3, 5.4.4, 5.4.5, 5.4.6, 5.4.7 y 5.4.8 se puede observar lo que ocurre internamente en cada uno de los procesos principales del sistema, así como los datos, entidades y/o almacenes que requiere cada uno de ellos.

Diagrama de Flujo de Datos

Fig. 5.4.1: Diagrama general - DFD de primer nivel



Explosión de procesos – DFD's de segundo nivel

Fig. 5.4.2: Proceso "Seguridad"

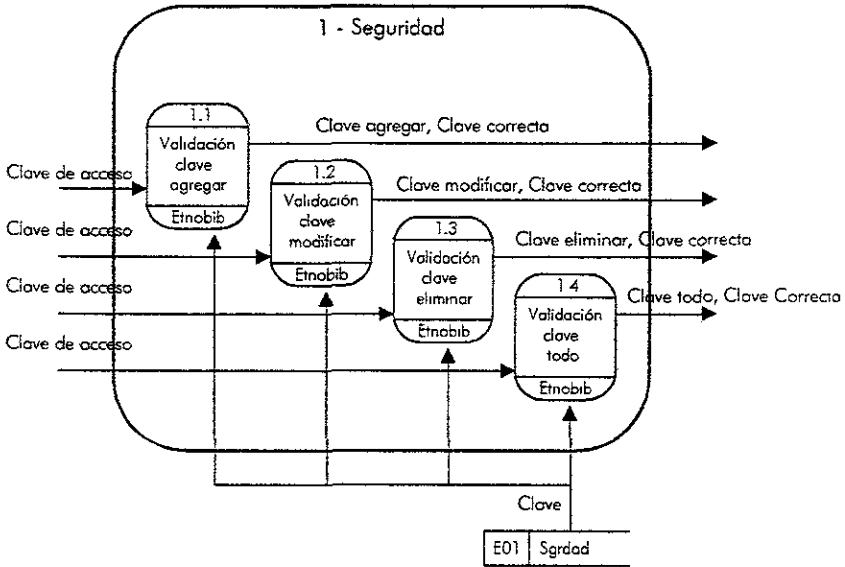


Fig. 5.4.3: Proceso "Agregar"

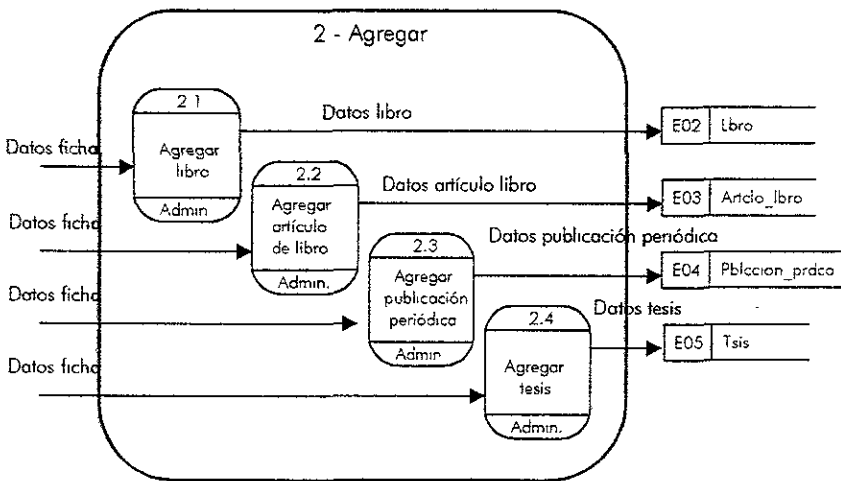


Fig. 5.4.4: Proceso "Modificar"

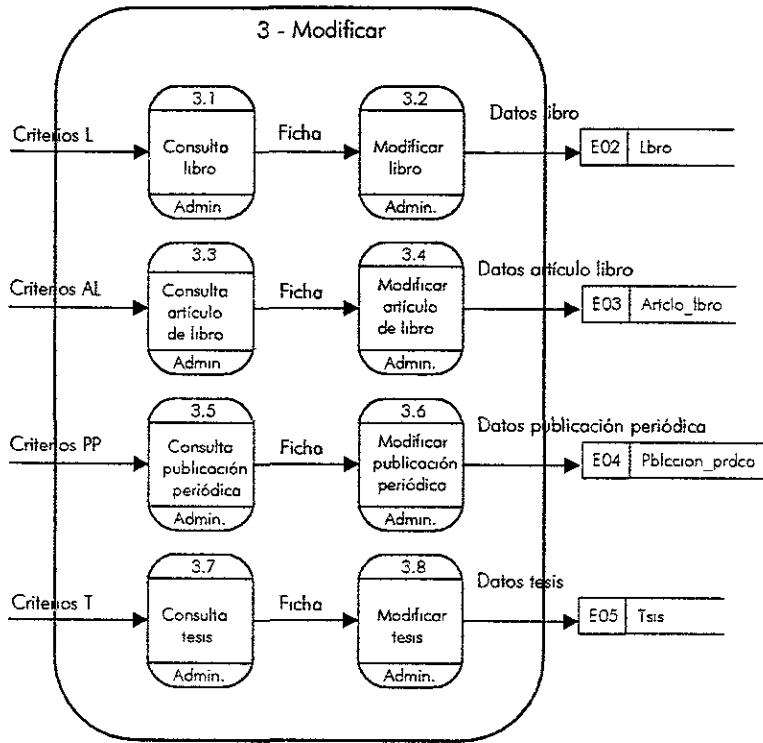


Fig. 5.4.5: Proceso "Eliminar"

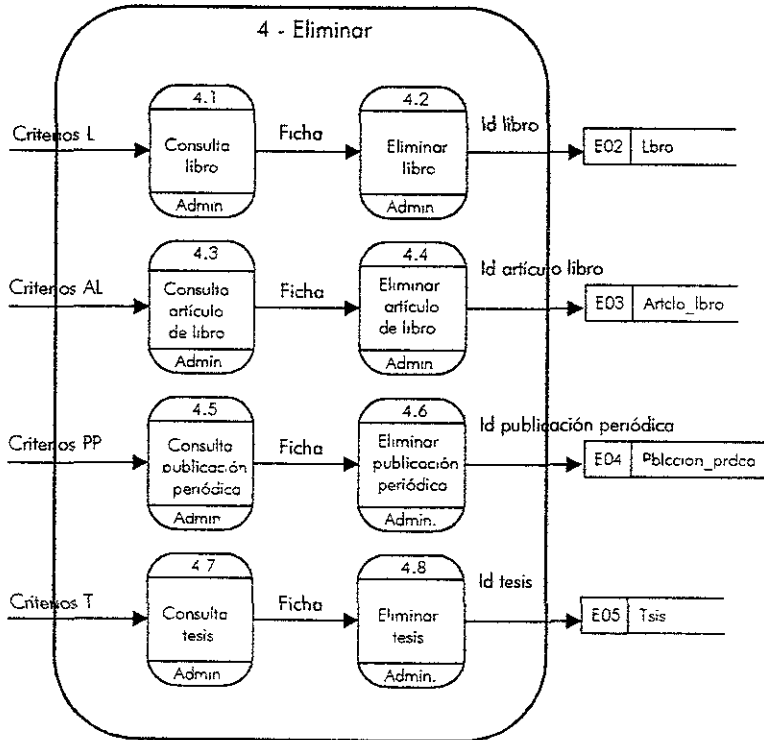


Fig. 5.4.6: Proceso "Consultar"

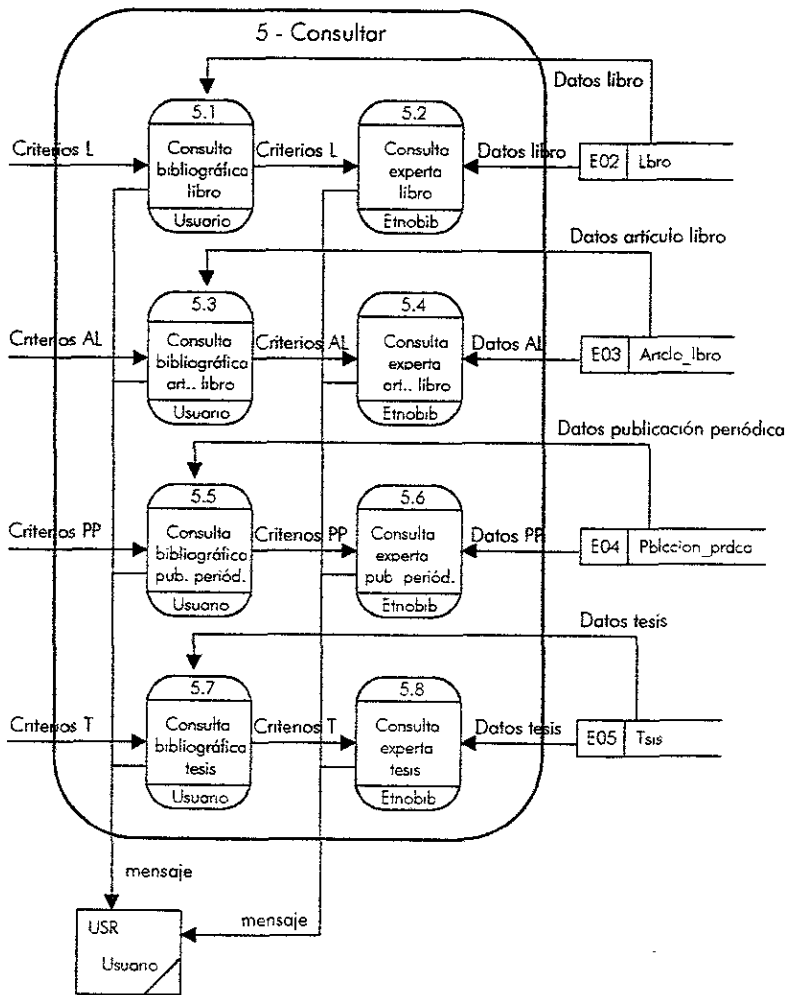


Fig. 5.4.7: Proceso "Reportes"

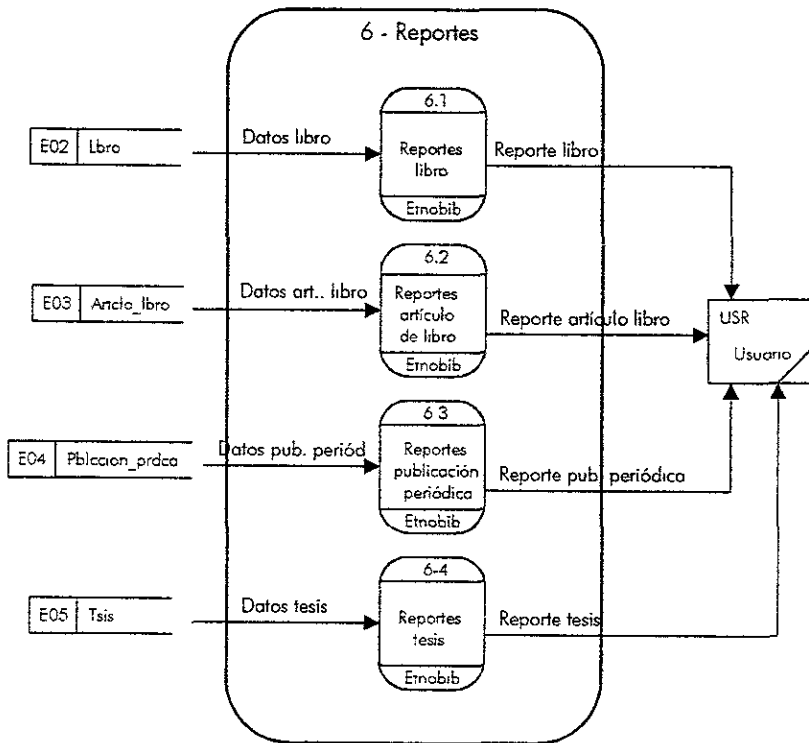
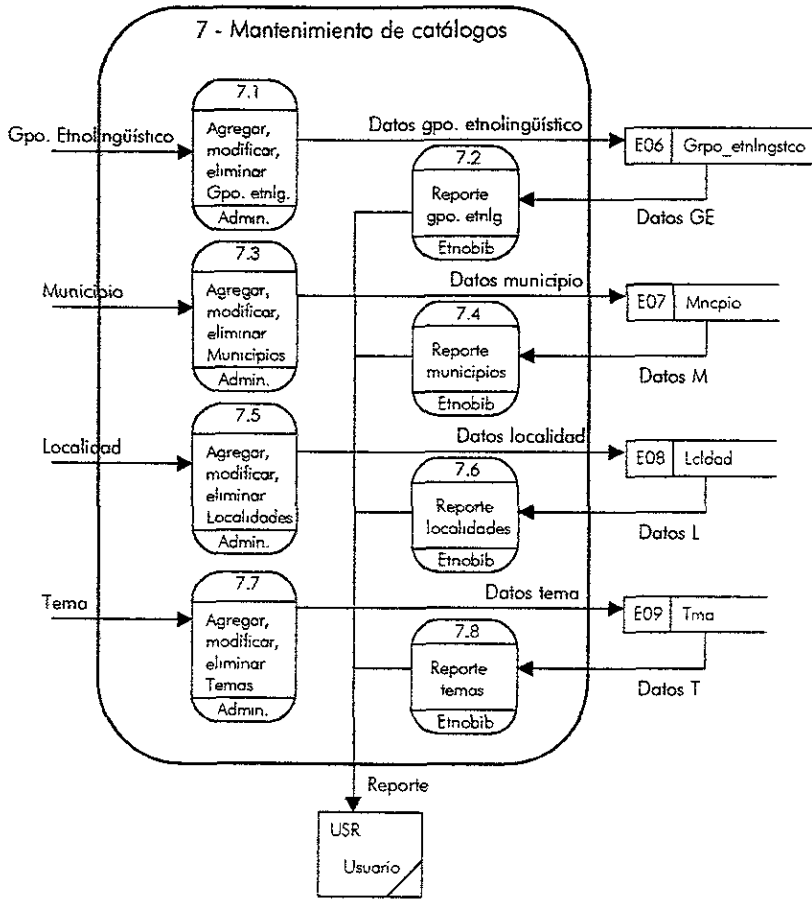


Fig. 5.4.8: Proceso "Mantenimiento de catálogos"



5.5 DISEÑO DEL SISTEMA

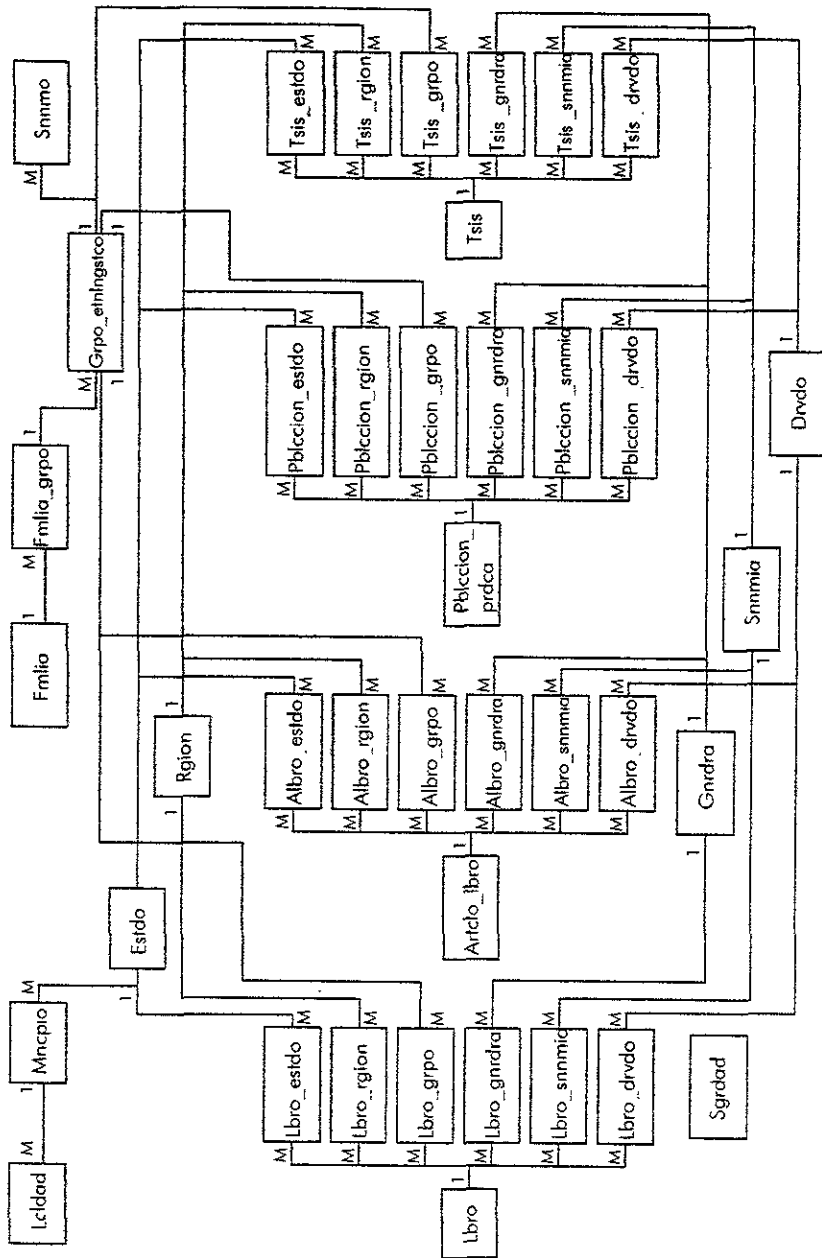
5.5.1 Base de datos

El diseño de bases de datos es un proceso de organización de los campos de datos necesarios para una o más aplicaciones, poniéndolos en una estructura organizada, la cual debe propiciar las relaciones necesarias entre los campos cumpliendo al mismo tiempo con las restricciones físicas del sistema específico de manejo de base de datos que se utilice.

Hay dos partes en el proceso: el diseño lógico de la base de datos, el cual consta del análisis y organización de los datos que serán utilizados por el sistema; y el diseño físico de la base de datos, que significa "traducir" el diseño lógico a la computadora, es decir, a un archivo físico almacenado en un medio de almacenamiento.

En el siguiente diagrama se presenta la base de datos diseñada para este proyecto, mostrando todas sus tablas y el tipo de relaciones que existen entre ellas. Inmediatamente se muestran los campos (atributos) que conforman cada una de las tablas, así como la descripción de cada uno de ellos: el nombre técnico, el tipo de dato que contiene (byte, texto, entero largo, memo, etc.) y el tamaño que delimita la capacidad de almacenamiento en cada atributo.

5.5.2 Diagrama Entidad/Relación



5.5.3 Definición de atributos y tablas

TABLA: ALBRO_DRVDO

Atributos

Nombre	Tipo	Tamaño
ARTL_id	Número (largo)	4
DVD_id	Número (largo)	4

Relaciones

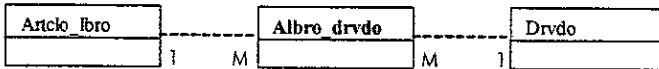


TABLA: ALBRO_ESTDO

Atributos

Nombre	Tipo	Tamaño
EDO_id	Número (largo)	4
ARTL_id	Número (largo)	4

Relaciones

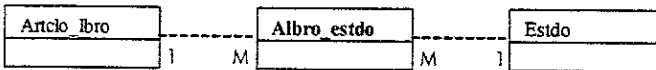


TABLA: ALBRO_GNRDRA

Atributos

Nombre	Tipo	Tamaño
ARTL_id	Número (largo)	4
GRA_id	Número (largo)	4

Relaciones

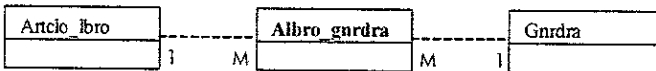


TABLA: ALBRO_GRPO

Atributos

Nombre	Tipo	Tamaño
ETL_id	Número (largo)	4
ARTL_id	Número (largo)	4

Relaciones

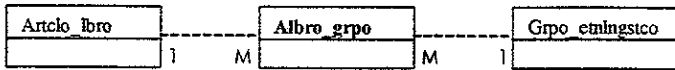


TABLA: ALBRO_REGION

Atributos

Nombre	Tipo	Tamaño
RGN_id	Número (largo)	4
ARTL_id	Número (largo)	4

Relaciones

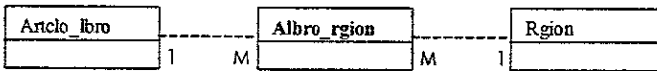


TABLA: ALBRO_SNNMIA

Atributos

Nombre	Tipo	Tamaño
ARTL_id	Número (largo)	4
SNN_id	Número (largo)	4

Relaciones

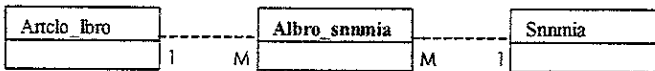


TABLA: ARTCLO_LBRO

Atributos

Nombre	Tipo	Tamaño
ARTL_id	Número (largo)	4
LBR_id	Número (largo)	4
ARTL_autor	Texto	120
ARTL_tlo	Texto	230
ARTL_idma	Texto	20
ARTL_año	Número (entero)	2
ARTL_mncpio	Texto	200
ARTL_icldad	Texto	200
ARTL_cnntrio	Memo	-
ARTL_pp_mcio	Número (entero)	2
ARTL_pp_fin	Número (entero)	2

Relaciones

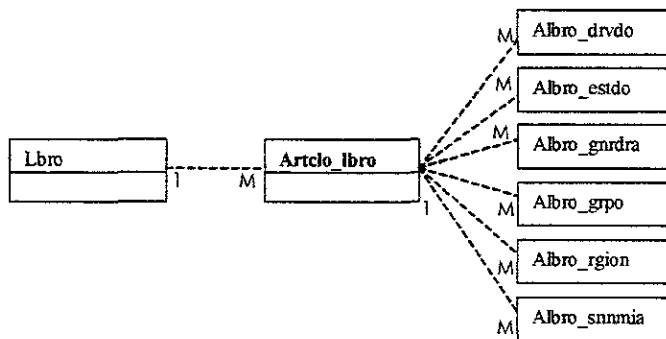


TABLA: PBLCCION_PRDCA

Atributos

Nombre	Tipo	Tamaño
PP_id	Número (largo)	4
PP_autor	Texto	110
PP_titlo	Texto	250
PP_rvsta	Texto	100
PP_editorial	Texto	100
PP_idma	Texto	20
PP_lgar	Texto	40
PP_epca	Texto	20
PP_nmro	Texto	25
PP_vlmen	Texto	25
PP_fcha_pblccion	Texto	60
PP_año	Número (entero)	2
PP_mncpio	Texto	200
PP_lcldad	Texto	200
PP_cmntrio	Memo	-
PP_pp_inicio	Número (entero)	2
PP_pp_fin	Número (entero)	2
PP_issn	Texto	20

Relaciones

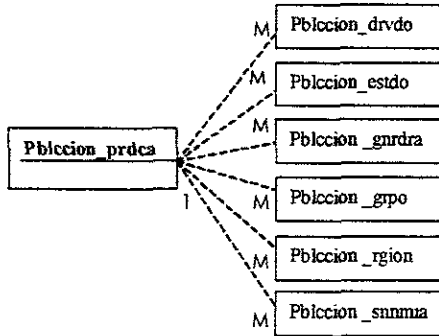


TABLA. DRVDO

Atributos

Nombre	Tipo	Tamaño
DVD_id	Número (largo)	4
DVD_nombre	Texto	40
SNN_id	Número (largo)	4

Relaciones

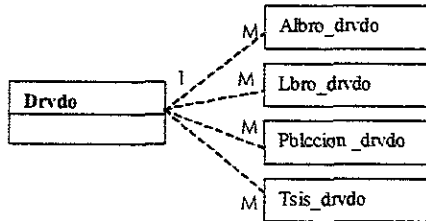


TABLA. ESTDO

Atributos

Nombre	Tipo	Tamaño
EDO_id	Número (largo)	4
EDO_nombre	Texto	21

Relaciones

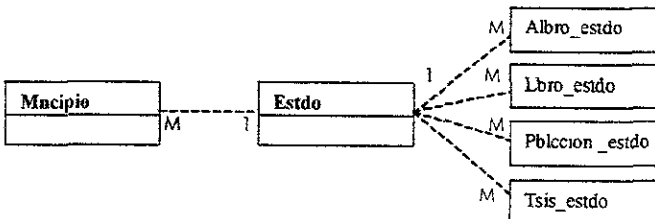


TABLA: FMLIA

Atributos

Nombre	Tipo	Tamaño
FML_id	Número (largo)	4
FML_nombre	Texto	25

Relaciones

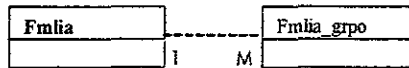


TABLA: FMLIA_GRP0

Atributos

Nombre	Tipo	Tamaño
GRP_id	Número (largo)	4
GRP_nombre	Texto	25
FML_id	Número (largo)	4

Relaciones

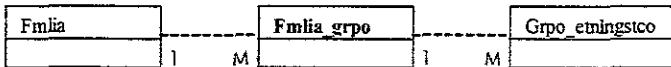


TABLA: GNRDRA

Atributos

Nombre	Tipo	Tamaño
GRA_id	Número (largo)	4
GRA_nombre	Texto	30

Relaciones

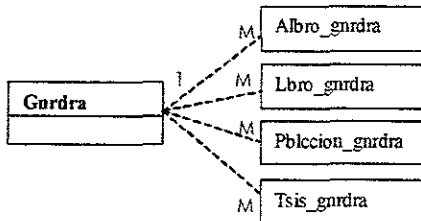


TABLA: GRPO_ETNNGSTCO

Atributos

Nombre	Tipo	Tamaño
ETL_id	Número (largo)	4
ETL_lngua	Texto	35
GRP_id	Número (largo)	4

Relaciones

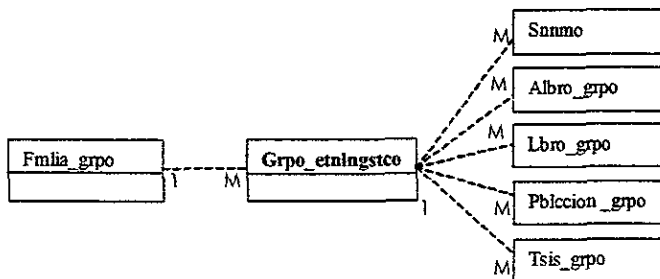


TABLA: LBRO

Atributos

Nombre	Tipo	Tamaño
LBR_id	Número (largo)	4
LBR_autor	Texto	120
LBR_titlo	Texto	230
LBR_idma	Texto	20
LBR_lgar	Texto	40
LBR_edtrial	Texto	160
LBR_año	Número (entero)	2
LBR_prmra_edcion	Número (entero)	2
LBR_srie	Texto	40
LBR_nmro_srie	Texto	25
LBR_mpa	Sí/No	1
LBR_fto	Sí/No	1
LBR_ilstrcion	Sí/No	1
LBR_nmro_pgnas	Número (entero)	2
LBR_cmntrio	Memo	-
LBR_bblgrfia	Memo	-
LBR_indce	Memo	-
LBR_mncpio	Texto	200
LBR_lcldad	Texto	200
LBR_edcion	Número (entero)	2
LBR_isbn	Texto	20

Relaciones

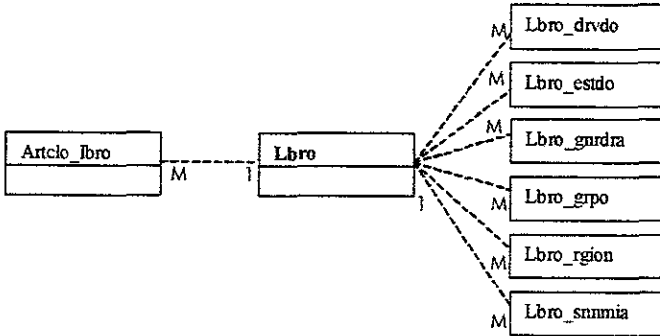


TABLA: LBRO_DRVDO

Atributos

Nombre	Tipo	Tamaño
LBR_id	Número (largo)	4
DVD_id	Número (largo)	4

Relaciones

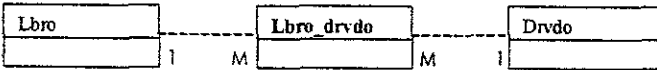


TABLA: LBRO_ESTDO

Atributos

Nombre	Tipo	Tamaño
EDO_id	Número (largo)	4
LBR_id	Número (largo)	4

Relaciones

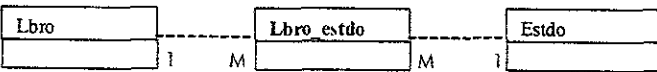


TABLA: LBRO_GNRDRA

Atributos

Nombre	Tipo	Tamaño
LBR_id	Número (largo)	4
GRA_id	Número (largo)	4

Relaciones

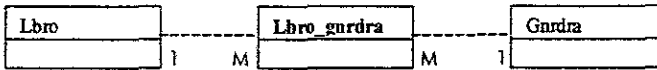


TABLA: LBRO_GRPO

Atributos

Nombre	Tipo	Tamaño
ETL_id	Número (largo)	4
LBR_id	Número (largo)	4

Relaciones

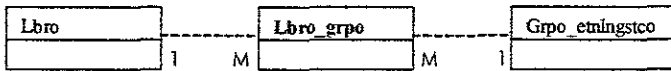


TABLA: LBRO_RGION

Atributos

Nombre	Tipo	Tamaño
RGN_id	Número (largo)	4
LBR_id	Número (largo)	4

Relaciones

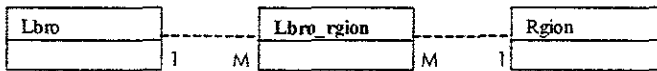


TABLA: LBRO_SNNMIA

Atributos

Nombre	Tipo	Tamaño
LBR_id	Número (largo)	4
SNN_id	Número (largo)	4

Relaciones



TABLA: LCLIDAD

Atributos

Nombre	Tipo	Tamaño
--------	------	--------

LCL_id	Número (largo)	4
LCL_nombre	Texto	50
MCP_id	Número (largo)	4

Relaciones

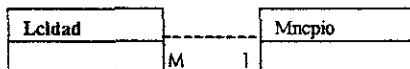


TABLA: MNCPIO

Atributos

Nombre	Tipo	Tamaño
MCP_id	Número (largo)	4
MCP_nombre	Texto	50
EDO_id	Número (largo)	4

Relaciones

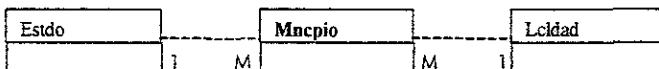


TABLA REGION

Atributos

Nombre	Tipo	Tamaño
RGN_id	Número (largo)	4
RGN_nombre	Texto	50

Relaciones

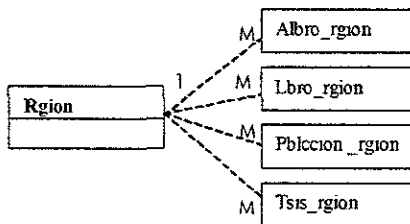


TABLA: PBLCCION_DRVDO

Atributos

Nombre	Tipo	Tamaño
PP_id	Número (largo)	4
DVD_id	Número (largo)	4

Relaciones

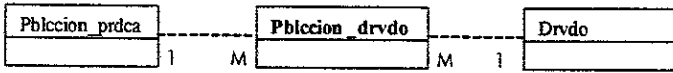


TABLA: PBLCCION_ESTDO

Atributos

Nombre	Tipo	Tamaño
EDO_id	Número (largo)	4
PP_id	Número (largo)	4

Relaciones

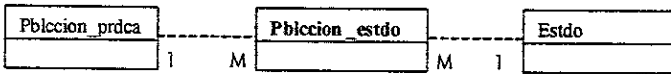


TABLA: PBLCCION_GNRDRA

Atributos

Nombre	Tipo	Tamaño
GRA_id	Número (largo)	4
PP_id	Número (largo)	4

Relaciones

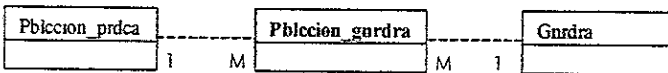


TABLA: PBLCCION_GRP0

Atributos

Nombre	Tipo	Tamaño
ETL_id	Número (largo)	4
PP_id	Número (largo)	4

Relaciones

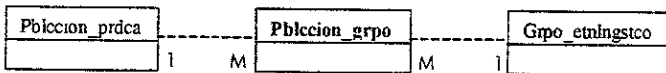


TABLA: PBLCCION_RGION

Atributos

Nombre	Tipo	Tamaño
RGN_id	Número (largo)	4
PP_id	Número (largo)	4

Relaciones

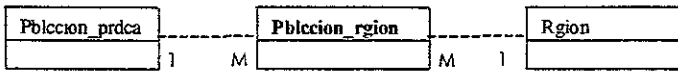


TABLA: PBLCCION_SNNMIA

Atributos

Nombre	Tipo	Tamaño
PP_id	Número (largo)	4
SNN_id	Número (largo)	4

Relaciones

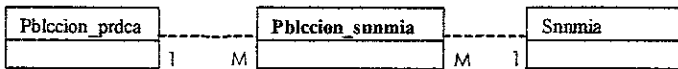


TABLA: SGRDAD

Atributos

Nombre	Tipo	Tamaño
SGR_id	Número (Byte)	1
SGR_psswr	Texto	10

TABLA: SNNMIA

Atributos

Nombre	Tipo	Tamaño
SNN_id	Número (largo)	4
SNN_nombre	Texto	80
GRA_id	Número (largo)	4
GRA_Tpo	Texto	1

Relaciones

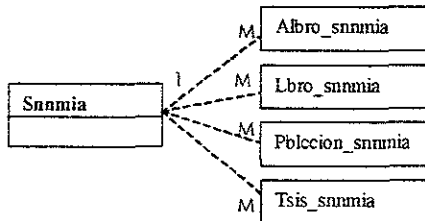


TABLA: SNNMO

Atributos

Nombre	Tipo	Tamaño
--------	------	--------

SNM_id	Número (largo)	4
SNM_nombre	Texto	40
ETL_id	Número (largo)	4

Relaciones

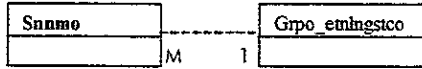


TABLA: TSIS

Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
TSS_autor	Texto	120
TSS_titulo	Texto	230
TSS_idma	Texto	20
TSS_universidad	Texto	100
TSS_grdo	Texto	80
TSS_lgar	Texto	40
TSS_nmro_pgnos	Número (entero)	2
TSS_año	Número (entero)	2
TSS_indce	Memo	-
TSS_cmntrio	Memo	-
TSS_bblgrfia	Memo	-
TSS_mncpio	Texto	200
TSS_iddad	Texto	200

Relaciones

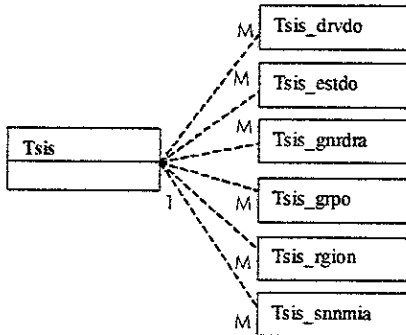


TABLA: TSIS_DRVDO

Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
DVD_id	Número (largo)	4

Relaciones

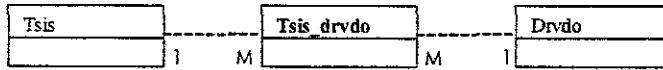


TABLA: TSS_ESTDO

Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
EDO_id	Número (largo)	4

Relaciones

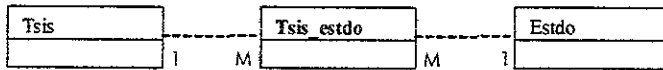


TABLA: TSS_GNRDRA

Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
GRÁ_id	Número (largo)	4

Relaciones

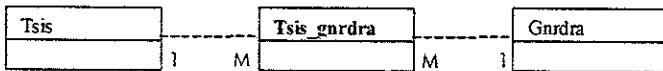


TABLA: TSS_GRPO

Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
ETL_id	Número (largo)	4

Relaciones

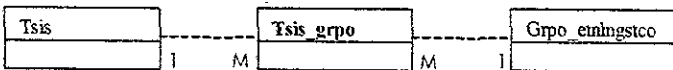


TABLA: TSS_REGION

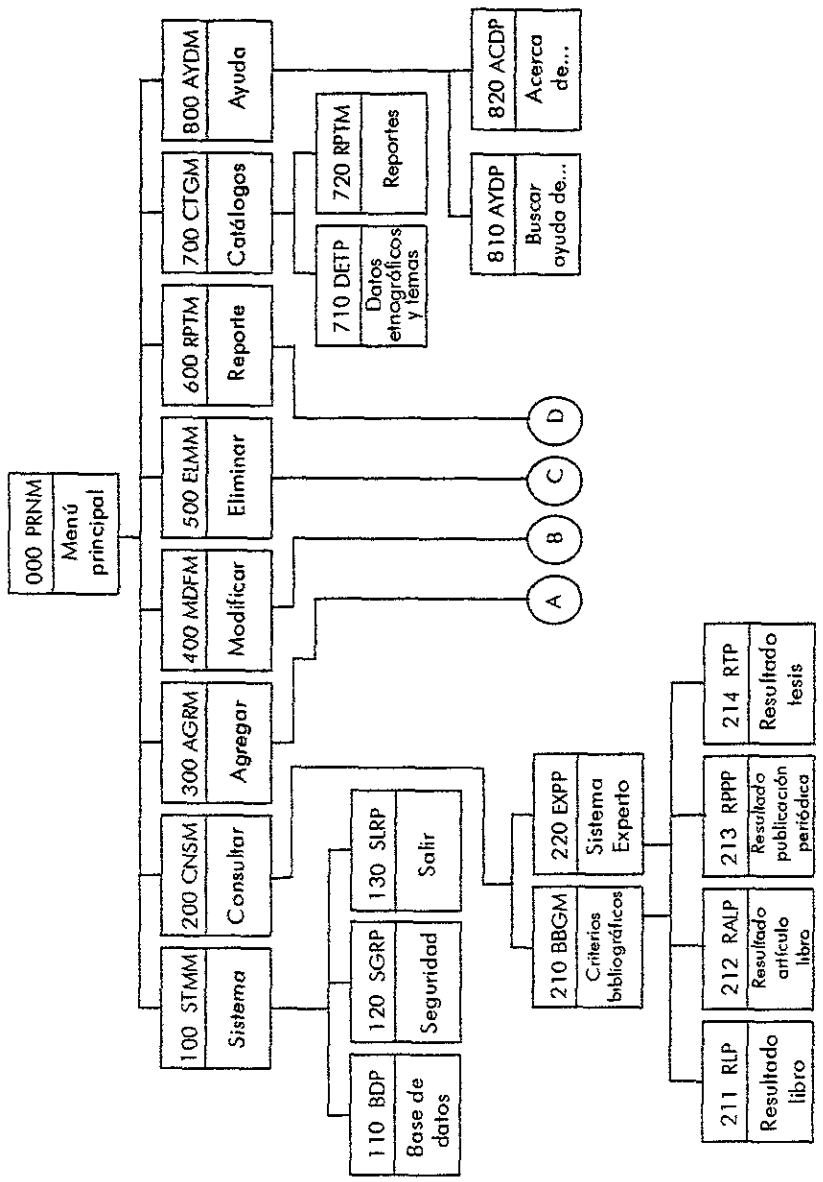
Atributos

Nombre	Tipo	Tamaño
TSS_id	Número (largo)	4
RGN_id	Número (largo)	4

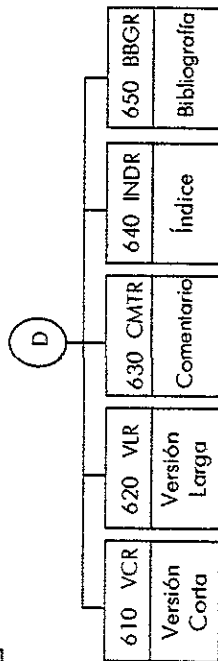
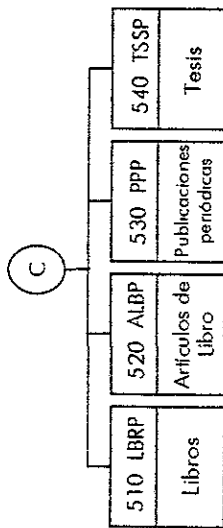
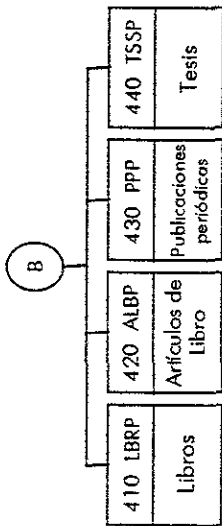
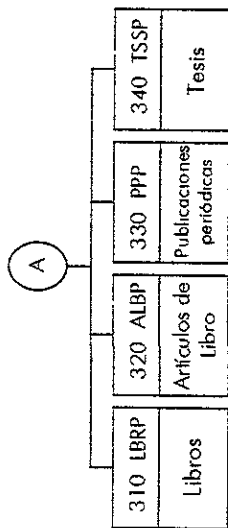
5.5.4 Diagrama estructurado

El siguiente diagrama muestra los principales módulos de los que se compone el sistema de información ETNOBIB, así como la jerarquía que existe entre ellos. Cada módulo se ejecuta a través de su correspondiente interfaz.

Diagrama Estructurado



Continuación



5.5.5 Máquina de inferencias

El motor de inferencia utilizado es el encadenamiento hacia adelante, ya que depende de antecedentes (los cuales son los criterios de búsqueda tecleados por el usuario), para con base en ello servir como antecedentes que puedan activar nuevas reglas, o bien, para brindar información correspondiente a la ficha buscada.

5.5.6 Base de reglas

La representación del conocimiento que se utilizó para el desarrollo del sistema experto fueron las reglas de producción (70 reglas), esto debido a que el experto humano empleaba un razonamiento similar para solucionar el problema:

“Si no existe la consulta con todos los criterios de búsqueda deseados, entonces descartar un criterio y realizar la consulta con los restantes; luego descartar otro criterio y volver a realizar la consulta con los restantes, y así sucesivamente, descartando otro criterio hasta terminar con ellos.”

Cada una de las reglas empleadas se basan en consultas (*queries*) a la base de datos. Enseguida se muestra el pseudocódigo que muestra la estructura del conjunto de reglas para el submódulo: *Exprto_Cnsitas_Articio_Lbro*.

n= criterios a consultar (número)

Comenzar

Elimina criterio n

Si criterio 'autor' = verdadero, entonces

realiza consulta,

Si existen fichas, entonces

obtén fichas autor.

si no existen fichas, entonces

decrementa n,

comenzar.

Terminar.

Terminar

Si criterio 'título' = verdadero, entonces

realiza consulta,

Si existen fichas, entonces

Si no existe un criterio anterior, entonces

obtén fichas título.

si existe un criterio anterior, entonces

obtén fichas título,

compara fichas título con fichas de un criterio anterior.

Si existen fichas en común, entonces

obtén fichas.

Si no existen fichas en común, entonces
decrementa n,
comenzar.
Terminar
Terminar
si no existen fichas, entonces
decrementa n,
comenzar.
Terminar.
Terminar.
Si criterio 'idioma' = verdadero, entonces
realiza consulta,
Si existen fichas, entonces
Si no existe un criterio anterior, entonces
obténc fichas idioma.
si existe un criterio anterior, entonces
obténc fichas idioma,
compara fichas idioma con fichas de un criterio anterior.
Si existen fichas en común, entonces
obténc fichas.
Si no existen fichas en común, entonces
decrementa n,
comenzar.
Terminar
Terminar
si no existen fichas, entonces
decrementa n,
comenzar.
Terminar.
Terminar.
Si criterio 'año' = verdadero, entonces
realiza consulta,
Si existen fichas, entonces
Si no existe un criterio anterior, entonces
obténc fichas año.
si existe un criterio anterior, entonces
obténc fichas año,
compara fichas año con fichas de un criterio anterior.
Si existen fichas en común, entonces
obténc fichas.
Si no existen fichas en común, entonces
decrementa n,
comenzar.
Terminar
Terminar

si no existen fichas, entonces
decrementa n,
comenzar.

Terminar.

Terminar.

Si criterio 'tema' = verdadero, entonces
realiza consulta,

Si existen fichas, entonces

Si no existe un criterio anterior, entonces
obtén fichas tema.

si existe un criterio anterior, entonces

obtén fichas tema,

compara fichas tema con fichas de un criterio anterior.

Si existen fichas en común, entonces

obtén fichas.

Si no existen fichas en común, entonces

decrementa n,

comenzar.

Terminar

Terminar

si no existen fichas, entonces
decrementa n,
comenzar.

Terminar.

Terminar.

Si criterio 'grupo etnolingüístico' = verdadero, entonces
realiza consulta,

Si existen fichas, entonces

Si no existe un criterio anterior, entonces
obtén fichas grupo etnolingüístico.

si existe un criterio anterior, entonces

obtén fichas grupo etnolingüístico,

compara fichas gpo etningstico con fichas de un criterio anterior.

Si existen fichas en común, entonces

obtén fichas.

Si no existen fichas en común, entonces

decrementa n,

comenzar.

Terminar

Terminar

si no existen fichas, entonces
decrementa n,
comenzar.

Terminar.

Terminar.

Si criterio 'región' = verdadero, entonces
realiza consulta,
Si existen fichas, entonces
 Si no existe un criterio anterior, entonces
 obtén fichas región.
 si existe un criterio anterior, entonces
 obtén fichas región,
 compara fichas región con fichas de un criterio anterior.
 Si existen fichas en común, entonces
 obtén fichas.
 Si no existen fichas en común, entonces
 decrementa n,
 comenzar.
 Terminar
Terminar
si no existen fichas, entonces
 decrementa n,
 comenzar.
Terminar.
Terminar.
Si criterio 'estado' = verdadero, entonces
realiza consulta,
Si existen fichas, entonces
 Si no existe un criterio anterior, entonces
 obtén fichas estado.
 si existe un criterio anterior, entonces
 obtén fichas estado,
 compara fichas estado con fichas de un criterio anterior.
 Si existen fichas en común, entonces
 obtén fichas.
 Si no existen fichas en común, entonces
 decrementa n,
 comenzar.
 Terminar
Terminar
si no existen fichas, entonces
 decrementa n,
 comenzar.
Terminar.
Terminar.
Si criterio 'municipio' = verdadero, entonces
realiza consulta,
Si existen fichas, entonces
 Si no existe un criterio anterior, entonces
 obtén fichas municipio.

si existe un criterio anterior, entonces
 obténgase fichas municipio,
 compara fichas municipio con fichas de un criterio anterior.
 Si existen fichas en común, entonces
 obténgase fichas.
 Si no existen fichas en común, entonces
 decrementa n ,
 comenzar.
 Terminar

Terminar
si no existen fichas, entonces
 decrementa n ,
 comenzar.
Terminar.

Terminar.
Si criterio 'localidad' = verdadero, entonces
 realiza consulta,
 Si existen fichas, entonces
 Si no existe un criterio anterior, entonces
 obténgase fichas localidad.
 si existe un criterio anterior, entonces
 obténgase fichas localidad,
 compara fichas localidad con fichas de un criterio anterior.
 Si existen fichas en común, entonces
 obténgase fichas.
 Si no existen fichas en común, entonces
 decrementa n ,
 comenzar.
 Terminar

Terminar
si no existen fichas, entonces
 decrementa n ,
 comenzar.
Terminar.

Terminar.
Concluir.

Nótese que el pseudocódigo anterior solo es aplicable cuando se realiza una consulta de artículos de libro, el cual tiene sus propios criterios de búsqueda. Para los demás casos (libros, publicaciones periódicas y tesis), la estructura es semejante a la anterior, con la diferencia de que cada uno de ellos tiene más, menos o distintos criterios de búsqueda. En el apéndice se puede observar parte del código programado para el módulo del sistema experto.

5.6 DESCRIPCIÓN DE LAS INTERFACES DEL SISTEMA

5.6.1 Consultas por criterios bibliográficos

El sistema permite realizar consultas con base en los criterios bibliográficos comunes (por autor, título, año e idioma) limitando la búsqueda al tipo de documento deseado (libro, artículo de libro, publicación periódica o tesis).

Todos los campos de datos permiten realizar búsquedas, ya sea usándolos como criterio único o en combinación con otros campos. Para encontrar la información deseada, es necesario cumplir ciertos criterios al momento de llenar un campo. Todos los campos pueden ser consultados usando * (asterisco) como signo equivalente a "todo". La mayoría de estos también pueden ser consultados por palabra, fragmento de palabra o letra. La figura 5.6.1 muestra la ventana correspondiente para consultar libros.

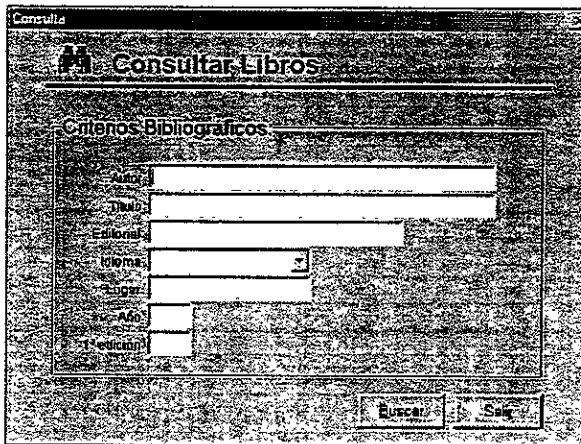


Fig. 5.6.1: Consulta bibliográfica de libros

Los campos con criterios especiales para realizar una búsqueda son:

- Autor(es): se escribe el primer apellido completo o en parte. El sistema sólo considera la primera palabra del campo.
- Título: se escribe cualquier palabra o letra. El sistema localizará la palabra o letra en cualquier parte del título.

Idioma: se puede escribir un idioma o elegir uno del listado que despliega el combo. Refiere al idioma en el que está escrito el texto.

- Año: se pueden buscar las publicaciones de un solo año o bien de un periodo. En este último caso, se deberá hacer click en el *checkbox* que aparece a la derecha de la caja de texto, después de lo cual aparece una nueva caja de texto, donde se introducirá el segundo año. La fecha no podrá ser menor a 1900.
- 1a. Edición: se pueden buscar las publicaciones por el año en que fue publicado originalmente y también por un periodo. En este último caso, se deberá hacer click en el *checkbox* que aparece a la derecha de la caja de texto, después de lo cual aparece una nueva caja de texto, donde se introducirá el segundo año. La fecha no podrá ser menor a 1900.

5.6.2 Consulta experta

Esta opción permite realizar consultas con base en los criterios bibliográficos, pero además considerando criterios de clasificación etnográficos y temáticos.

Se puede seleccionar esta opción para consultar individualmente cada tipo de documento (libro, artículo de libro, publicaciones periódicas o tesis), pero además se puede realizar una consulta considerando todos los tipos de documento simultáneamente. Ver la figura 5.6.2.

Fig. 5.6.2: Consulta experta

Los criterios de consulta son múltiples en la medida en que incluyen criterios bibliográficos, etnográficos y temáticos.

Para realizar la consulta es necesario llenar por lo menos un campo, de lo contrario el programa no ejecutará acción alguna. Cuanto mayor sea la cantidad de criterios introducidos, el resultado de la consulta será más específico, pero también se reducirán las probabilidades de que una ficha cumpla con todos los criterios solicitados.

Si no hay fichas que cumplan con los criterios, el sistema experto se encarga de volver a buscar para encontrar algún resultado aproximado.

5.6.3 Resultado de consulta

Una ventana de resultado aparece después de realizar una consulta exitosa (correspondiente a los distintos tipos de documento considerados: libro, artículo de libro, publicaciones periódicas y tesis). La figura 5.6.3 muestra un ejemplo de una consulta realizada a un libro.

ETNOBIB - [Resultado de Consulta]

LIBRO

Datos Bibliográficos

Autor(es): BARTOLOME, Miguel Alberto y Alicia BARABAS Idioma: Español

Título: La pluralidad en peligro. procesos de transfiguración y extinción cultural en Oaxaca ISBN: 970-30-329-3

Editorial: INAH-INI

Serie: Regiones de México No. 329

Lugar: México

Año: 1996 Edición: 1

Mapas: [Fichas] Ilustraciones: [Fichas]

Datos Etnográficos

Grupos étnolingüísticos: chonsal, chocho, ucateco, zaque

Regiones: Oaxaca

Estados: Oaxaca

Municipios:

Lugares:

Temas

Complementos

E-Consulta

Bibliografía

Mapas

Ilustraciones

Consultar Salir

Fig. 5.6.3: Resultado de consulta de libro

Cuando la búsqueda se realiza con la opción todo, se abre una ventana que muestra el total de fichas encontradas por cada tipo de documento. Ver figura 5.6.3'

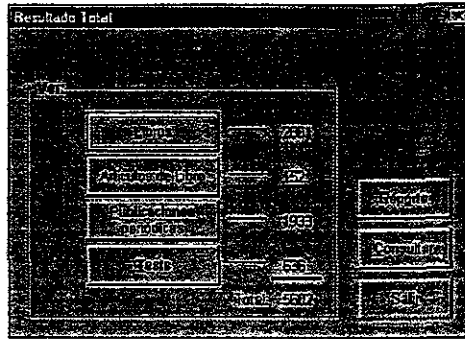


Fig. 5.6.3': Ventana de resultado total

5.6.4 Reportes

Una vez que la ventana de resultado de consulta ha aparecido, en el menú principal se habilita la opción Reporte, el cual permite ver el resultado de la consulta tal y como saldría impreso en una hoja de papel.

Independientemente del número de fichas que cumplieron los criterios de la consulta, el reporte desplegará un conjunto de 20 fichas a la vez.

Existen cinco tipos de reporte:

- Corto - muestra los campos que contienen información bibliográfica.
- Largo - muestra los campos que contienen información bibliográfica, etnográfica y temática.
- Comentario - muestra información adicional, comentarios y reseñas del documento mostrado en la ventana resultado de consulta.
- Bibliografía - muestra la bibliografía consultada por él o los autores del libro o tesis mostrado en la ventana resultado de consulta.
- Índice - muestra el índice del libro o tesis mostrado en la ventana resultado de consulta.

Cualquiera de los reportes mencionados anteriormente es desplegado en la ventana de reportes. En dicha ventana es posible imprimir y/o mandar a archivo el reporte en varios formatos.

Solo en el caso en que se ha seleccionado consultar por medio de la opción todo, el reporte se imprimirá desde la ventana de Resultado total, debe de hacerse esto continuamente hasta terminar con el total de fichas. La única forma de imprimir continuamente es permaneciendo en la ventana de resultado de consulta de la opción todo.

5.6.5 Ventana de reportes

La ventana de reportes tiene varias funciones: visualizar tres distintas ampliaciones en que puede observarse el reporte, imprimir la información contenida en la ventana de reportes, guardar el reporte como archivo para su uso en otras aplicaciones o exportar el reporte directamente a un destino de correo electrónico. La figura 5.6.5 muestra una ficha bibliográfica lista para imprimir.

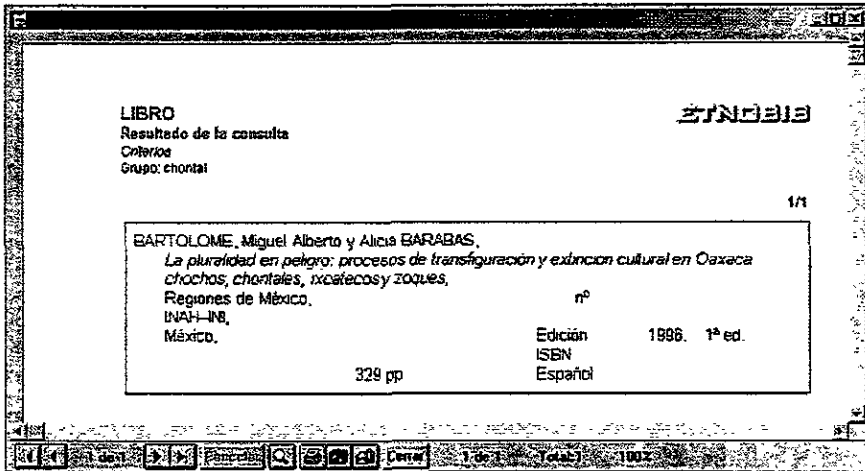


Fig. 5.6.5: Ventana de impresión

Existen algunos otros indicadores en la ventana, como el contador de páginas, que indica el número de hoja en la que se encuentra y el total de hojas del reporte. Por ejemplo, 1 de 2, indica que se encuentra en la página 1 de un informe de 2 páginas.

5.6.6 Agregar

Para agregar una ficha (libro, artículo de libro, publicación periódica o tesis) es necesario hacer click en el menú "Agregar" y después seleccionar el tipo de documento. Inmediatamente aparece una ventana para llenar los campos de información bibliográfica de una nueva ficha. La figura 5.6.6 muestra la ventana correspondiente para agregar un nuevo libro a la base de datos.

Fig. 5.6.6: Ventana para agregar un libro

La información de cada ficha bibliográfica está separada en campos para mejorar la capacidad de búsqueda. Además, los campos están ordenados en cuatro grandes apartados:

Información bibliográfica – conjunta los campos propios de una ficha bibliográfica comenzando por autor (apellidos en altas, nombres en altas y bajas, de los dos primeros autores), título (del libro, de la tesis o del artículo, según sea el caso), lugar de publicación y editorial (véase listado en siglas). Se incluye, además, un campo de idioma que permite buscar las fichas con este criterio.

Las fichas de libro incluyen, además: nombre de la serie o colección a la que pertenece y número de serie; año de la publicación y año de primera edición; número total de páginas (pp.)

Las fichas de artículo de libro incluyen, además: los nombres de los autores del artículo y del compendio; los títulos del artículo y del compendio, y el número de páginas en que se ubica el artículo.

Las fichas de publicaciones periódicas incluyen, además: el nombre de la publicación; número, volumen, época, fecha, año, y páginas en las que se ubica la publicación.

Las fichas de tesis incluyen, además: nombre de la universidad donde se presenta la tesis y grado alcanzado.

- Información etnográfica – conjunta los campos de ubicación espacial (región, estado, municipio y localidad) y el identificador de grupo etnolingüístico (lengua o término de autoadscripción).
- Información temática – se forma de un campo en donde pueden ser ingresados uno o más temas.
- Complementos – conjunta algunos botones que remiten a información adicional (en los casos en los que se tuvo acceso al documento). Estos son: bibliografía (consultada por los autores del documento), índice (del libro o tesis) y comentarios (información aclaratoria sobre el contenido del documento). Además, en la ventana de artículo de libro hay un botón que remite a la ficha del libro o compendio de donde deriva la ficha consultada.

5.6.7 Modificar

Esta función permite modificar las fichas de libro, artículo de libro, publicaciones periódicas o tesis. Es necesario hacer click en el menú "modificar", después elegir el tipo de documento e inmediatamente aparece la ventana para buscar la(s) ficha(s) a modificar.

El resultado de la consulta se muestra en la misma ventana que se emplea para agregar la ficha correspondiente. El procedimiento para modificar es similar al realizado para agregar una ficha.

5.6.8 Eliminar

Esta función permite eliminar las fichas de libro, artículo de libro, publicaciones periódicas o tesis. Es necesario hacer click en el menú "eliminar", después elegir el tipo de documento e inmediatamente aparece la ventana para buscar la ficha a eliminar.

El resultado de la consulta se muestra en la misma ventana que se emplea para agregar la ficha correspondiente. Para eliminar alguna otra ficha es necesario repetir el proceso.

5.6.9 Catálogos

Esta ventana (ver figura 5.6.9) permite dar mantenimiento a los catálogos que se tiene en la base de datos, maneja cuatro tipos de catálogos:

- Grupos etnolingüísticos - despliega familias, grupos, lenguas y sinónimos.
- Regiones - despliega las regiones.
- Municipios y localidades - despliega los municipios y localidades de un estado.
- Temas - despliega las palabras generadoras, sinonimias, conceptos derivados y conceptos relacionados.

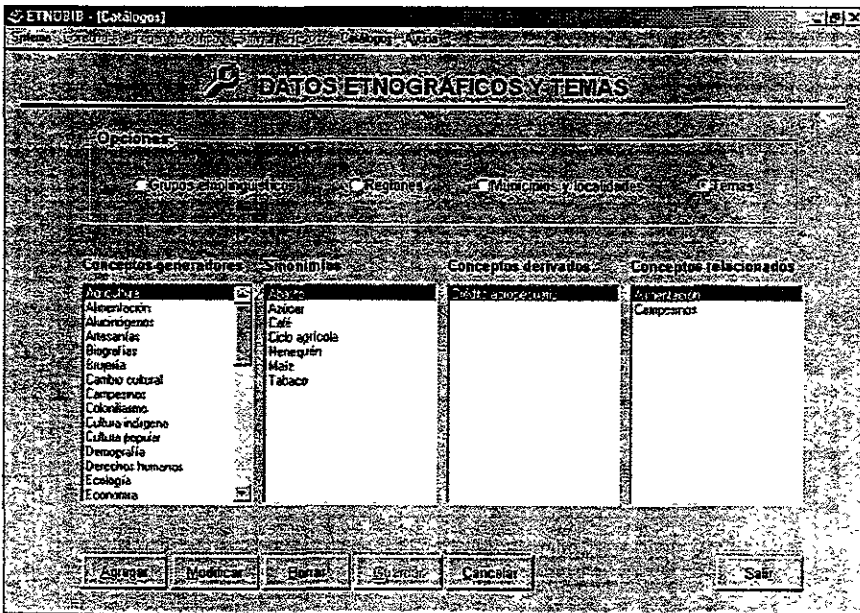


Fig. 5.6.9: Mantenimiento de catálogos

Se tienen las funciones agregar, modificar, borrar y guardar para realizar cualquier tipo de transacción sobre los catálogos en la base de datos.

5.6.10 Reportes de catálogos

Esta función permite generar el reporte de cada uno de los catálogos, o bien, genera el reporte de un conjunto de palabras en particular relacionadas a un solo tópic. En la figura 5.6.10 se muestra dicha ventana.

The screenshot shows a software window titled "ETNOBIB - [Reporte]" with a subtitle "RepORTE DE CATALOGOS". The interface is divided into three main sections for filtering search results:

- Grupos Familiares:** Includes radio buttons for "Familias", "Grupos", "Sinónimos", and "Sinónimos Relativos". It also features three input fields: "Grupos de la Familia", "Lenguas del Grupo", and "Distancia de la Lengua".
- Regiones:** Includes radio buttons for "Regiones" and "Municipios". It features two input fields: "Municipios del Estado" and "Localidades del Municipio".
- Temas:** Includes radio buttons for "Conceptos Generales (CG)", "Sinónimos del CG", "Sinónimos", "Conceptos Derivados (CD)", "CD de la Sinonimia", "Conceptos Relacionados (CR)", and "Conceptos Relativos". It features three input fields: "Sinónimos del CG", "CD de la Sinonimia", and "Conceptos Relativos".

At the bottom right of the window, there are two buttons: "Reporte" and "Salir".

Fig. 5.6.10: Reporte de catálogos

5.7 PRUEBAS

5.7.1 Pruebas de unidad

Conjunto de pruebas efectuadas por cada programador antes de la integración de la unidad en un sistema más grande. Se realizaron dos tipos de pruebas:

- a) Pruebas funcionales – consistió en la aplicación de valores nominales de entrada para los cuales se conocían los resultados esperados, además de valores límites y valores especiales.
- b) Pruebas de tensión – aquellas diseñadas para “romper” intencionalmente la unidad.

5.7.2 Pruebas de integración

Estrategia empleada para integrar los componentes de un sistema de *software* en un todo funcionando. Consiste en pruebas de unidad, seguidas por pruebas de subsistemas y luego por pruebas del sistema completo.

- a) Pruebas de unidad – su objetivo era descubrir errores en los módulos individuales del sistema.
- b) Pruebas de subsistemas – se verificó la operación de las interfaces entre módulos en el subsistema. Se probaron tanto las interfaces de control como las de datos.
- c) Pruebas del sistema – se verificaron cuidadosamente las interfaces, la lógica de decisión, el flujo del control, los procedimientos de recuperación, la eficiencia global, la capacidad y las características de tiempo del sistema entero.

5.7.3 Pruebas de aceptación

Implicaron la planeación y la ejecución de pruebas funcionales y de tensión para demostrar que el sistema implementado satisface los requisitos. Se aplicaron dos conjuntos de pruebas de aceptación: aquellas realizadas por los programadores, y las realizadas por el cliente.

5.7.4 Validación y verificación del conocimiento

Es el proceso que consta de validar y verificar el conocimiento hasta que su calidad sea óptima. Esta etapa se aplicó prácticamente en todo el proceso de desarrollo del sistema inteligente.

En lo que refiere a las pruebas de la base de conocimiento, éstas se enfocaron a los siguientes puntos:

- Verificación – se realizó observando que el sistema brindará respuestas correctas para cada caso posible. Para esto se planteó la siguiente pregunta:

“¿se está construyendo correctamente el sistema?”

- Validación – se realizaron usando las capacidades de explicación que tienen las herramientas de desarrollo. Si estas explicaciones son coherentes, se puede decir que se ha estructurado correctamente el conocimiento. Se trata de resolver la siguiente pregunta:

“¿se está construyendo el sistema correcto?”

CONCLUSIONES

Los sistemas de información han ganado importancia debido al creciente valor de la información como un recurso. Los datos que estos sistemas manejan normalmente son el recurso máspreciado de una empresa. Dicha información concierne, tanto a los procesos y los recursos internos de la organización (bienes, personal, departamentos, etc.), como a la información de fuentes externas (proveedores, clientes, etc.). Cada vez más, las organizaciones están incorporando sus procesos manuales a sistemas de información, y están confiando sus datos a las computadoras. Su objetivo es lograr que los procesos sean más eficientes y que la información esté disponible rápidamente. Por estas razones se construyó el sistema inteligente ETNOBIB.

El sistema ETNOBIB es un instrumento de gran utilidad para todos aquellos vinculados con la etnografía en nuestro país, ya que la base de datos con que cuenta contiene el mayor volumen de información en cuanto a bibliografía etnográfica se refiere. Las ventajas sobre aquellos sistemas con los que cuenta el INAH es que tiene un mejor funcionamiento; cuenta con una mayor cantidad de criterios de búsqueda y es posible hacer combinaciones entre ellos; y cuenta con una gran variedad de reportes para el usuario y para aquellos encargados de dar mantenimiento y actualizar la base de datos.

ETNOBIB supera las restricciones de los sistemas de consulta que tiene el INAH actualmente en cuanto a los resultados de consultas se refiere. Tiene la capacidad de brindar resultados, ya sea exactos o muy aproximados a la información que el usuario desea obtener. No resulta de vital importancia que el usuario teclee correcta o exactamente los criterios, por ejemplo, el nombre de un autor, el título del libro, el año exacto de publicación o la lengua. En caso de algún fallo, el sistema experto comienza a realizar las consultas y arrojará un resultado que probablemente sea justo el deseado por el usuario, de no ser así, el resultado tendrá información relacionada a lo que el usuario desea obtener.

Durante el desarrollo del sistema, resultó particularmente problemático el intercambio de opiniones e ideas con el experto humano. Este es un problema que normalmente surge en la relación "ingeniero del conocimiento-experto humano". Sin embargo, el ingeniero del conocimiento debe tener capacidad, sensibilidad, inteligencia y paciencia para llevar a cabo una buena relación interpersonal con el experto humano, y así derribar estas limitaciones para lograr que el proyecto tenga éxito.

Se consideraron las limitaciones y restricciones de *hardware* y *software*, de tal forma que la interfaz gráfica es un elemento que se diseñó basándose en las necesidades y peticiones del INAH, siendo sencillas de manejar, de fácil aprendizaje, y que al mismo tiempo permiten al sistema cumplir los alcances y objetivos planteados en cuanto a funcionalidad y capacidad se refiere. Las herramientas para desarrollar sistemas

expertos, los lenguajes de programación y las bases de datos tienen gran alcance en nuestros días, pero puede ser muy complicado cuando se combinan entre sí, sin embargo, en este caso se logró la exitosa aplicación de diferentes técnicas de la computación.

La aplicación de las técnicas de inteligencia artificial al desarrollo de *software* brinda oportunidades de desarrollo que antes no hubieran sido posibles. Los sistemas expertos son una de las ramas de la inteligencia artificial que ha tenido mayor auge, sin embargo, aún existe desconfianza y escepticismo en la implementación de estas nuevas técnicas, por lo que es necesario dar un mayor impulso a todas aquellas propuestas relacionadas con la inteligencia artificial.

Mediante este sistema se consiguió realizar una arquitectura híbrida entre una base de datos relacional y un sistema experto, dejando ver que la unión de estos elementos de la ingeniería en computación ayuda a crear un camino para la resolución de problemas diferente de las técnicas tradicionales, no siendo menos eficiente, sino igual o mejor a lo ya conocido.

BIBLIOGRAFÍA

- Mark L. Gillenson. "Introducción a la base de datos". 1988. McGraw-Hill. 1º edición. México.
- C. J. Date. "Introducción a los sistemas de bases de datos". 1986. Addison-Wesley Iberoamericana. 3º edición. E.U.A.
- Alice Y. H. Tsai. "Sistemas de base de datos. Administración y uso". 1990. Prentice-Hall Hispanoamericana, S. A. 1º edición. México.
- Richard Frost. "Bases de datos y sistemas expertos. Ingeniería del conocimiento". 1986. Ediciones Díaz de Santos, S.A. España.
- Wendy B. Rauch-Hindin. "Aplicaciones de la inteligencia artificial en la actividad empresarial, la ciencia y la industria (Fundamentos-aplicaciones)". 1985. Ediciones Díaz de Santos, S.A. España.
- Stuart J. Russell, Peter Norvig. "Inteligencia artificial: un enfoque moderno". 1996. Prentice Hall Hispanoamericana, S.A. 1º edición. México.
- Varios autores bajo la coordinación de Jose Mompín Poblet. "Inteligencia artificial. Conceptos, técnicas y aplicaciones". Serie: Mundo Electrónico. 1987. Marcombo Boixareau Editores. España.
- Efraim Turban. "Expert systems and applied artificial intelligence". 1992. Macmillan, USA.
- Kemper V. Nicolás, "Sistemas Expertos Aplicados a Negocios". 1998. Notas del Curso Internacional, Editorial UNI, Lima, Perú.
- Roger S. Pressman. "Software engineering". 1992. McGraw-Hill. 3º edición. E.U.A.
- Richard Fairley. "Ingeniería de software". 1988. McGraw-Hill. 1º edición. México.
- Ghezzi Carlo, Jazayeri Mehdi, Mandrioli, Dino. "Fundamentals of Software Engineering". 1991. Prentice-Hall. E.U.A.

APÉNDICE

A continuación se muestran el código empleado en una sección del sistema experto. Esta sección se encarga de los artículos de libro.

```
Public Sub Exprto_Cnsitas_Articlo_Lbro()
Dim MyRs_UnSorted As Recordset
Dim c, cont, Cnta, num As Integer
Dim Total_aux_ARTL_FML, Total_ARTL_FML, ARTL_FML(), m, i, k, Dfrnte As Integer
Dim Año_Inicio, Año_Fin, Id_Exprto_Fnal(500) As Integer

For c = 0 To 500
  If Id_Exprto_Fnal(c) <> 0 Then
    Id_Exprto_Fnal(c) = 0
  Else
    Exit For
  End If
Next

For c = 1 To Nmro_Cnsitas_Exprtas
  Exste_Cnslta = False
  num = 1
  If Exprto_Cnsitas(Nmro_Cnsitas_Exprtas - c) <> num Then
    If Bndra_Autor = True Then 'AUTOR
      Set MyRs = MyDb.OpenRecordset("SELECT ARTL_id FROM Articlo_Lbro WHERE
      ARTL_autor LIKE " & "" & TXTautor_articlo_lbro.Text & "" & "ORDER BY
      ARTL_autor", dbOpenSnapshot)
      Vrfca_Mnsje
      If BNDRA_SLIR = False Then
        Obtencion_de_ids
        Exste_Cnslta = True
      Else
        GoTo Sgnte_Brrda
      End If
    End If
  End If

  num = 2
  If Exprto_Cnsitas(Nmro_Cnsitas_Exprtas - c) <> num Then
    If Bndra_Tito = True Then 'TITULO
```

```

Set MyRs = MyDb.OpenRecordset("SELECT ARTL_id FROM Artclo_lbro WHERE
ARTL_titlo LIKE " & "" & TXTtitlo_artclo_lbro.Text & "" & "ORDER BY ARTL_autor",
dbOpenSnapshot)
Vrfca_Mnsje
If BNDRA_SLIR = False Then
    If Exste_Cnsltta = False Then 'Cuando es consulta padre.
        Obtencion_de_ids
        Exste_Cnsltta = True
    Else 'Cuando es consulta hija.
        Obtencion_de_ids_Tmpral
        Compara_ids
        If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
        End If
    End If
Else
    GoTo Sgnte_Brrda
End If
End If
End If

num = 3
If Exprto_Cnsltta(Nmro_Cnsltta_Exprtas - c) <> num Then
    If Bndra_idma = True Then 'IDIOMA
        Set MyRs = MyDb.OpenRecordset("SELECT ARTL_id FROM Artclo_lbro WHERE
ARTL_idma LIKE " & "" & TXTidma_artclo_lbro.Text & "" & "ORDER BY
ARTL_autor", dbOpenSnapshot)
        Vrfca_Mnsje
        If BNDRA_SLIR = False Then
            If Exste_Cnsltta = False Then 'Cuando es consulta padre.
                Obtencion_de_ids
                Exste_Cnsltta = True
            Else 'Cuando es consulta hija.
                Obtencion_de_ids_Tmpral
                Compara_ids
                If BNDRA_SLIR = True Then
                    GoTo Sgnte_Brrda
                End If
            End If
        Else
            GoTo Sgnte_Brrda
        End If
    End If
End If

```

num = 6

```
If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
  If Año_Rngo = True Or Año = True Then 'AÑO
    If Año = True Then 'Cuando es solo un año.
      Año_Incio = CInt(TXTaño_artclo_lbro1.Text)
      Set MyRs_UnSorted = MyDb.OpenRecordset("SELECT ARTL_id, ARTL_autor
      FROM Artclo_Lbro WHERE ARTL_año = " & Año_Incio, dbOpenSnapshot)
      MyRs_UnSorted.Sort = "ARTL_autor"
      Set MyRs = MyRs_UnSorted.OpenRecordset()
      MyRs_UnSorted.Close
      Vrfca_Mnsje
      If BNDRA_SLIR = False Then
        If Exste_Cnslt = False Then 'Cuando es consulta padre.
          Obtencion_de_Ids
          Exste_Cnslt = True
        Else 'Cuando es consulta hija.
          Obtencion_de_Ids_Tmpral
          Compara_Ids
          If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
          End If
        End If
      Else
        GoTo Sgnte_Brrda
      End If
    ElseIf Año_Rngo = True Then 'Cuando es un rango de años.
      Año_Incio = CInt(TXTaño_artclo_lbro1.Text)
      Año_Fin = CInt(TXTaño_artclo_lbro2.Text)
      Set MyRs_UnSorted = MyDb.OpenRecordset("SELECT ARTL_id, ARTL_autor
      FROM Artclo_Lbro WHERE ARTL_año >= " & Año_Incio & "AND
      ARTL_año <= " & Año_Fin, dbOpenSnapshot)
      MyRs_UnSorted.Sort = "ARTL_autor"
      Set MyRs = MyRs_UnSorted.OpenRecordset()
      MyRs_UnSorted.Close
      Vrfca_Mnsje
      If BNDRA_SLIR = False Then
        If Exste_Cnslt = False Then 'Cuando es consulta padre.
          Obtencion_de_Ids
          Exste_Cnslt = True
        Else 'Cuando es consulta hija.
          Obtencion_de_Ids_Tmpral
          Compara_Ids
          If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
          End If
        End If
      End If
    End If
  End If
End If
```

```

End If
    Else
        GoTo Sgnte_Brrda
    End If
End If
End If
End If

num = 8
If Exprto_Cnsltas(Nmro_Cnsltas_Exprias - c) <> num Then
    If Bndra_Tma = True Then "TEMA"
        If LSTtma_tpo.List(0) = "r" Then
            Set MyRs = MyDb.OpenRecordset("SELECT Albro_gnrdr.ARTL_id FROM
            Artclo_lbro INNER JOIN Albro_gnrdra ON Artclo_lbro.ARTL_id =
            Albro_gnrdra.ARTL_id WHERE Albro_gnrdra.GRA_id = " &
            CInt(LSTtma_id.List(0)) & " ORDER BY Artclo_lbro.ARTL_autor",
            dbOpenSnapshot)
        ElseIf LSTtma_tpo.List(0) = "s" Then
            Set MyRs = MyDb.OpenRecordset("SELECT Albro_snmnia.ARTL_id FROM
            Artclo_lbro INNER JOIN Albro_snmnia ON Artclo_lbro.ARTL_id =
            Albro_snmnia.ARTL_id WHERE Albro_snmnia.SNN_id = " &
            CInt(LSTtma_id.List(0)) & " ORDER BY Artclo_lbro.ARTL_autor",
            dbOpenSnapshot)
        ElseIf LSTtma_tpo.List(0) = "d" Then
            Set MyRs = MyDb.OpenRecordset("SELECT Albro_drvdo.ARTL_id FROM
            Artclo_lbro INNER JOIN Albro_drvdo ON Artclo_lbro.ARTL_id =
            Albro_drvdo.ARTL_id WHERE Albro_drvdo.DVD_id = " &
            CInt(LSTtma_id.List(0)) & " ORDER BY Artclo_lbro.ARTL_autor",
            dbOpenSnapshot)
        End If
        Vrfca_Mnsje
        If BNDRA_SLIR = False Then
            If Exste_Cnslt = False Then 'Cuando es consulta padre.
                Obtencion_de_Ids
                Exste_Cnslt = True
            Else 'Cuando es consulta hija.
                Obtencion_de_Ids_Tmpral
                Compara_Ids
                If BNDRA_SLIR = True Then
                    GoTo Sgnte_Brrda
                End If
            End If
        End If
    Else
        GoTo Sgnte_Brrda
    End If

```

```

End If
End If

num = 9
If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
  If Bndra_Tma = True Then 'TEMA
    If LSTtma_id.ListCount = 2 Or LSTtma_id.ListCount = 3 Then
      If LSTtma_tpo.List(1) = "r" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_gnrdrd.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_gnrdrd ON Artclo_lbro.ARTL_id =
        Albro_gnrdrd.ARTL_id WHERE Albro_gnrdrd.GRA_id = " &
        CInt(LSTtma_id.List(1)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      ElseIf LSTtma_tpo.List(1) = "s" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_snnmia.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_snnmia ON Artclo_lbro.ARTL_id =
        Albro_snnmia.ARTL_id WHERE Albro_snnmia.SNN_id = " &
        CInt(LSTtma_id.List(1)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      ElseIf LSTtma_tpo.List(1) = "d" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_drvdo.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_drvdo ON Artclo_lbro.ARTL_id =
        Albro_drvdo.ARTL_id WHERE Albro_drvdo.DVD_id = " &
        CInt(LSTtma_id.List(1)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      End If
      Vrfca_Mnsje
      If BNDRA_SLIR = False Then
        If Exste_Cnslta = False Then 'Cuando es consulta padre.
          Obtencion_de_ids
          Exste_Cnslta = True
        Else 'Cuando es consulta hija.
          Obtencion_de_ids_Tmpral
          Compara_ids
          If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
          End If
        End If
      Else
        GoTo Sgnte_Brrda
      End If
    End If
  End If
End If

```

```

num = 10
If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
  If Bndra_Tma = True Then 'TEMA
    If LSTtma_id.ListCount = 3 Then
      If LSTtma_tpo.List(2) = "r" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_gnrdr.a.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_gnrdr.a ON Artclo_lbro.ARTL_id =
        Albro_gnrdr.a.ARTL_id WHERE Albro_gnrdr.a.GRA_id = " &
        CInt(LSTtma_id.List(2)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      ElseIf LSTtma_tpo.List(2) = "s" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_snmia.a.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_snmia.a ON Artclo_lbro.ARTL_id =
        Albro_snmia.a.ARTL_id WHERE Albro_snmia.a.SNN_id = " &
        CInt(LSTtma_id.List(2)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      ElseIf LSTtma_tpo.List(2) = "d" Then
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_drvdo.a.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_drvdo.a ON Artclo_lbro.ARTL_id =
        Albro_drvdo.a.ARTL_id WHERE Albro_drvdo.a.DVD_id = " &
        CInt(LSTtma_id.List(2)) & " ORDER BY Artclo_lbro.ARTL_autor",
        dbOpenSnapshot)
      End If
      Vrfca_Mnsje
      If BNDRA_SLIR = False Then
        If Exste_Cnslta = False Then 'Cuando es consulta padre.
          Obtencion_de_ids
          Exste_Cnslta = True
        Else 'Cuando es consulta hija.
          Obtencion_de_ids_Tmpraf
          Compara_ids
          If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
          End If
        End If
      Else
        GoTo Sgnte_Brrda
      End If
    End If
  End If
End If

num = 11
If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
  If Bndra_Grpo = True Then 'GRUPO

```

```

If Total_Lngua = 0 Then
    ReDim Id_Exprto_Lbro(0)
    Id_Exprto_Lbro(0) = 0
    Exste_Cnsita = Faise
    GoTo Sgnte_Brrda
End If
Total_aux_ARTL_FML = 0
m = 0
For i = 0 To Total_Lngua - 1
    Set MyRs = MyDb.OpenRecordset("SELECT Albro_grpo.ARTL_id FROM
    Artclo_lbro INNER JOIN Albro_grpo ON Artclo_lbro.ARTL_id =
    Albro_grpo.ARTL_id WHERE Albro_grpo.ETL_id = " & id_GPO(i) & "
    ORDER BY Artclo_lbro.ARTL_autor", dbOpenSnapshot)
    If Not MyRs.EOF Then
        MyRs.MoveLast
        Total_aux_ARTL_FML = Total_aux_ARTL_FML + MyRs.RecordCount
    End If
Next
MyRs.Close
If Total_aux_ARTL_FML > 0 Then
    ReDim ARTL_FML(Total_aux_ARTL_FML - 1)
    Total_ARTL_FML = 0
    For i = 0 To Total_Lngua - 1
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_grpo.ARTL_id FROM
        Artclo_lbro INNER JOIN Albro_grpo ON Artclo_lbro.ARTL_id =
        Albro_grpo.ARTL_id WHERE Albro_grpo.ETL_id = " & id_GPO(i) & "
        ORDER BY Artclo_lbro.ARTL_autor", dbOpenSnapshot)
        Do While MyRs.EOF = False
            For k = 0 To Total_aux_ARTL_FML - 1
                If ARTL_FML(k) <> MyRs![ARTL_id] Then
                    Dfrnte = Dfrnte + 1
                Else
                    Exit For
                End If
            Next
            If Dfrnte = Total_aux_ARTL_FML Then
                ARTL_FML(m) = MyRs![ARTL_id]
                m = m + 1
                Total_ARTL_FML = Total_ARTL_FML + 1
            End If
            MyRs.MoveNext
            Dfrnte = 0
        Loop
    Next
MyRs.Close

```

```

If Exste_Cnslta = False Then
    ReDim Id_Artclo_Lbro(Total_ARTL_FML - 1)
    For k = 0 To Total_ARTL_FML - 1
        Id_Artclo_Lbro(k) = ARTL_FML(k)
    Next
    Total_Artclo_Lbro = Total_ARTL_FML
    Exste_Cnslta = True
Else
    ReDim Id_Temp(Total_ARTL_FML - 1)
    For k = 0 To Total_ARTL_FML - 1
        Id_Temp(k) = ARTL_FML(k)
    Next
    Total_Temp = Total_ARTL_FML
    Exste_Cnslta = True
    Compara_Ids
    If BNDRA_SLIR = True Then
        GoTo Sgnite_Brrda
    End If
End If
Else
    If Opcion_Tdo = False Then
        Exste_Cnslta = False
        ReDim Id_Expcto_Lbro(0)
        Id_Expcto_Lbro(0) = 0
        GoTo Sgnite_Brrda
    End If
End If
End If
End If

num = 12
If Expcto_Cnsltas(Nmro_Cnsltas_Expctas - c) <> num Then
    If Bndra_Rgion = True Then 'REGIÓN
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_rgion.ARTL_id FROM Rgion
INNER JOIN (Artclo_lbro INNER JOIN Albro_rgion ON Artclo_lbro.ARTL_id =
Albro_rgion.ARTL_id) ON Rgion.RGN_id = Albro_rgion.RGN_id WHERE
Rgion.RGN_nmbre LIKE " & "" & CMBrgion.Text & "" & " ORDER BY
Artclo_lbro.ARTL_autor", dbOpenSnapshot)
        Vrfca_Mnsje
        If BNDRA_SLIR = False Then
            If Exste_Cnslta = False Then 'Cuando es consulta padre.
                Obtiencion_de_ids
                Dstnct_ID
                Exste_Cnslta = True
            Else
                'Cuando es consulta hija.

```

```

        Obtencion_de_Ids_Tmpral
        Compara_Ids
        Dstnct_ID
        If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
        End If
    End If
Else
    GoTo Sgnte_Brrda
End If
End If
End If

```

num = 13

```

If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
    If Bndra_Estdo = True Then 'ESTADO
        Set MyRs = MyDb.OpenRecordset("SELECT Albro_estdo.ARTL_id,
        Artclo_lbro.ARTL_autor FROM Artclo_lbro INNER JOIN (Estdo INNER JOIN
        Albro_estdo ON Estdo.EDO_id = Albro_estdo.EDO_id) ON Artclo_lbro.ARTL_id
        = Albro_estdo.ARTL_id WHERE Estdo.EDO_nombre LIKE " & "" & CMBestdo.Text
        & "" & " ORDER BY Artclo_lbro.ARTL_autor", dbOpenSnapshot)
        Vrfca_Mnsje
        If BNDRA_SLIR = False Then
            If Exste_Cnsita = False Then 'Cuando es consulta padre.
                Obtencion_de_Ids
                Dstnct_ID
                Exste_Cnsita = True
            Else 'Cuando es consulta hija.
                Obtencion_de_Ids_Tmpral
                Compara_Ids
                Dstnct_ID
                If BNDRA_SLIR = True Then
                    GoTo Sgnte_Brrda
                End if
            End If
        Else
            GoTo Sgnte_Brrda
        End If
    End If
End If

```

num = 14

```

If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
    If bndra_mncpio = True Then 'MUNICIPIO

```

```

Set MyRs = MyDb.OpenRecordset("SELECT ARTL_id FROM Artclo_Lbro WHERE
ARTL_mncpio LIKE " & "*" & CMBmncpio.Text & "*" & "ORDER BY ARTL_autor",
dbOpenSnapshot)
Vrfca_Mnsje
If BNDRA_SLIR = False Then
    If Exste_Cnslta = False Then 'Cuando es consulta padre.
        Obtencion_de_Ids
        Exste_Cnslta = True
    Else 'Cuando es consulta hija.
        Obtencion_de_Ids_Tmpral
        Compara_Ids
        If BNDRA_SLIR = True Then
            GoTo Sgnte_Brrda
        End If
    End If
Else
    GoTo Sgnte_Brrda
End If
End If
End If

```

```

num = 15
If Exprto_Cnsltas(Nmro_Cnsltas_Exprtas - c) <> num Then
    If bndra_Icldad = True Then 'LOCALIDAD
        Set MyRs = MyDb.OpenRecordset("SELECT ARTL_id FROM Artclo_Lbro WHERE
ARTL_Icldad LIKE " & "*" & CMBIcldad.Text & "*" & "ORDER BY ARTL_autor",
dbOpenSnapshot)
        Vrfca_Mnsje
        If BNDRA_SLIR = False Then
            If Exste_Cnslta = False Then 'Cuando es consulta padre.
                Obtencion_de_Ids
                Exste_Cnslta = True
            Else 'Cuando es consulta hija.
                Obtencion_de_Ids_Tmpral
                Compara_Ids
                If BNDRA_SLIR = True Then
                    GoTo Sgnte_Brrda
                End If
            End If
        Else
            GoTo Sgnte_Brrda
        End If
    End If
End If
End If

```

```

Sgnte_Brrda:
If Id_Exprto_Lbro(0) <> 0 Then
  i = 0
  For k = 0 To Total_Artclo_Lbro - 1
    Id_Exprto_Fnal(cont) = Id_Exprto_Lbro(k)
    cont = cont + 1
  Next
End If
Next

c = 0
Cnta = 0
If Id_Exprto_Fnal(c) <> 0 Then
  For c = 0 To 100
    If Id_Exprto_Fnal(c) <> 0 Then
      Cnta = Cnta + 1
    Else
      Exit For
    End If
  Next
  Total_Artclo_Lbro = Cnta
  ReDim Id_Artclo_Lbro(Cnta - 1)
  For c = 0 To Cnta - 1
    Id_Artclo_Lbro(c) = Id_Exprto_Fnal(c) 'lds finales.
  Next
  Crtio_Autor = TXTautor_artclo_lbro.Text
  Crtio_Titlo = TXTtlo_artclo_lbro.Text
  Crtio_Idma = TXTidma_artclo_lbro.Text
  If TXTaño_artclo_lbro2.Text <> "" Then
    Crtio_Año = TXTaño_artclo_lbro1.Text & "-" & TXTaño_artclo_lbro2.Text
  ElseIf TXTaño_artclo_lbro1.Text <> "" Then
    Crtio_Año = TXTaño_artclo_lbro1.Text
  Else
    Crtio_Año = ""
  End If
  Crtio_Grpo = CMBgrpo.Text
  Crtio_Rgion = CMBrgion.Text
  Crtio_Estdo = CMBestdo.Text
  Crtio_Mncpio = CMBmncpio.Text
  Crtio_Lcldad = CMBlcldad.Text
  If LSTagrgar.ListCount = 1 Then
    Crtio_Tma = LSTagrgar.List(0)
  ElseIf LSTagrgar.ListCount = 2 Then
    Crtio_Tma = LSTagrgar.List(0) & ", " & LSTagrgar.List(1)
  ElseIf LSTagrgar.ListCount = 3 Then

```

```
Critrio_Tma = LSTagrgar.List(0) & ", " & LSTagrgar.List(1) & ", " & LSTagrgar.List(2)
Else
    Critrio_Tma = ""
End if
Rstido_Articlo_Lbro.Show
Unload Me
Else
    MsgBox "No existen artículos relacionados al que está consultando.", vbOKOnly +
vbExclamation, "Aviso"
    MousePointer = 0
End If
End Sub
```