CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
INSTITUTO POLITÉCNICO NACIONAL

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

TITULO

VISUALIZACIÓN DE UNA BASE DE DATOS ESPACIAL

TESIS PARA OBTENER EL GRADO DE:

DOCTOR EN CIENCIAS
EN LA ESPECIALIDAD DE COMPUTACIÓN

QUE PRESENTA:

M.C. JU SHI-GUANG

DIRECTOR DE TESIS

DR.SERGIO VICTOR CHAPA VERGARA

XD
CLASIF.: 96.6
ADQUIS.: RI-14848
PROCED.: 29 Oct. 96
PROCED.: TESIS-1996
$

A mi esposa

Zhang Xiao-Hong

A mi hija

Ju Lin-Na

AGRADECIMIENTOS

Estoy agradecido con mis profesores y compañeros cuya ayuda profesional, convirtieron la realización de este trabajo en una tarea muy agradable. En particular deseo agradecer profundamente al Dr. Sergio Víctor Chapa Vergara por sus enseñazas, consejos y su valiosa amistad, al M.C. Noé Sierra y Ing. Pedro Alday por sus aportaciones y sugerencias.

También agradezco a la Dra. Ana Ma. Antonia Martínez Enríquez, Dr. Angel Ortega Herrega, Dr. Manuel Sabino Lazo Cortés, Dr. Adolfo Guzmán Arenas y Dra. Yolanda Fernández Ordoñez por sus valiosos comentarios y sugerencias.

Finalmente, me gustaria dar las gracias a la Sección de Computación del Departamento de Ingeniería Eléctrica del Centro de Investagación y de Estudios Avanzados del I.P.N, por las facilidades ofrecidas durante el desarrollo de este trabajo.

Ju Shi-Guang

México D.F. 6 agosto de 1996,

INDICE

OBJETIVO GENERAL DEL PROYECTO

El proyecto "**Visualización de una Base de Datos Espacial**" (VISDA) considera dos objetivos primordiales. El **primero**, estriba en la conceptualización y el diseño de un nuevo lenguaje visual de consulta para el procesamiento de información geográfica que en particular se aplica a la exploración petrolera. Y el **segundo,** considera el desarrollo e implementación de un gestor de bases de datos espaciales.

La aportación principal de VISDA consiste en que la especificación del lenguaje es de forma visual y manejan información espacial. La especificación de las operaciones se lleva a cabo mediante elementos visuales que manejan información geográfica relativa a la cartografía de las regiones vinculada con información de: pozos, secciones sísmicas y geología, entre otras. En consecuencia, para el manejo de información espacial y nominal, se desarrolló un motor de bases de datos espaciales que incluye un modelo lógico relacional de datos y un modelo físico de datos basado en árboles B y multilistas.

Los objetivos fueron logrados en consecución obteniendo primeramente un motor de base de datos espacial el cual incluye un modelo relacional de datos y un gestor físico basado en árboles B y multilistas. Posteriormente, se procedio a proponer un nuevo lenguaje de consulta y manejo de datos que se implemento desarrollando toda una interfaz visual sobre el motor de base de datos. El desarrollo en X-windows consideró un ambiente visual para la definición de datos y un lenguaje visual de consulta. En ambos casos, el objeto de visualización son primordialmente las entidades geográficas.

Dado el enfoque de aplicación del lenguaje, se lleva a cabo un caso de estudio, en donde el modelo es una base de datos con un enfoque a exploración petrolera.

INTRODUCCIÓN

El tema de **Visualización** dentro de computación ha surgido rápidamente, presentandose en diversos tópicos como son: visualización científica, medios ambientes visuales y programación visual. Siendo para este último, la comunicación del humano y la máquina ha sido la razón principal para su amplio desarrollo. Con la idea de tener un estado del arte de lenguajes y programación visual, Ephraim P. Glinert presentó en 1990 un primer compendio en dos volumenes denominado "Visual Programming Enviroments" [Glin90(a)]; en donde se exploren en terminos de tutorial las alternativas visuales. Mientras el primer volumen comprende aplicaciones y temas; el segundo[Glin90(b)] trata directamente con los paradigmas y los sistemas. Casi al mismo tiempo aparecen dos revistas Journal of Visual Languages and Computing y Visualzation and Computer Graphics(transaction del IEEE). Relativo al tema con base de datos en donde trata con tópicos de visualización de información y accesos visuales a las bases de datos se tiene la publicación vigente IEEE Transactions on Knowledge and Data Engineering. En particular, en la Sección de Computación del CINVESTAV, la investigación en el área de lenguajes y programación visual inicia en esta década principalmente. Se tienen los desarrollos en lenguajes visuales aplicados a bases de datos como son: [Chap91],[Hern94], [Alda96], [Mend96], y [Tecl96]; y de lenguajes de proposito general como [Sier95]. En este contexto, mi trabajo de investigación se planteo con el interés de desarrollar un lenguaje visual que hiciera gestión sobre base de datos espaciales o pictográficas[Ju92] y [Chap94a]. Para tal fin, se consideró dos áreas de trabajo:

a) Los Lenguajes Visuales (LV), concebidos como un nuevo paradigma en donde el lenguaje usa símbolos visuales para la definición y el manejo de la información.

b) Y las bases de datos espaciales (BDE), cuya característica fundamental es que incluyen entidades gráficas como parte de los datos almacenados en ellas.

En otro orden de ideas, la importacia economica que tiene para México la exploración petrolera, nos ha llevado ha desarrollar Sistemas de Información Geográfica (SIG) aplicado a la exploración petrolera. Algunos estudios [Chap93] sobre la definición de la conceptualización y estructura lógica de la información geográfica de interés para PEMEX, desprendio en un primer sistema [Fior96] desarrollado en PC y ambientes de sistema operativo DOS.

De lo anterior, con estas premisas, se definió el presente proyecto: VISDA (VIsualization of Spatial DAtabase), cuyo objetivo principal es tener un ambiente visual para: recuperar, salvar, manipular y entender gran cantidad de la información contenida en una base de datos de este tipo. VISDA alcanza este objetivo aprovechando las características de los LV's junto con las nuevas técnicas para el manejo de información espacial, conformando un Sistema cuyo usuario final puede no estar familiarizado con el manejo y programación de una BD.

En la figura siguiente (Figura 1) se presenta la organización de este trabajo, de acuerdo a las dos líneas de investigación que lo conforman:

En el ***capitulo uno*** se justifica y se describe porqué la Base de Datos Espacial es el núcleo de diversas aplicaciones. Se describen las características principales de los sistemas de información espacial, incluyendo: los dispositivos de entrada y salida de gráficos; y el sistema de gestión de almacenamiento de los datos pictográficos. La arquitectura del sistema se definió considerando que la base de datos pictórica y la alfanumérica se integran para proporcionar una interfaz uniforme; aunque las representaciones y los procesos deben distinguirse claramente. Se define, además, la terminología relacionada con visualización incluyendo algunos ejemplos de sistemas que trabajan bajo esta metodología. Por último, se presenta la filosofía para la construcción de VISDA. Estos conceptos proporcionan el marco teórico sobre el que gira todo el proyecto.

En el ***capitulo dos*** se muestra el modelado conceptual de la base de datos espacial aplicada a la exploración petrolera. Se usa el sistema EVE [ALDA96] para la creación de la base de datos tomando como prototipo el modelo de datos propuesto en "Exploración petrolera de Pemex" según [CHAP93].

Uno de los aspectos principales dentro del diseño de sistemas de bases de datos espaciales son las estructuras de datos que contemplen una representación gráfica [Sammet90]. De esta manera, en el ***capitulo tres*** se muestran las estructuras de datos utilizadas en la implementación de la base de datos. Se trata con bases de datos convencionales que han desarrollado varios tipos de modelos con estructura espacial, como son: Quadtrees, R-trees y R^+-trees. En nuestro caso, el sistema tiene una estructura de datos que incorpora dos tipos de archivos: a) Archivo para datos nominales y b) Archivo para datos gráficos. Mientras el primero, usa estructuras de árbol-B, y el segundo utiliza estructuras de multilistas, respectivamente.

En el ***capitulo cuatro*** se muestra el lenguaje de manipulación de datos (DML) que apoya el manejo y procesamiento de los objetos de la base de datos bajo un ambiente de visualización. En esta parte, se explican las caraterísticas del DML, junto con la estructura y la representación gráfica de los archivos, los registros y la base de datos en general.

Cuando se quiere recuperar o consultar cualquier tipo de base de datos, es necesario tener un medio para lograr este propósito. De aquí se originan los llamados lenguajes de consulta, de lo cual se habla en el ***capítulo cinco.*** Iniciamos dando un panorama general de estos lenguajes, para particularizar en un lenguaje de consulta gráfico, Card Query Language (CQL), el cual se define formalmente a CQL. Y en referencia al mismo en el ***capítulo seis*** se describe la interfaz visual de usuario bajo la cual trabaja.

Para mostrar la capacidad de CQL, en ***capítulo siete*** se compara el lenguaje CQL con el lenguaje comercial SQL, mostrando la bondades del primero fundamentalmente como lenguaje visual y en el manejo de información gráfica.

ESQUEMA DEL PROYECTO

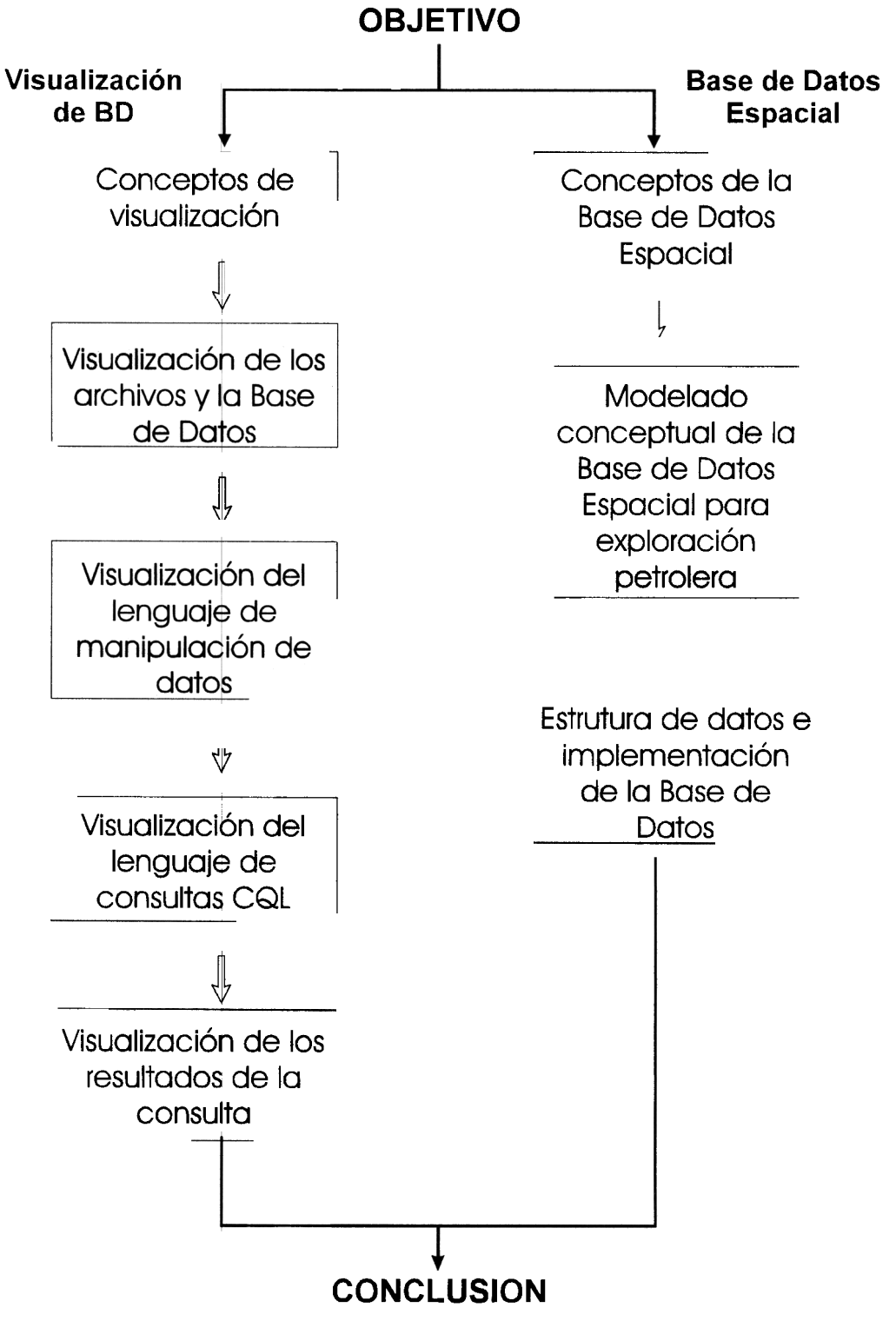


Figura Líneas de investigación desarrolladas

El ***capítulo ocho*** nos detalla como se visualizan los resultados obtenidos de una consulta. Iniciamos haciendo un estudio de diversas representaciones gráficas para mapas, con la finalidad de obtener un conjunto estándar de los símbolos gráficos empleados. Posteriormente, se describen los algoritmos más importantes para el trazado de mapas, los cuales son la representación del resultado de una consulta.

CAPITULO 1

VISUALIZACIÓN Y BASES DE DATOS ESPACIALES

Las computadoras se han convertido en una herramienta poderosa para la producción rápida y económica de ilustraciones, prácticamente no existe ninguna área en la cual no puedan utilizarse los despliegues gráficos con alguna ventaja; así que no es sorprendente hallar gráficas por computadoras en una infinidad de aplicaciones. Las primeras aplicaciones de la graficación por computadora se llevaron a cabo con equipos costosos y complicados. El objetivo, desde el inicio, fue el de desarrollar sistemas que tuvieran una comunicación interactiva entre el humano y la máquina. Con el decremento del costo del "hardware" la tecnología de graficación interactiva por computadora presentó un avanze significativo derivando en una gran cantidad de aplicaciones (revista Computer Graphics and Applications).

De esta manera cada día se incorporaron nuevos subsistemas de graficación dentro de otros mas complejos que tienen sus usos en: la administración, la industria, la medicina y el gobierno, entre otros. Aunque, los usos son diversos la aplicaciones gráficas han llegado a ser comunes y los subsistemas de graficación han tenido que formar parte inherente de otros que realizan tareas mas generales. En particular, un de ellos son las bases de datos espaciales o pictográficas, las cuales pueden almacenar y recuperar cierta información gráfica, es decir gestionar sobre una base de datos espacial.

Los sistemas de base de datos espacial por sus enfoques dan una primera taxonomia (que en terminos en ingles): Office Information (OI), Computer Aided Design(CAD), Imagen Understanding (IU), Computer Integrated Manufactured (CIM) y Geographical Information System (GIS). Para todos ellos lo común es la base caracterizada por la aplicación que se pretende llevar a cabo y un sistema de gestión con su interfaz visual con los usuarios (ver figura 1-1).

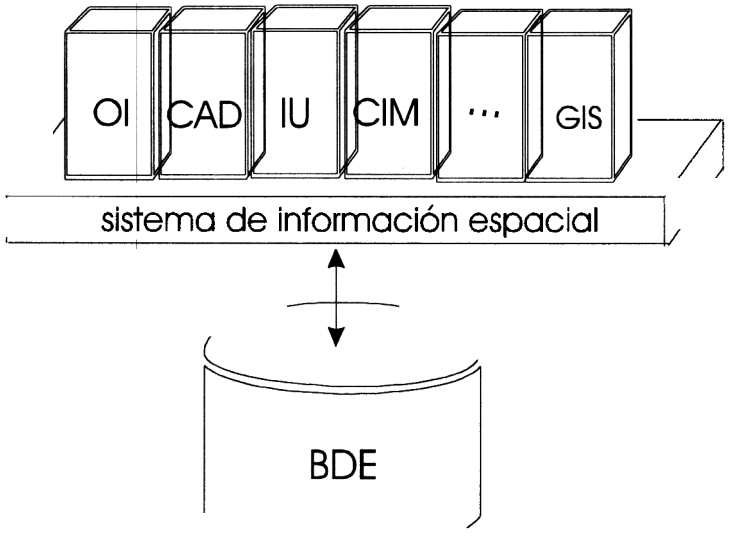


Figura 1-1 La base de datos espacial es núcleo de los sistemas de aplicación

De lo mencionado anteriarmente, podemos decir que de acuerdo al enfoque se tienen seis áreas principales de aplicación [CHAN89].

1) Sistema de computación de la quinta generación.

Los sistemas de computación de la quinta generación son los sistemas que procesan información basada en conocimiento[Moto81], e integran funciones que les dan la capacidad para ser considerados como máquinas inteligentes. Una de las funciones son las interfaces inteligentes capaces de entender las imagenes y el lenguaje natural. En este sentido, la base de conocimiento deberá esperar una especificación en terminos espaciales. El propósito será tener una interfaz inteligente de usuario capaz de una comunicación pictográfica y de entendimiento de imágenes.

2) Sistema de Información de Oficina (OI).

En las oficinas se requiere manejar documentos y formas. En casi todos los sistemas de información se requieren ciertos campos con una representación natural gráfica, por ejemplo: firma, fotografía, y algunos símbolos especiales entre otros.

Los documentos frecuentemente contienen datos tipo texto y tipo gráficos, pero algunas veces es mejor tratarlos como gráficos que necesitan generarse , editarse, guardarse, recuperarse y transmitirse.

3) Sistema de diseño asistido por computadora (CAD).

Desde hace varios años, con el uso más extenso de las gráficas por computadora, CAD ha sido un auxiliar en el diseño, en donde los métodos son poderosas herramientas para el diseño de partes. Cuando se han especificado las dimensiones de un objeto al sistema de computación, los diseñadores pueden observar cualquier lado del objeto para apreciar cómo será después de su construcción. Pueden hacerse cambios experimentales con libertad ya que, a diferencia del dibujo mecánico manual, el sistema CAD incorpora rápidamente modificaciones en el despliegue del objeto.

El sistema CAD es tan importante en esta área que se aplicó en la base de datos pictorial y su importancia se incrementa cada día en muchas universidades y laboratorios que ya han emprendiendo investigaciones sobre el diseño de la base de datos de CAD/CAM (computer-aided manufacturing), que es muy parecido en el diseño de la base pictorial por lo que recibió gran atención en los países avanzados.

4) Sistema de reconocimiento de imágenes (IU).

Los sistemas de reconocimiento de imágenes IU (imagen understanding) son de gran interés para las industrias involucradas con la defensa nacional. Actualmente, su interés se expandió a otras industrias, como la aérea (aerospace) y de semiconductores. Una combinación del reconocimiento de imágenes que es la base de conocimiento y el manejo de información pictórica es necesaria para diseñar un sistema complejo de reconocimiento de imágenes. Y el sistema de información pictórica es una herramienta útil e importante.

5) Sistema de manufactura integrado en computadora (CIM)

Para un sistema de visión por computadora en robots y CAD/CAM, el sistema de información pictorial también trabaja como un subsistema. Por ejemplo, en la automatización de inspección de chips de VLSI (Very Large Scale Integrated), existen miles de patrones de marca que se necesitan almacenar en la base de datos de patrones la cual es una base de datos pictorial.

En el sistema de CIMS (Computer Integrated Manufacturing System), se integran varias bases heterogéneas de datos que forman e integran un repertorio de información, lo cual es una consecuencia importante. La base de datos pictorial es útil no solo por sus grandes aplicaciones sino como una herramienta de descripción de otra base de datos. (por ejemplo, en el diseño de CAD o la base de datos de CAM).

6) Sistema de información geográfica (GIS)

Los GIS son sistemas de computadora que asisten al usuario en el procesamiento de datos geográficos. Manejan información espacial-nominal que va desde la captura hasta el despliegue dentro de un espacio gráfico en mapas. Los GIS contienen fundamentalmente una base de datos (espacial-nominal), con su sistema manejador que permite realizar la gestión de la información dentro de un ámbito de aplicación del análisis espacial.

1.1 EL SISTEMA DE INFORMACIÓN ESPACIAL

Una sistema de información espacial es un sistema que: controla y maneja los dispositivos de entrada, salida de gráficas; tiene un sistema de almacenamiento e interfaces de comunicación gráfica y proporciona un conjunto de datos pictóricos para un fácil acceso a la información por parte del usuario. Un sistema de este tipo puede incorporarse como parte de un sistema más complejo con aplicabilidad en muchas áreas.

1.1.1 Esquema del sistema de información espacial

Hasta la fecha no existe una norma general que nos permita definir una estructura de los sistemas de información espacial; sin embargo, realizando un análisis de los diferentes sistemas existentes es posible converger a un esquema en donde la parte más importante es todo un gestor de información gráfica, como se puede observar en la figura 1-2.

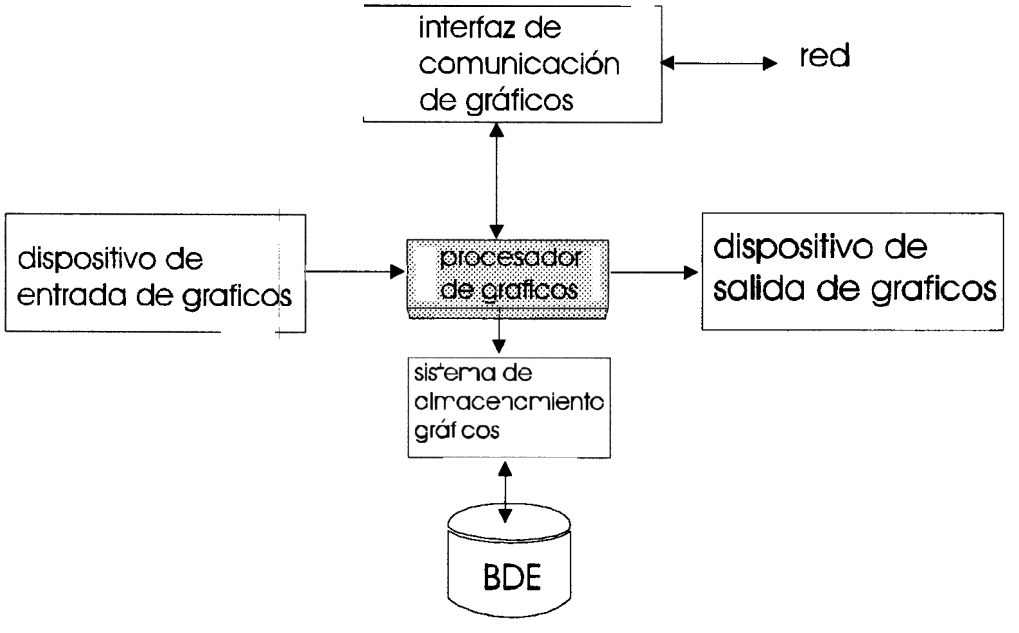


Figura 1-2 Diagrama del Sistema de Información espacial

a) Dispositivos de entrada. Existe en el mercado scanners de alta y baja resolución, cámaras de vídeo y digitalizadores, entre otros. Todos disponibles para minis y microcomputadores.

b) Procesadores de gráficas. Son microprocesadores de propósito especifico que trabajan como esclavos en minis y microcomputadoras. Las funciones de procesamiento que incluyen son: procesos de punto por punto, despliegue espacial, acercamiento, rotación, vectores de punto flotante, operaciones sobre matrices, aumento y alisamiento.

c) Dispositivo de salida para gráficas. Monitores de alta resolución para el despliegue de gráficas a color con una resolución de hasta 1024 por 1024 puntos y múltiples colores, que ofrecen alta calidad en la salida. Impresoras láser y plotters de alta resolución, que puede usarse para generar múltiples copias.

d) Dispositivo de almacenamiento. Actualmente los discos ópticos (CD) se ofrecen como medios de almacenamiento para grandes volúmenes de información. Hay también interés en usar micro fichas para el almacenaje y recuperación de gran cantidad de gráficos. Estas técnicas, por sus características son aplicables a un sistema de base de datos espacial.

e) Redes de comunicación en computadoras. Los adelantos recientes en interfaces que permiten la transmisión de datos de diversos tipos (voz,gráficas,texto) usando las propiedades actuales de los manejadores de redes, proveen los medios para interconectar muchas estaciones de trabajo. Cada una de estas resolviendo una tarea y al final conjuntándolas para resolver un problema muy complejo.

Los sistemas de información espacial tienen una serie de necesidades las cuales algunas de ellas suelen ser mas apremiantes de acurdo al sistema que se tiene. En la tabla 1-1 se resumen las necesidades contra los sistemas marcando con una "x". Los puntos mas importantes para cada área de aplicación.

Tabla 1-1 Necesidades de un sistema de información espacial

NECESIDAD / SISTEMA	interfaz flexible de usuarios	capacidad transmición de imagenes	herramienta de diseño para varios aplicaciones
5ta generación	x		x
OI	x	x	
CAD			x
IU			x
CIM	x	x	
GIS	x	x	x

Debido al incremento en el número de aplicaciones: automatización de oficinas, CAD interactivo, robótica, procesamiento de datos geográficos, procesamiento de datos médicos, monitoreo remoto de los recursos de la tierra y aplicaciones en la cartografía, la complejidad que requiere el manejo de la información espacial requiere de nuevas capacidades que conformen adecuadamente y en buen nivel los sistemas de infromación espacial.

1.1.2 Requerimientos para un Sistema de Información Espacial.

Antes de mencionar los requerimientos de un sistema de información espacial, primero listamos las funciones ideales que deben incluir los sistemas de este tipo [JOSE94].

a) Adquisición. Significa la captura o digitalización de imágenes, que originalmente se pueden encontrar sobre papel, película o son captadas a través de una cámara.

b) Edición. Medio por el cual se modifican, crea o elimina el contenido de una digitalización.

c) Procesado. Es la aplicación de técnicas para el aumento, disminución, detección de bordes, análisis de textura, y segmentación sobre una imagen. Adicionalmente existen técnicas de reconocimiento de patrones, así como también algoritmos y medios para la transferencia de información digitalizada.

d) Almacenamiento. El formateo, codificación , decodificación, definición de estructuras de datos y modos de indexación de imágenes para el almacenamiento en un medio dado.

e) Recuperación. La recuperación de una imagen desde una base de datos pictórica por indexación o por medios más flexibles de recuperación usando técnicas similares o algún tipo de lenguaje de consulta.

f) Desplegado. Modo en que las imágenes son desplegadas, junto con formas de obtener múltiples copias.

g) Comunicación. Modo en que una imagen es transmitida de una estación de trabajo a otra.

Dadas las funciones ideales de un SIE, entonces podemos formar una tabla (tabla 1-2) que según la aplicación . Se tenga una valoración de los distintos requerimientos que se necesitan.

Tabla 1-2 Requerimientos de los sistemas de información pictorial.

requeri-miento / sistema	adquisi-ción de gráficos	edición de gráficos	procesa-miento de gráficos	almacena-miento de gráficos	recupera-ción de gráficos	despliegue de gráficos	comunica-ción de gráficos
5ta ge-neración	en algunos casos		poco	si	si	si	en algunos casos
OI	en algunos casos	en algunos casos	poco	si	si	si	si
CAD	en algunos casos	si		si	si	si	
IU	si		si	en algunos casos	en algunos casos	si	
CIM	en algunos casos		poco	si	si	si	si
GIS	si	si	si	si	si	si	si

Generalmente en un sistema de información espacial, la información (de tipo espacial) es almacenada, procesada y analizada. De la figura 1-1, sabemos que la BDE es el núcleo de varios tipos de sistemas de información. Tales sistemas comprenden, entre otros, a los Sistemas de Información Geográfica (GIS's), a los Sistemas de Diseño y Manufactura Asistidos por computadora (CAD/CAM). A pesar de su creciente difusión, en la mayoría de las bases de datos se requiere conjuntar información nominal y espacial, la representación de esta última aparece forzada porque este tipo de bases de datos fueron desarrolladas originalmente usando sistemas manejadores sin soporte para datos espaciales, o bien se hicieron usando sistemas propios para el procesamiento de imágenes o gráficas por computadoras. En la siguiente sección describiremos los conceptos relacionados con las bases de datos espaciales.

1.2 CONCEPTOS DE LA BASE DE DATOS ESPACIAL

Mientras, las BD tratan con el procesamiento de datos tradicional solamente, las BDE incorporan funciones para el manejo de datos gráficos, obteniendo por consiguiente una base de datos que gestiona información nominal y gráfica. Las BDE fueron introducidas aproximadamente desde hace más de una década siendo Shi-Kuo Chang [Chan90] el que proporcionó una clasificación en base a los objetos y operaciones. En seguida nosotros damos la clasificación de los tipos de bases de datos espaciales.

1.2.1 Tipos de bases de datos espaciales

En base a los objetos y las operaciones sobre los mismos los BDE se clasifican como sigue

1) Aquellos que manejan una gran cantidad de figuras empleadas para diversos fines. En estos sistemas, los datos no siempre se guardan en memoria central (on-line), sino que se utilizan cintas debido a que con frecuencia estos son demasiado grandes. Un ejemplo típico son las imágenes terrestres captadas de manera remota por satélites.

2) Bases de datos que recuperan características de la información. Este tipo de base de datos no contienen datos de imágenes en 2D. Por lo tanto, se pueden emplear DBMS´s comerciales para procesar datos de este tipo. Ejemplo de estos son: datos de la adquisición, altura, localidad y cualidad.

3) Los manejadores de datos de imágenes y mapas orientados al procesamiento espacial. Incluye funciones como sobrecarga (overlaying) y coloreado.

4) Los bases de datos para el manejo de información estructurada, descrita a través de figuras, escenas o elementos gráficos primitivos. Este tipo se usa en la implementación de sistemas CAD.

5) Trabaja con una colección estándar de imágenes para el estudio experimental de algoritmos.

1.2.2 Manejador de la base de datos espacial

De la sección anterior se tiene que las operaciones y los objetos de una BDE determinan una clasificación la cual es independiente de una arquitectura común que toma en cuenta una componente de base de datos pictórica y otra alfanumérica que se intergran para proporcionar una interfaz uniforme. Sin embargo, las representaciones, las gestiones y los procesamientos de la información deben ser tratada de acuerdo con su naturaleza.

Primero, una **imagen** no puede representarse en código alfanumérico por lo que el usuario, para consultar y recuperar los objetos pictóricos y la relación espacial entre ellos, debe usar su forma analógica usando los dispositivos apropiados. De esta forma se logra que los usuarios manejen directamente la base de datos espacial.

Segundo, el **usuario** consulta y busca un objeto espacial a partir de las relaciones existentes entre un conjunto de dichas relaciones.

Tercero, aunque el **lenguaje de manipulación** es capaz de especificar el procesamiento pictórico y alfanumérico, la sintaxis no se puede mezclar entre si ya que son de naturaleza distinta. Por ejemplo, la sintaxis del lenguaje gráfico no puede traducirse al lenguaje de manipulación de relaciones.

Cuarto, **el procesamiento de imágenes** necesita un procesador de propósito específico dependiendo de los requerimientos de software y hardware.

Estas premisas sugieren una arquitectura de un sistema que pueda soportar una base de datos, desde muy sencillas hasta muy complejas.

Aunque el procesador alfanumérico y el pictórico son distintos, es necesario que exista un intercambio datos entre ellos. Esto lo podemos apreciar en la figura 1-3.

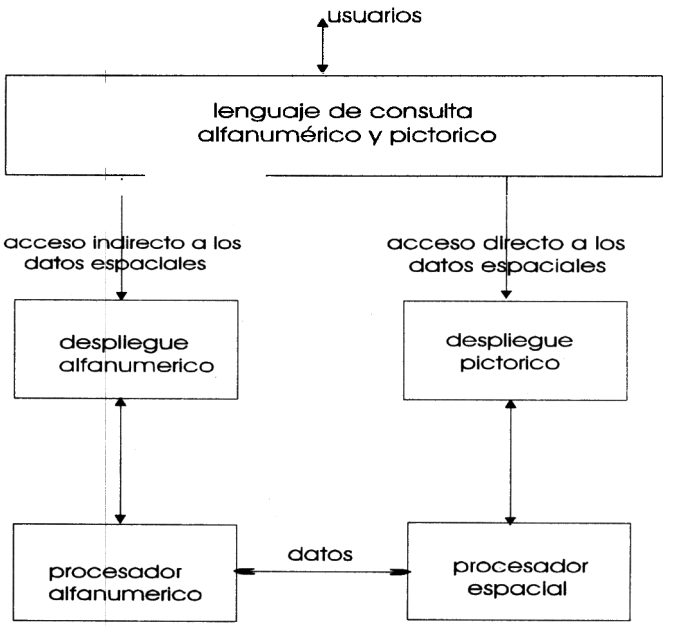


Figura 1-3. El manejador de la base de datos espacial

Desde el punto de vista del usuario, los requerimientos del manejador de una base de datos espacial son:

(1) Deber ser capaz de dar soporte directo para la manipulación de objetos espaciales y alfanuméricos.

(2) Incluir opciones de búsqueda espacial en alguna área determinada. Por ejemplo, "buscar todos los puentes del río 'x'".

(3) Deber poseer una interfaz avanzada de usuario para la especificación de la entrada y salida de gráficos.

Por las características de la información almacenada en una base de datos espacial, existen un conjunto de operaciones de manipulación y consulta que se acoplan a las necesidades de manejo de este tipo de sistemas. A continuación, enumeramos las más importantes.

Operaciones topológicas. Están relacionadas con aspectos topológicos tales como *adyacencia* ***dentro de, conectividad***, y otros. Ejemplos de estas operaciones las tenemos al preguntar: *dentro de una* área X, las cules son las entidades ***adyacentes*** a la entidad Y?

Operaciones geométrica. Estas se basan en la noción de distancia entre dos entidades (normalmente se usa la distancia Euclidiana). Por ejemplo, ***cerca de, lejos de,*** y ***sobre de***.

Operaciones de orientación: ***izquierda de, derecha de, arriba de, abajo de,*** entre otras.

Por las características que debe poseer un sistema que soporte al manejo de información espacial, es necesario trabajar internamente con estructuras de datos orientadas precisamente para la representación de este tipo de información.

1.3 CONCEPTOS DE VISUALIZACIÓN

Existe un dicho popular que dice: "una imagen dice más que mil palabras", si tomamos en cuenta esta afirmación es posible representar un gran conjunto de datos usando una buena imagen. La visualización es una tecnología que se utiliza para comprender rápidamente, y recordar mejor. Por ejemplo, el diccionario DUDEN es famoso en el mundo debido a que usa figuras para explicar las palabras. Las ideas así representadas tratan sobre asuntos cotidianos y se usan de manera exhaustiva, lo que se propone actualmente es incorporar éste tipo de elementos dentro de la computación como medios para definir procesos y datos. Esto ha creado una nueva área denominada visualización.

1.3.1 Definiciones

Primero daremos un serie de definiciones de los términos relacionados con el área.

Definición 1-1 Visualización

El estudio del mecanismo existente entre una computadora y el hombre, los cuales están relacionados para entender, usar y comunicarse usando información visual [MCCO87].

Definición 1-2 Ambiente visual de programación(VPE)

Es un ambiente para la definición de procesos (programas) en el que intervienen elementos visuales con la finalidad de facilitar las tareas del usuario.

Definición 1-3 Programación visual

Es un proceso que incluye elementos gráficos para la definición de un programa.

Definición 1-4 VPL (Visual Programming Language)

Hablamos de un VPL cuando la sintaxis de un lenguaje de programación incluye expresiones visuales como diagramas, iconos y manipulación gráfica.

Examinando los trabajos recientes podemos decir que los progresos de la programación visual se desarrollaran principalmente en tres direcciones. Como se ve en la figura 1-4.

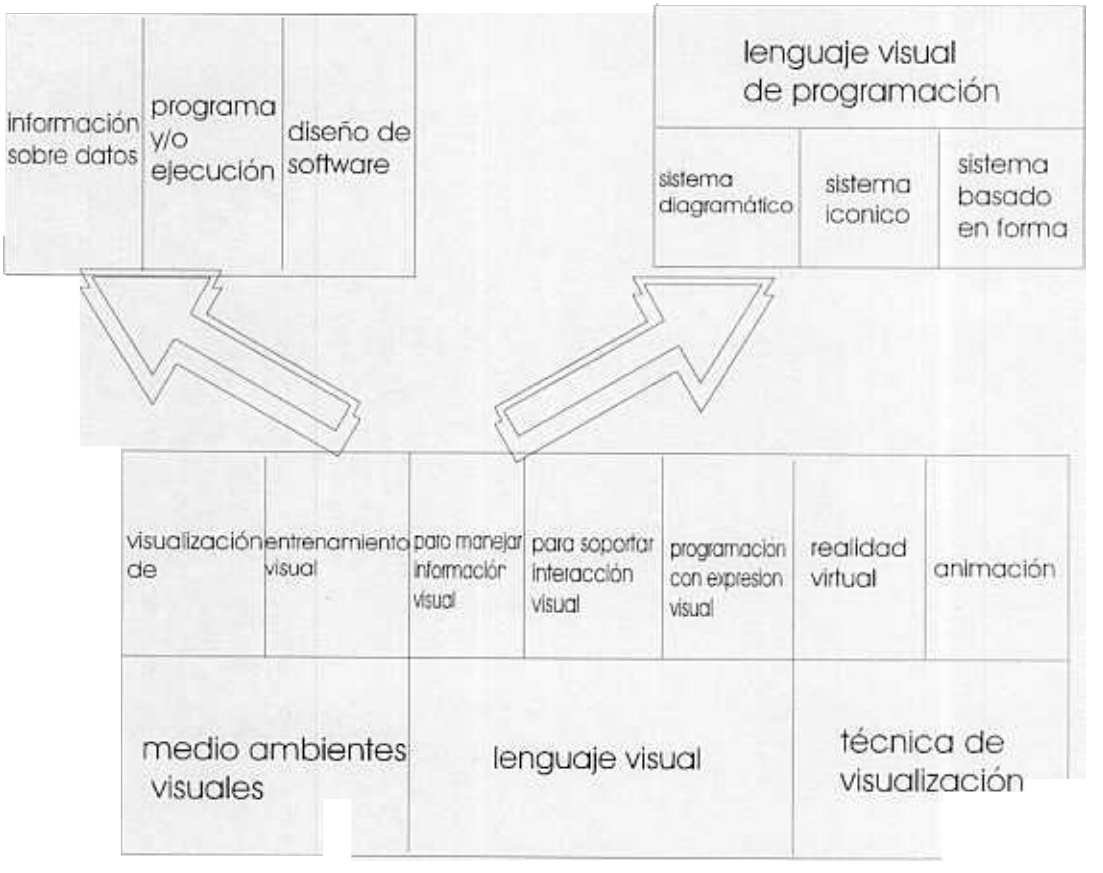


Figura 1-4 Clasificación de la Programación Visual.

El primer enfoque se refiere a los medios ambientes visuales con investigación en las siguientes áreas:

*Desarrollar medio ambientes visuales para la construcción de programas.

*La recuperación y representación de la información.

*Diseñar y mejorar la comprensión del Software.

La otra dirección es la que corresponde a los lenguajes visuales que son diseñados para:

*Manejar información visual.

*El soporte de interacción visual.

*Programar usando expresiones visuales.

1.3.2 Las formas de la visualización

Gerald L. Lolise [GERA94] hizo una buena clasificación sobre las formas y métodos de representación visual, seleccionó sesenta ejemplos de acuerdo a sesenta tipos de representaciones visuales para clasificarlos en ocho categorías. A continuación se describe las categorías de concepto utilizado para cada tipo de representación .

(1) Gráficas Codifica la información en términos de la posición y la magnitud de los objetos geométricos. Los datos numéricos en una, dos o tres dimensiones se mapean a un sistema de coordenadas cartesianas o polares.

(2) Tablas Son los arreglos de palabras, números, signos o la combinación de ellos que exhiben un conjunto de hechos o relaciones en un formato compacto. Las tablas tienen menos abstracción simbólica que las gráficas.

(3) Diagrama de tiempo Muestra datos en función del tiempo. Se distinguen de las tablas por el énfasis puesto en la dependencia del tiempo.

(4) Diagramas de red Muestra relaciones entre componentes, los cuales indican la presencia o ausencia mediante símbolos. Las relaciones entre componentes ya sea proximidad, similitud o contención se muestran con líneas y flechas.

Hay dos tipos de diagramas, los dos expresan datos espaciales:

* **Diagramas de Estructura** Son las descripciones estáticas del objeto físico. El dato espacial expresa las coordenadas dimensionales exactas del objeto.

* **Diagramas de Procesos** Describe las interrelaciones y los procesos asociados con los objetos físicos. Los datos espaciales expresan relaciones dinámicas continuas o temporales entre los objetos en un diagrama de procesos.

(5) Mapas Son representaciones simbólicas de geográficas física.

Los mapas describen localidades geográficas con características particulares, usando símbolos o letras. Por ejemplo: los diagramas marinos, mapas de carreteras, mapas topográficos y varias proyecciones de mapas del mundo. La cartografía difiere de los mapas ya que esta sobrepone datos cuantitativos sobre un mapa base.

(6) Cartografía Son mapas espaciales que muestran datos cuantitativos. Por ejemplo: chloropleth, isopleths, mapas de punto y de flujo.

* **Chloropleth** Usa color, escala de grises o textura para áreas de código de igual valor.

* **Isopleths** Usa líneas para unir puntos con la misma cantidad o valor. Por ejemplo, el contorno de los mapas.

* **Los mapas de punto** Usan símbolos para mostrar la localización de puntos individuales en un mapa.

* **Los mapas de flujo** Muestran las direcciones del movimiento de acuerdo al número, al ancho y a la dirección de líneas y flechas.

(7) Iconos Es una imagen, representación, ilustración, grabado o esquema que representa un concepto, idea, dato u operación.

Actualmente, la inclusión de iconos en ambientes de trabajo ha resurgido rápidamente, el uso de estos elementos para transmitir información tiene su origen desde hace muchos siglos; por ejemplo, mientras los egipcios los utilizaron en su escritura denominada jeroglífica, los aztecas y los mayas "escribieron" sus códices con elementos gráficos dentro de un contexto plástico. Una de las razones por lo que el uso de iconos ha resurgido rápidamente, se debe a la facilidad por parte del hombre para captar un mensaje basado en ellos, ya que la mente humana, cuando procesa imágenes, infiere relaciones sin necesidad de incluir texto en estas. Además, existen otras razones que nos invitan a iniciar la utilización de elementos visuales dentro de los medios ambientes de trabajo y desarrollo:

i) Las figuras son más poderosas que las palabras como un medio de comunicación. Pueden transmitir mas significados de una manera más concisa por unidad de expresión.

ii) Las figuras ayudan a entender y recordar.

iii) Las figuras pueden ser un incentivo para aprender a programar.

iv) Las figuras no tienen las barreras del lenguaje. Cuando son adecuadamente diseñadas, son entendidas independientemente del idioma que se hable.

Con lo anterior no queremos decir que la meta de los lenguajes visuales sea representar todo tipo de ideas y acciones mediante iconos sin incluir en estos texto; la finalidad es usar de manera armónica los dos tipos de representación para integrar ideas más claras.

(8) Gráficos de realidad virtual Son imágenes reales de un objeto o escena. Todas las fotografías, gráficos de entidades reales e imágenes forman un solo grupo, estas representaciones tienen una correspondencia uno a uno entre el mundo real y la imagen. Las propiedades entre intervalos y distancias del espacio entre objetos del mundo real son preservados en la imagen. Las imágenes tienen un tamaño muy fino determinado por la resolución de la misma. Cuando la imagen es cambiada de tamaño, hasta ser muy pequeña, los detalles de los objetos son más difíciles de ver y pierden precisión.

Finalmente, las imágenes tienen un tamaño definido y una forma que limita la cantidad de imágenes que pueden ser vistas al mismo tiempo.

1.4 LA VISUALIZACIÓN EN LA ACTUALIDAD

Podemos decir que el tema de visualización dentro de la computación es de actualidad. Con lo que se refiere a la novedad y actualidad del tema, es conveniente mencionar que una de las revistas del área es el " Journal of Visual Languages and Computing" el cual es de reciente aparición (1990). En algunos números han aparecido algunas propuestas de lenguajes visuales que tienen su aplicación a base de datos espaciales. Entre ellos se tiene: Cigales (Cartographical Interface Generating and Adapted Language for Extensible Systems) [**Calc 94**]. PQBE (Pictorial Query-By_Example). Visual Programming [**Shu 92**].

1.4.1 Programación Visual

La programación visual distingue entre lo que se describe y el como se describe. En los terminos que es lo que se describe son los programas que son expresados en terminos de la sintaxis y la semántica de los lenguajes de programación. Cuando al menos alguno de los elementos terminales de la gramática del lenguaje son gráficos, pictogramas o formas, decimos que los lenguajes tienen una sintaxis visual.

Una sintaxis visual puede incorporar información espacial relativa a la conectividad, tomando en cuenta además los atributos visuales tales como la localización y el color.

Ejemplos de lenguajes con sintaxis visual incluyen muchas clases de los lenguajes diagramáticos, que pueden ser los lenguajes de flujos de datos (el lenguaje LIDA [Chap 91]). Otros ejemplos son los iconicos; en ellos la programación se lleva a cabo mediante la composición de iconos que en esta tesis se amplio con el concepto de tarjeta (ver el capitulo 5).

1.4.2 Ambientes visuales de programación

La forma en como el programador trabaja para crear, modificar y examinar programas. La determina el ambiente de programación. Dichos ambientes consisten de un conjunto de herramientas y una interfaz de usuario para acceder las herramientas. Decimos que el sistema tiene un ambiente visual cuando las herramientas son gráficas, usa tecnicas gráficas para la manipulación de elementos pictográficos y para el desplegado de la estructura del programa.

Como ya se mencionó, el desarrollo intenso de la tecnologia en graficación ha permitido el establecimiento pleno y su comercialización de diversos ambientos visuales, como puede observarse en una sinopsis según la tabla 1-3.

Tabla 1-3 Algunos ejemplos de software desarrollados bajo un sistema de programación visual

Sistema	compañia	uso
Visual Basic.	Microsoft Corp	Prototipos de nuevos productos de software, base de datos de objetivo específico
OpenStep	Nextstep	Sistema operativo visual
VEE	Hewleh Packard	Herramienta visual para la adquisición y análisis de datos en ingenieria
AVS	Deift university of technology, Dept. of technical information (the Netheherlands)	Desarrollo nuevas técnicos de visualización
Lab-View	National Instruments Tektronix	Software para control de instrumentos
Visual Works	ParcPlaceSmalltalk	Ambiente Visual de Programación

Lab-View es un ambiente basado en gráficas para desarrollar, depurar y correr programas. El sistema es usado para crear y correr archivos de documentos Lab-View denominados instrumentos virtuales. En Lab-View el programador trabaja en los diagramas de ventanas, desarrollando la estructura de sobrelapado del programa mediante el uso de iconos interconstruidos. Muchos de los iconos son dinámicos y pueden ser expandidos para acomodar mas entradas o salidas, asi como, el redimencionamiento para proveer mas área para otros iconos y ligas.

VEE (Visual Engineering Environment) es una herramienta visual de programación para probar adquisición y análisis de datos de ingenieria. En VEE se crean diagramas de flujo de datos o modelos mediante la selección y colocación de objetos gráficos conectandolos con lineas o trayectorias de información. VEE es similar Lab-View en donde el desarrollador y constructor observa a través de un programa dibujado y un paquete de CAD. En VEE se tiene una vista de detalle y una vista de panel.

Open Step es todo un ambiente visual que puede correr actualmente sobre diversos sistemas operativos: MACH,SOLARIS y WINDOW NT. En Open Step las aplicaciones se crean fuera de los objetos existentes mediante la tecnica de alambrado. La lineas que conectan dos objetos, A y B, son etiquetadas por la selección de un mensaje legal de un menú de mensajes.

Visual Works es un ambiente de programación que incorpora una herramienta de pintado, en donde los objetos visuales y las decoraciones visuales son seleccionadas de una paleta y colocadas sobre un canvas. Los elementos en los los canvas pueden ser nombrados y estabonados uno a otro para subreponer objetos de aplicación. El acto de construir el canvas es para especificar los elementos de la pantalla. Visual Work y Open Step son ejemplos de ambiente de programación de multicapa. Sobre una es un ambiente visual con un lenguaje de programación de texto (Smalltalk y C objetivo, repectivamente). El ambiente visual consiste de browsers, explicadores, inspectores y trazadores visuales para la depuración. Un conjunto espacial de asistencia en la creación de ciertas clases de aplicaciones, que algunas son incorporadas a los GUI.

1.4.3 Lenguaje Visual

Podemos dividir a los lenguajes visuales en tres categorías:

(1) Lenguajes para manejar información visual. Son diseñados principalmente para el procesamiento de imágenes. Estos sistemas están motivados por la necesidad de tener lenguajes fáciles de usar para la manipulación y consulta de datos pictóricos. Un ejemplo de este tipo de sistema son los GIS. Se describe detalladamente en el capítulo 5,.

(2) Lenguajes que soportan interacción visual. Son diseñados para definir, crear y manipular símbolos gráficos soportando una representación e interacción visual, aunque los lenguajes mismos son textuales. Un ejemplo clásico de este tipo de sistemas es Windows de MicroSoft.

(3) Lenguajes visuales de programación con expresiones visuales. La idea central de este tipo d languaje es permitir a los usuarios programar mediante iconos.

Un ejemplo de la última categoría es LIDA[CHAPA91], que tiene símbolos gráficos que representan operaciones sobre datos. Los datos fluyen a través de líneas de flujo para entrar a módulos de transformación que son los iconos, los cuales tienen asociada una semántica y una imagen. Existen un conjunto de reglas sintácticas que disponen un conjunto de iconos relacionados en forma de una carta gráfica, o como se define en el proyecto LIDA, un Flujograma.

Las principales características de este sistema son:

* Se basa en el modelo de datos relacional
* Es un lenguaje de flujo de datos
* Su programación es visual

En términos generales, LIDA trabaja con relaciones, recibiéndolas como entrada y produciéndolas como salida.

El modelo de flujo de datos que tiene LIDA da un enfoque de abstracción procedural debido a que el usuario sólo trata con iconos que representan transformaciones de relaciones, además de que el flujo de datos determina implícitamente el control del flujo. Los iconos admiten argumentos que pueden ser: atributos, expresiones aritméticas o expresiones booleanas. Estas últimas se utilizarán para determinar bifurcaciones en el flujo de datos mostrando una vista en paralelo.

En la figura 1-5, se puede observar un flujograma creado bajo LIDA para resolver una consulta sobre el siguiente problema.:

Se tiene una base de datos la cual esta constituida de tres esquemas de relación: PROVEEDOR, PARTE y ORDEN:

esquema de relación proveedor:

prove(#s,snomb,edo,ciudad).

esquema de relación parte:

parte(#p,pnomb,color,peso,ciudad).

esquema de relación orden:

orden(#s,#p,cantidad).

Una consulta: **Obtenga los nombres de aquellos proveedores que suministran la parte p2**.

Una consulta a la base de datos por medio de una flujograma en LIDA como se muestra en la siguiente figura:

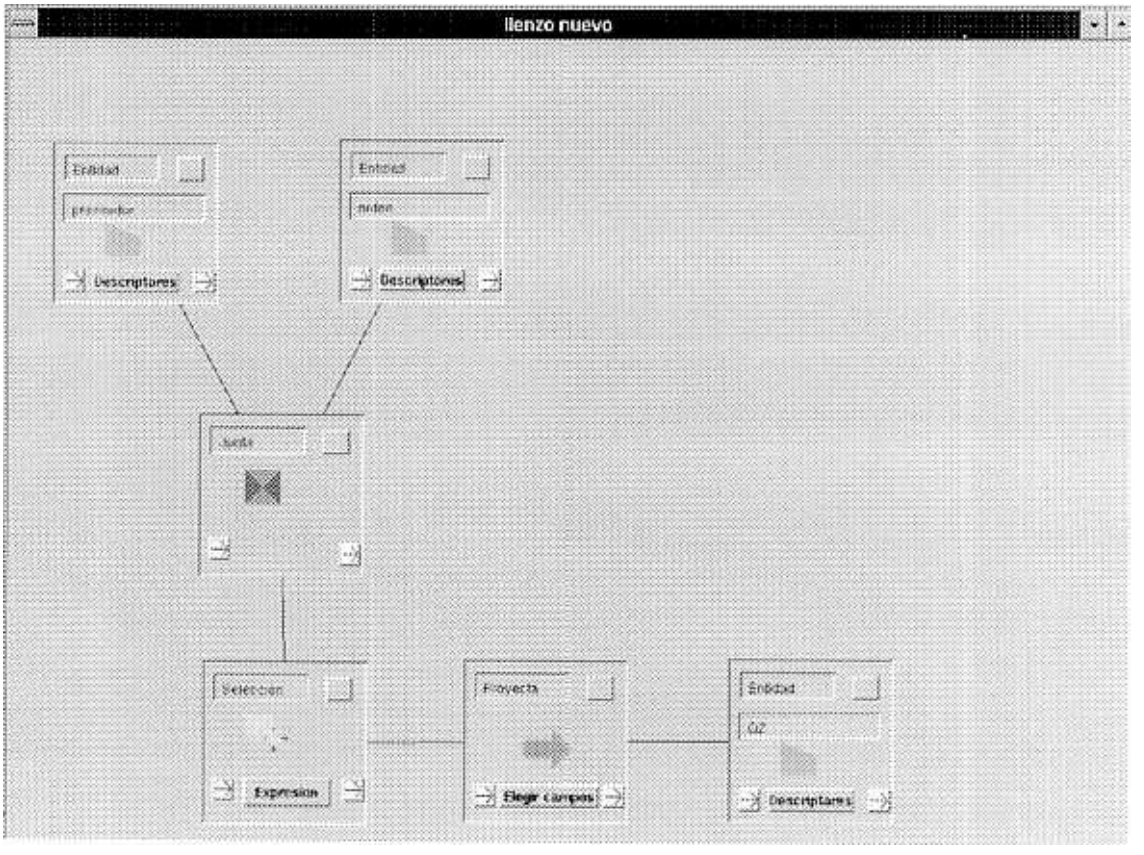


Figura.1-5 Consulta a la base de datos por medio de un flujograma usando LIDA

La visualización en el Diseño de Software consiste en proporcionar ambientes gráficos para obtener facilidades en diversos aspectos del desarrollo de software. La visualización se puede realizar desde los diferentes aspectos del desarrollo: los requerimientos, las especificaciones, las decisiones de diseño y las estructuras del sistema.

Dentro de estos sistemas, el usuario no necesita visualizar mentalmente los efectos de sus instrucciones mientras construye un programa, por ejemplo: HEVICOP[SIERRA95] (HErramienta VIsual para la COnstrucción de Programas). En términos generales, este sistema nos permite construir gráficamente esqueletos de control de un programa, para posteriormente generar el código en lenguaje C asociado al esqueleto.

Los programas dentro de éste, se definen usando un modelo de representación propuesto por E. P. Glinert llamado BLOX[GLIN87].

Dentro de BLOX cualquier objeto estructurado puede ser representado jerárquicamente usando un conjunto de diagramas bidimensionales interconectados a través de elementos especiales llamados 'gates'. Cada uno de estos diagramas esta compuesto por un conjunto de bloques con diferentes atributos como texto, color y restricción de ensamble (bordes y muescas). Esta última da pautas para controlar los aspectos sintácticos cuando el modelo se aplica a lenguajes de programación. HEVICOP, esta definido bajo los términos de un lenguaje visual, por lo cual esta compuesto de un Diccionario de Iconos, Un Diccionario de Operadores y una Base de Acciones Semánticas que interpreten las construcciones dentro del sistema.

En HEVICOP el DI esta formado por representaciones gráficas de las principales construcciones de control del lenguaje de programación 'C': while, do-while, for, switch, if_else, case, función, proposición simple, comentario, inicio de bloque y fin de bloque.

Cada elemento posee una entrada en un diccionario físico de información, el cual contiene los datos necesarios para poder dibujar el icono e interpretarlo. Algunas de estas representaciones se muestran en la siguiente figura1-6.

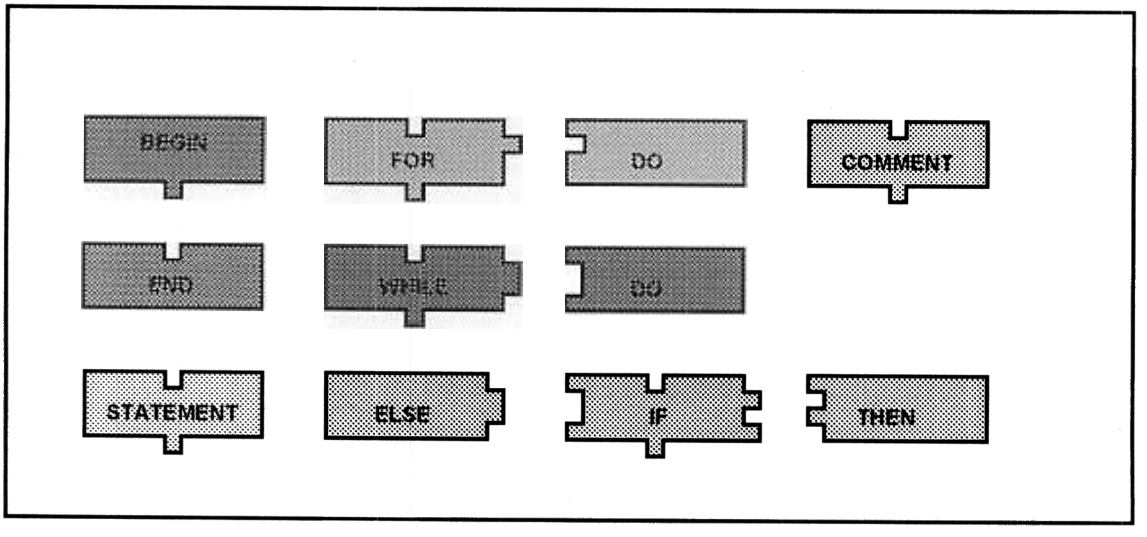


Figura 1-6 Algunos iconos creados al acoplar el paradigma BLOX a un lenguaje estructurado (Lenguaje C).

En seguida figura 1-7 se muestra una pantalla típica de una sesión de trabajo en HEVICOP.

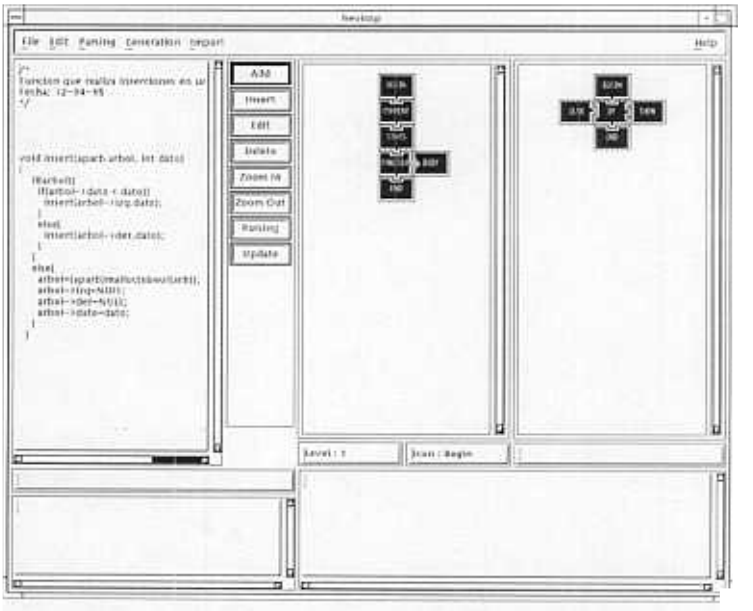


Figura 1-7 Pantalla después de construir varias funciones

1.5 FILOSOFÍA DEL SISTEMA VISDA

El objetivo de esta sección es presentar la filosofia y naturaleza del sistema VISDA, el cual se sustenta en una base de datos espacial y los lenguajes visuales. Los datos espaciales contienen conocimiento explicito e implicito. Mientras el primero, concierne a las entidades y objetos que se definen con los datos espaciales, que en muchos de los casos son objetos agregados en clases; el segundo refiere a las relaciones topológicas entre dichos objetos, considerando la posición relativa entre las diversas entidades que se extraen y almacenan.

El núcleo del sistema VISDA, que dá relieve a la filosofia del mismo, es su sistema visual de consulta. A través de él se pueden expresar los requerimientos relativos al dominio de interés con representaciones visuales. En este contexto podemos mencionar que es una de las principales contribuciones de este trabajo y que puede compararse con algunas otras que se reportan en el Journal Visual Languages and Computing de los últimos cinco años. Desde el punto de vista de visualización, usando el filosofia siguiente:

(1) Diseño de la base de datos usando la herramienta EVE.

(2) Definición de la representación visual de las estructuras de datos, archivos y atributos, así como de los algoritmos de visualización para desplegarlos.

(3) Visualización las operaciones: recuperar, salvar y manipular de la base de datos. Por ejemplo: el DML visual y el lenguaje visual de consulta.

(4) Visualización la información de la base de datos y de los resultados obtenidos en una operación.

CONCLUSIONES

La base de datos espacial es un núcleo de muchas aplicaciones. Una de las áreas de mayor interés en México por su importancia económica es la exploración petrolera. Además, el tema de visualización dentro de computación ha surgido rápidamente. Medios ambientes visuales y programación visual ha sido la razón principal para su amplio desarrollo. Se desarrollará el sistema VISDA, aprovechando las características de los LVs junto con las nuevas técnicas para el manejo de información espacial, cuyo usuario final no necesariamente tiene que estar familiarizado con el manejo y programación de una BDE.

CAPITULO 2

EL MODELADO CONCEPTUAL DE LA BASE DE DATOS ESPACIAL PARA EXPLORACIÓN PETROLERA

Durante mucho tiempo, el diseño de base de datos fue considerado más como un arte que como una ciencia. Sin embargo su desarrollo ha sido tal que actualmente se considera una disciplina con sus propios métodos y técnicas. El diseño de base de datos se hace normalmente en tres etapas: diseño conceptual, el cual se encarga de hacer una representación abstracta de la realidad; diseño lógico, el cual trasladará ésta representación a especificaciones que puedan ser implementadas y procesadas por un sistema computacional y el diseño fisico quién determinará la estructura física de almacenamiento y métodos requeridos para un acceso eficiente al contenido de una base de datos desde un dispositivo de almacenamiento.

El objetivo de este capítulo consiste en resolver un problema de diseño de la base de datos espacial. El propósito fue el diseño conceptual que inicia en un nivel alto iniciado en la especificación de los requerimientos que se presentan en un minimundo de aplicación petrolera.

Un modelo de datos es una colección de conceptos que pueden usarse para describir un conjunto de datos y operaciones para el manejo de los datos [ADOR 93]. El modelo de datos es una simplificación del modelo de empresa el cual representa una descripción de alto nivel de la naturaleza de la organización y sus métodos.

Mundo Real →Modelo Empresa→Modelo de datos→Base de datos

Los pasos básicos de ésta metodología son:

*1.- **Análisis de requerimientos con el modelo entidad-vínculo-extendido.*** El análisis de requerimientos se realiza en estrecha comunicación con el usuario final y consta básicamente de:

- Identificación y clasificación de entidades y atributos.
- Identificación de jerarquías de generalización y especialización.
- Definición de vínculos entre las entidades.
- Integración de diferentes vistas de entidades, atributos y vínculos.
- Creación del modelo de Empresa, y el diseño conceptual de la base de datos.

*2.- **Transformación del diagrama Entidad-Vínculo-Extendido al modelo relacional de Codd.*** Sobre la base de datos de una categorización de los constructores del modelo entidad-vínculo-extendido y un conjunto de reglas de mapeo, cada vínculo y sus entidades asociados, son transformados en un conjunto de relaciones de Codd eliminando las relaciones redundantes.

3. Normalización de relaciones: Para cada relación derivada del diagrama entidad-vínculo-extendido se analizan sus dependencias funcionales y para la normalización se aplica algún método de análisis o síntesis. Aquí se emplea el algoritmo de Bernstein.

4. Diseño físico de la base de datos: Finalmente la última etapa corresponde al diseño físico que es muy dependiente del manejador de la base de datos. El diseño físico es muy dependiente del manejador de base de datos donde se implementa el diseño (se describió en capítulo 3).

El diseño se llevo a cabo como un proyecto, el cual se reporta en [CHAP 94]. Se tomó como prototipo el diseño de la base de datos para la base de datos propuesta " exploración petrolera de Pemex". En el proyecto, se hizo un buen análisis del flujo de la información de Pemex y se realizó usando la herramienta EVE [Alday96], que nos permite obtener las bases para el diseño físico a partir de un esquema de entidad-vínculo-extentido, como se muestra los procesamientos en la figura 2-0:

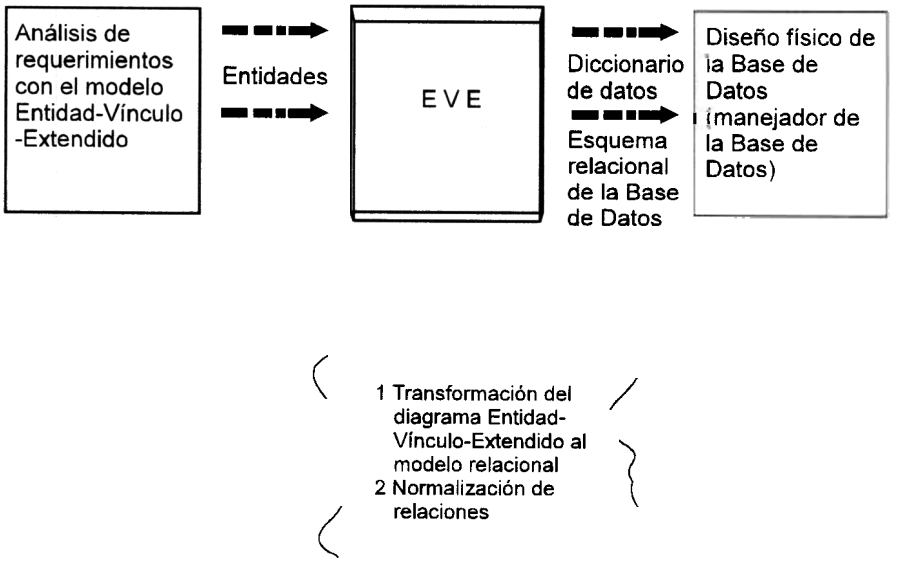


figura 2-0 diagrama del diseño de la base de datos

En seguida, se mencionará brevemente las partes importantes.

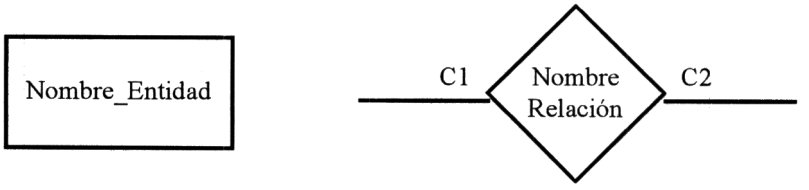
2.1 ESQUEMAS DE ENTIDAD-VÍNCULO

Un esquema de entidad-vínculo es la representación gráfica de la estructura de la información contenida en una base de datos. Una base de datos en este sentido, esta conformada por un conjunto de archivos físicos de datos. Los elementos importantes en un diagrama de este tipo son:

a) Entidades: representadas por un rectángulo.

b) Relaciones o vínculos: representadas por un rombo, etiquetado con indicadores de cardinalidad.

La representación gráfica que poseen se muestra en la figura siguiente (figura 2-



C1,C2: cardinalidad de la relación

Figura 2-1 Representaciones gráficas para en diagrama entidad-vínculo.

Los pasos que se siguen para el diseño de un esquema entidad-vínculo son:

1. Identificación de entidades.
2. Identificación de relaciones, y construcción de subesquemas.
3. Integración de subesquemas, para formar el esquema global.
4. Análisis de conflictos y resolución a los mismos.
5. Unión y análisis de redundancia.

Identificación de entidades dio como resultado la obtención de la tabla de entidades. cada entidad tiene los atributos de importancia en la solución del problema (se explica mas detalle en sección 2.2).

Identificación de relaciones, y construcción de subesquemas. a seguir es buscar las formas en que las entidades se pueden relacionar unas con otras, y la forma en que se pueden relacionar (cardinalidad 1 a 1, 1 a muchos, por ejemplo). En este sentido, algunas de las relaciones existentes en la información analizada se muestran a continuación (figura 2-2)

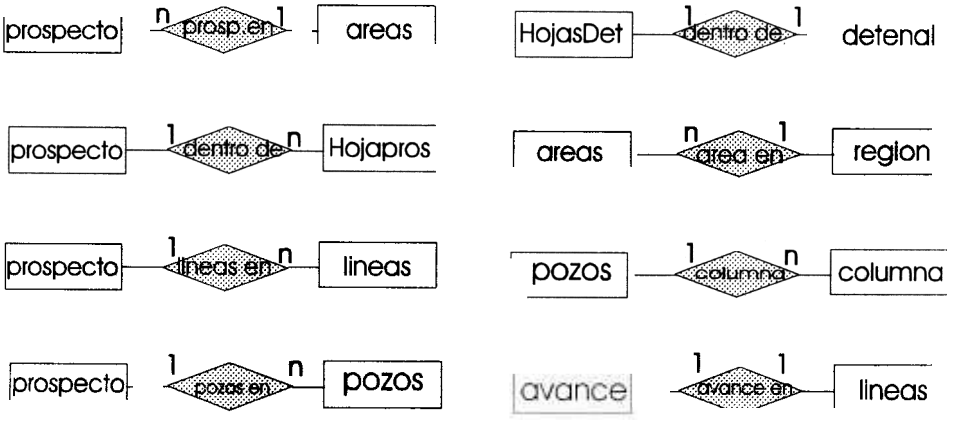


Figura 2-2 Ejemplos de relaciones binarias.

Esto da como resultado un conjunto de subesquemas, los cuales se van coleccionando para que al final se reúnan y se obtenga el esquema o subesquema global.

Integración de subesquemas, para formar el esquema global. es la culminación del esquema de datos, que detalla completamente un esquema de datos, usando consistentemente todos los elementos del modelo, con las descripciones de los datos y las relaciones existentes entre ellos.

Durante esta actividad se hace un análisis de todos los atributos y relaciones para verificar si las ocurrencias de alguna entidad pueden tener atributos no especificados o no pertenecen a alguna relación. Posteriormente se determinan los identificadores de las entidades en el esquema.

En este punto, el esquema de datos tiene una representación completa y detallada de todas las clases de objetos provistos por el modelo de E-V. Un ejemplo del esquema completo del ambiente interpretación, proceso y operación geofísica, se muestran en la figura 2-3.

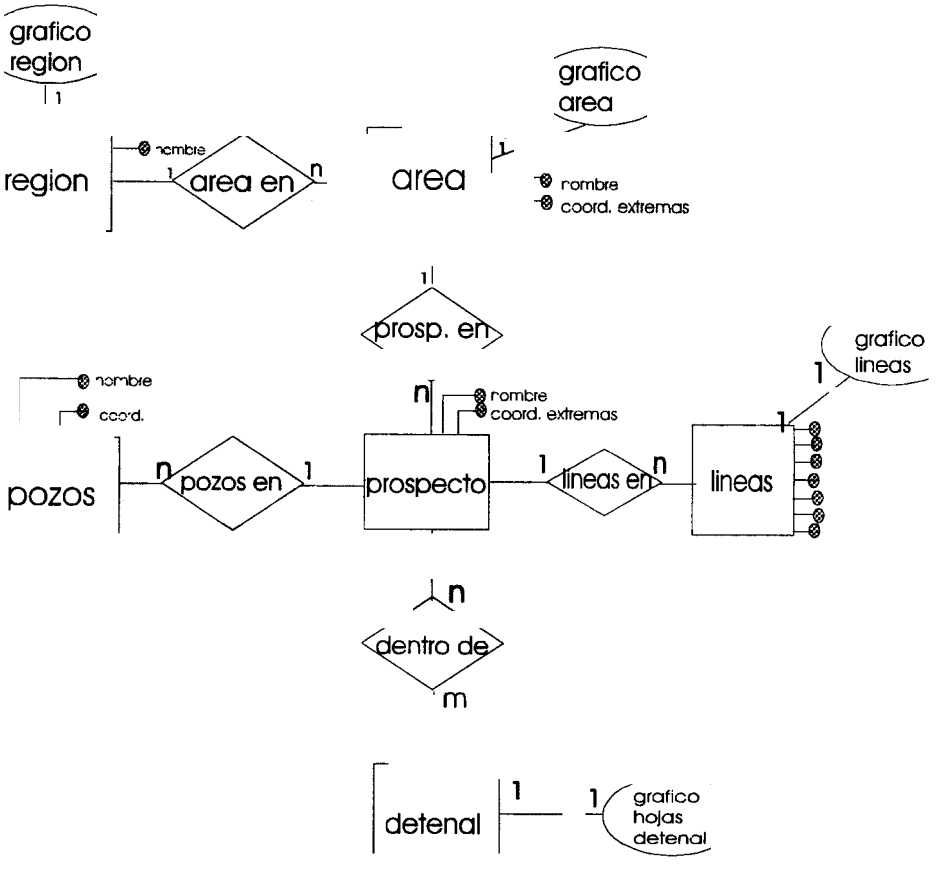


Figura 2-3 Esquema de operaciones: Operación Geofísica.

Análisis de conflictos y resolución a los mismos, se analiza dos tipos de conflicto:

- **Homónimos**.- cuando el mismo nombre se asigna a diferentes clases de objetos (entidades).
- **Sinónimos**.- Con diferentes nombres se asigna a la misma clase de objeto (entidad).

Estos problemas son resueltos generalmente utilizando el diccionario de datos como un repositorio de cambios hechos a los datos originales. Si no existen conflictos se procede al siguiente paso.

Unión y análisis de redundancia, los conflictos para la representación de objetos de todos los esquemas, ya han sido resueltos, por lo tanto, se puede llevar a cabo la unión mediante una simple superposición de conceptos comunes y propiedades interesquemas. El resultado de la unión es un esquema global como se muestra en la figura 2-4

Durante esta actividad, se analizan las redundancias en la representación de datos, para esto se deben considerar pares de trayectorias y las relaciones entre los conceptos terminales (entidad o relación) de las trayectorias que serán analizados para verificar si ellos son semánticamente equivalentes.

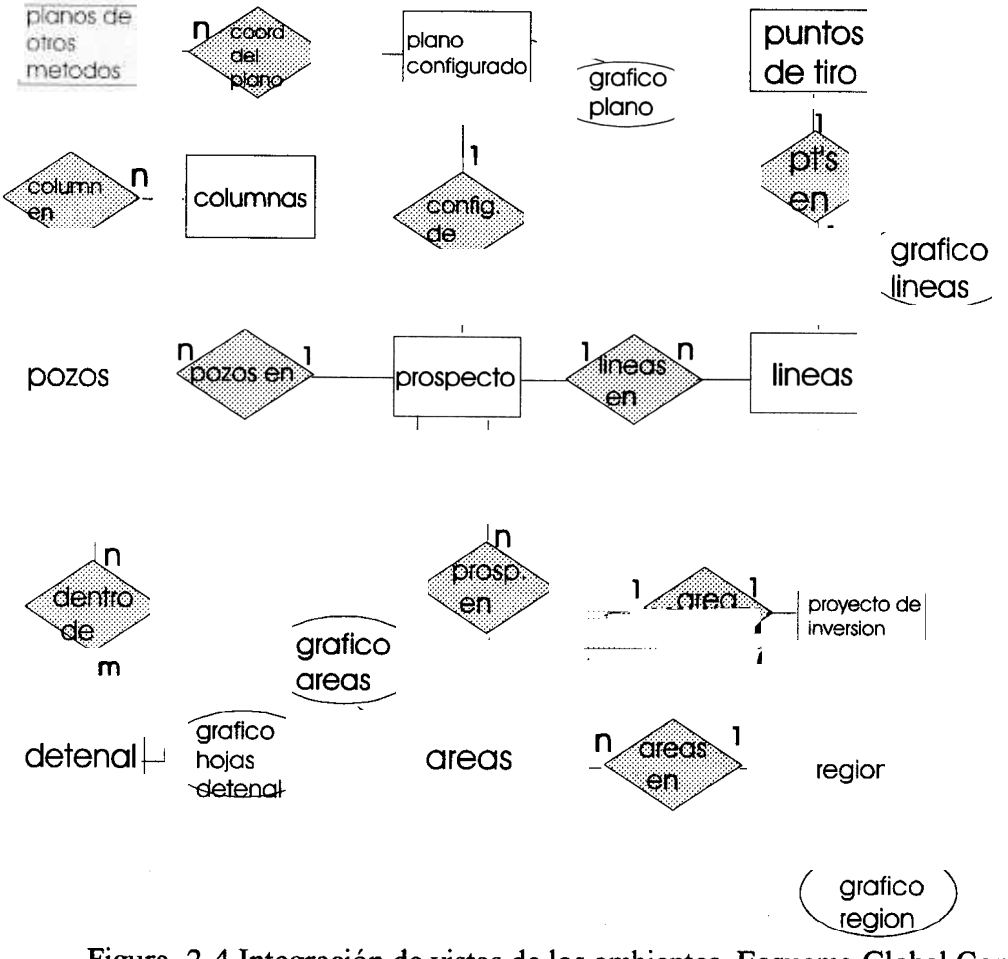
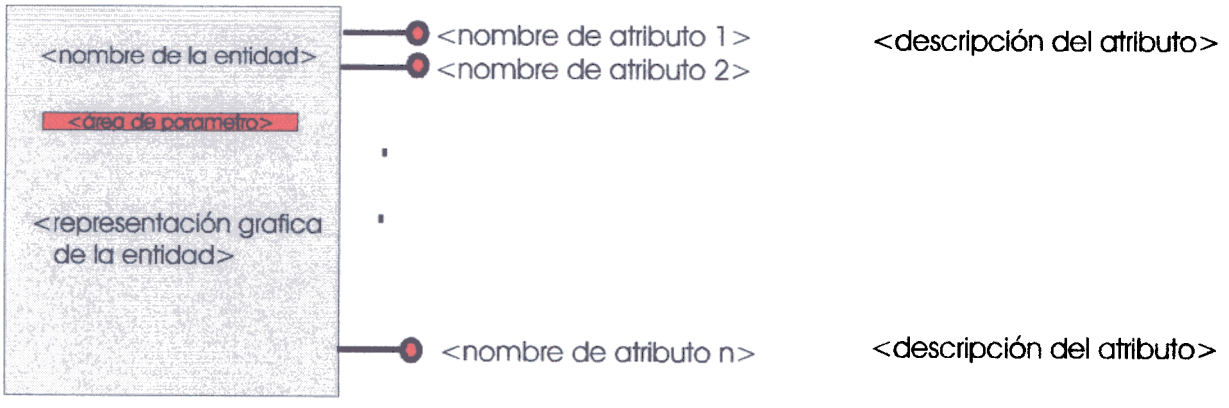


Figura. 2-4 Integración de vistas de los ambientes. Esquema Global Conceptual

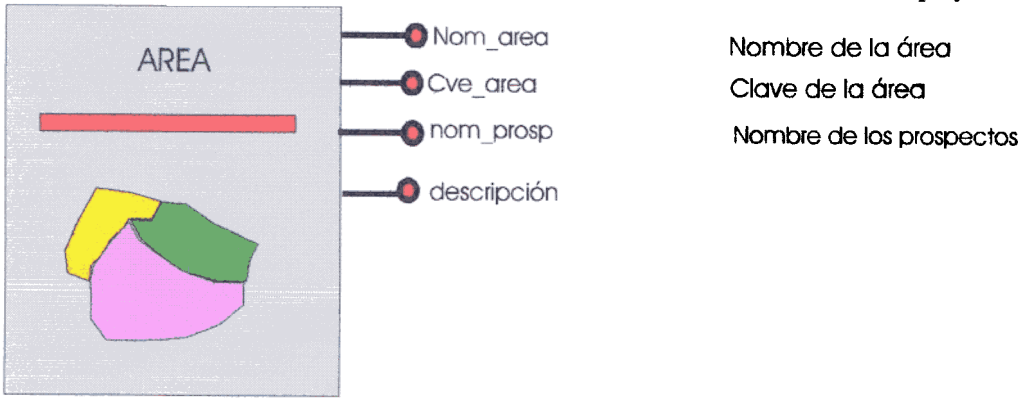
2.2 IDENTIFICACIÓN DE ENTIDADES.

A continuación se presenta una definición concreta de las entidades que se emplearon en el diseño de la base de datos como primero paso del diseño de un esquema entidad-vínculo. Se describe en nombre, presentación, atributos y descripción para cada entidad. Estas entidades les llamamos entidades conceptuales (capítulo 5). Para cada definición se usa el formato siguiente:

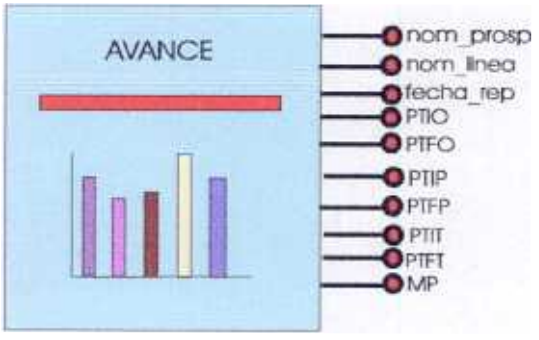
<nombre de la entidad>,
<descripción de la entidad>,



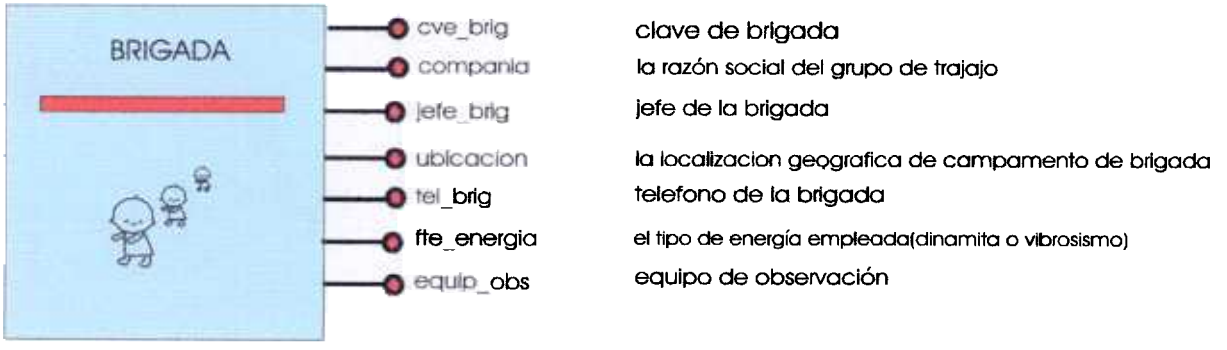
Ejemplos:
La entidad **ÁREA**:
representa la superficie terrestre o acuática en donde se desarrolla las entidades, los proyectos.



La entidad **AVANCE**:
Es una entidad que proporciona el nivel de avance en un prospecto determinado.

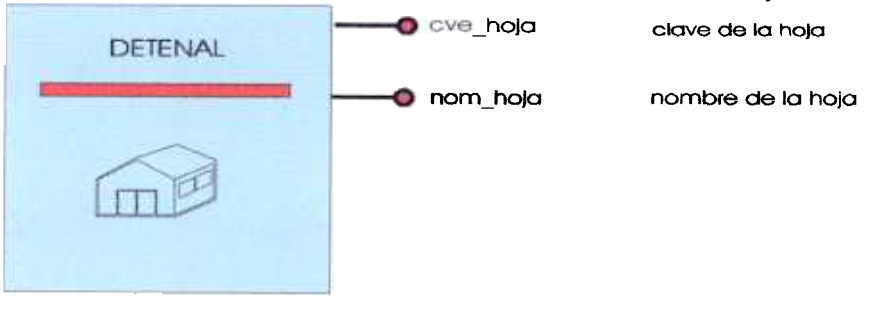


La entidad **BRIGADA**:
Es una entidad que ejecuta un trabajo de prospección petrolera dentro de un prospecto.

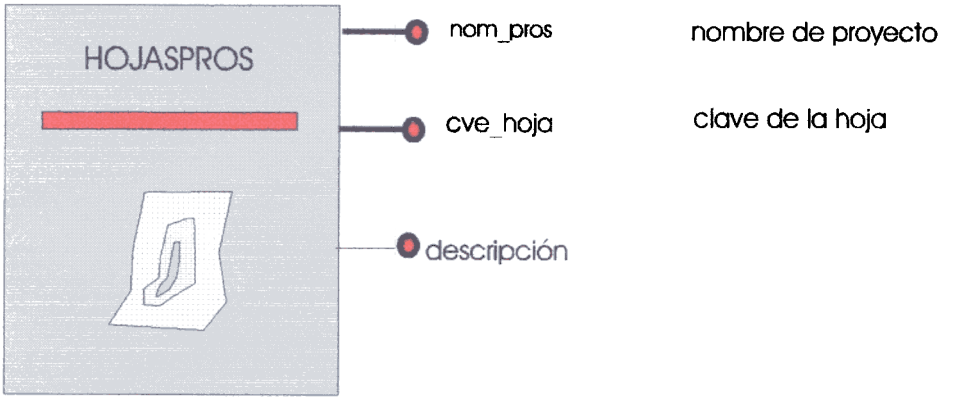


La entidad **COLUMNA**:
Esta entidad se refiere, a la columna geológica que se va cortando al perforar el pozo.

La entidad **DETENAL:**
Esta entidad tiene el nombre de la dependencia que proporciona las hojas o planos base del territorio mexicano; Entonces esta entidad guarda las hojas que se dan de alta en el sistema y a su vez genera los archivos de detalle.



La entidad **HOJASPROS**:
En esta entidad se definen las HOJAS que conforman un PROSpecto o estudio.



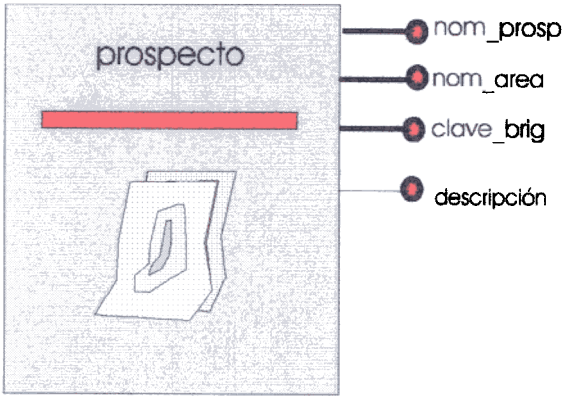
La entidad **LINEA_SISMO**:
Es una entidad que pertenece al prospecto, y que es generada por la brigada. Esta formada por nodos, coordenadas, elevaciones (no necesario), un dato z que puede ser el tiempo (no necesario) y la línea que forman los nodos; no necesariamente es una recta.

nom_linea: nombre de la línea.
nom_prosp: nombre de prospectos.
PTI: punto tiro inicial.
PTF: punto tiro final.
fechai: fecha inicio.
fechat:fecha de terminación. Las fechas indican el inicio y la terminación de los trabajos sobre la línea.
patron_detec: patrón de detección. Lo que define la colección de los sismodetectores(geofones) sobre la línea sísmica.
dist_estac: distancia entre estacas. Lo que es la distancia que existe entre nodos. El avance, nos proporciona el porcentaje de desarrollo de los trabajos sobre la línea dentro de un período de tiempo, este avance se proporciona en % de Pt's realizados.
patro_tiro: patrón de tiro.
cubierta_detec: cubierta detección.
tiempo_grab: tiempo de grabación.
num_carretes: es la cantidad de carretes empleadas para grabar las respuestas del terreno provocadas por la explosión de la dinamita.
loc_carretes: localización de carrtes.
avance_proc: avance del procesado.
tipo_proc: tipo de proceso.

La entidad **POZO**:
Esta entidad se refiere a los pozos de exploración petrolera y demás atributos referentes a ellos (coordenadas, profundidad, registros, etc.).

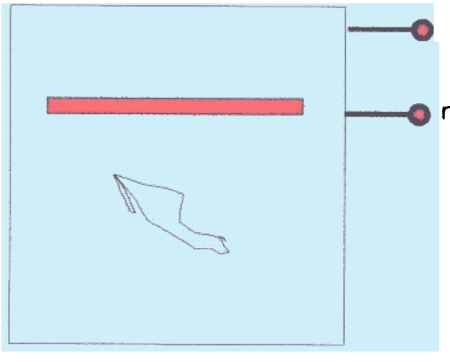
Nombre del pozo
Coordenedas X, Y (UTM)
tamaño del pozo
Metro_bajo_M.Rotaria
Nivel_bajo_M.rotaria
ley de velocidades
Profundidad total
fecha de creyó

La entidad **PROSPECTO:**
Es una entidad que presenta datos sobre el prospecto desarrollado.



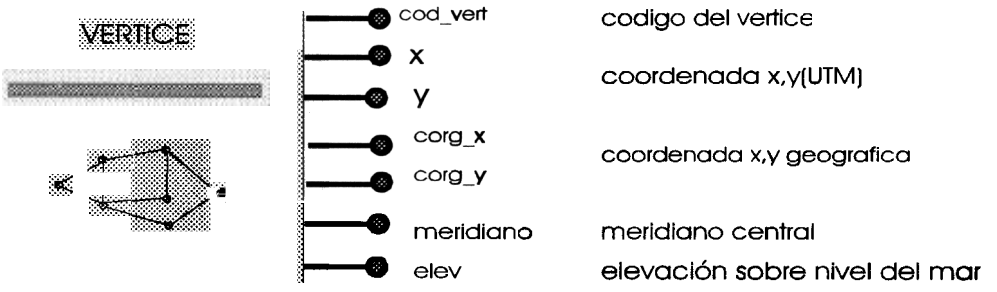
nombre de prospecto
nombre de área
clave de brigada

La entidad REGION:
Es una porción geográfica de gran extensión, en la que están contenidas áreas.



clave de region
nombre de region

La entidad VÉRTICE:
Son puntos de referencia topográfica, lo forman su nombre o clave, coordenadas x,y; el meridiano central y su elevación sobre el nivel del mar.



CONCLUSIONES

En este capítulo se mostró el procedimiento para modelar la base de datos espacial para exploración petrolera. Se usa la metodología, la cual integra una serie de esquemas de datos, operaciones y eventos en esquemas globales. Usamos el sistema EVE, el cual cuenta con representación gráfica y es el estándar de herramientas automatizadas para soportar el diseño de la base de datos. El último paso de la metodología es el diseño físico de la base de datos. El cual es dependiente del manejador de base de datos donde se implementa el diseño. En el siguiente capítulo, se describirá la arquitectura del manejador de la base de datos.

CAPITULO 3

ARQUITECTURA DEL MANEJADOR DE LA BASE DE DATOS ESPACIAL

Los sistemas manejadores de bases de datos espaciales requieren esquemas de datos capaces de almacenar tantas entidades espaciales con sus vínculos, como esquemas conceptuales de información nominal que soporten de manera integrada la manipulación espacial y nominal. Algunas de las estructuras más conocidas son: estructura de datos quadtree [SAME90]; y estructuras indexadas basadas en R-trees [GUTT84]. En las siguientes secciones se definirán estos elementos.

3.1 QUADTREES

La estructura de datos quadtrees se basan en el principio de la descomposición recursiva. Esta técnica es la más utilizada cuando una imagen es una matriz A de bits de tamaño $2^n \times 2^n$ con ($n \geq 1$). Entonces A puede ser dividido en cuadrantes cuyos tamaños son de 2^{n-1} bits. Esta división puede ser repetida recursivamente n veces. El proceso es generalmente representado como un árbol [SAME90], en el cual el nodo raíz corresponde a la imagen completa, cada nodo no terminal tiene cuatro hijos, y cada nodo terminal corresponde a una área cuadrada uniforme. Los nodos están conectados si uno de los cuadrados inmediatos correspondientes contiene a los otros. Lo anterior se ilustra en la figura 3-2 con la imagen de la figura 3-1, considerando todos sus puntos como esquinas superiores izquierdas de cuadros uniformes.

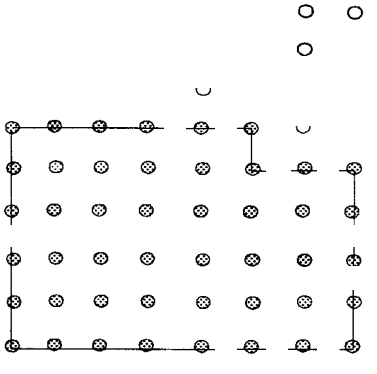


figura 3-1 región de puntos negros limitada por polígonos

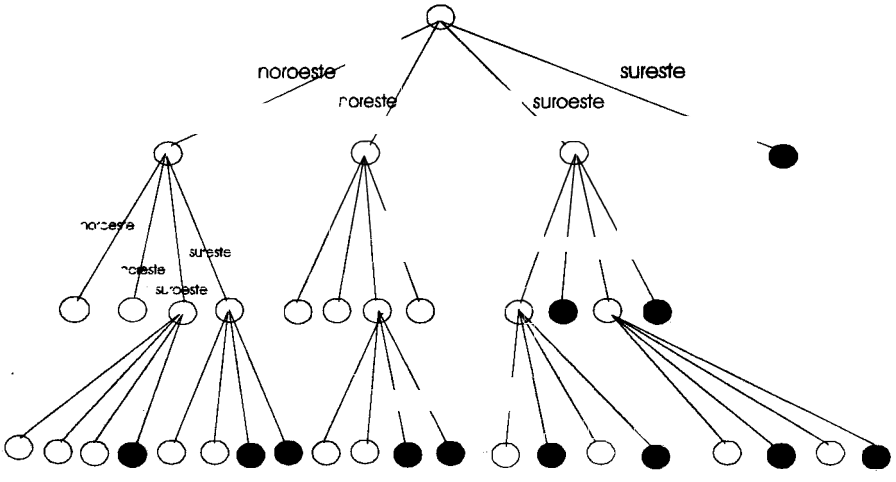


Figura 3-2 Quadtree del cuadrado y el triángulo inferior de la figura3-

Los quadtree son eficaces en la búsqueda de objetos espaciales. Esta estructura admite un almacenamiento eficiente para la información relativamente homogénea. También proporcionan resolución variable de las estructuras de datos, donde los nodos terminales pueden estar en cualquier nivel dependiendo del nivel donde se de la agrupación de nodos hermanos. Esta característica permite que la geometría sea accesible fácilmente.

3.2. R-TREES y R^+-TREE

3.2.1 R-tree

Los R-trees fueron propuestos por primera vez por [Gutt84], son una extensión de los B-trees. Los árboles están compuestos por un nodo raíz, por nodos intermedios y nodos-hojas. A todos los nodos-intermedios se les asigna un número mínimo y máximo de entradas y cada entrada contiene un identificador que localiza su nodo hijo y también la región espacial, (x_{min}, y_{min},x_{max},y_{max}), que el hijo cubre. En la figura 3-3 están representados varios objetos en la figura 3-4 el R-tree correspondiente, con un número máximo de cuatro apuntadores por nodo.

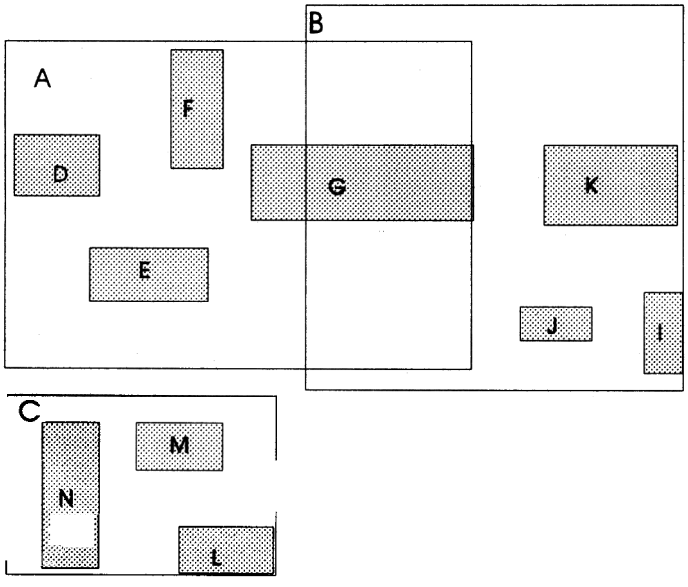


Figura 3-3 Organización de los datos para un R-tree.

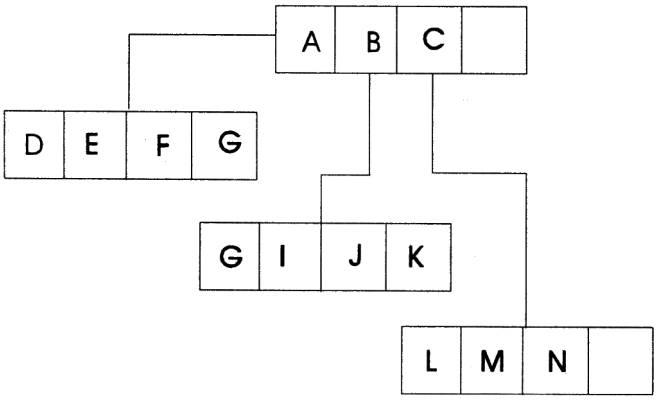


Figura. 3-4 El R-tree correspondiente a la figura 3-4.

Cada nodo intermedio tiene asociada una región rectangular, que encierra todos los rectángulos más pequeños de los nodos descendientes. A esta región se le llama rectángulo delimitaste mínimo (minimal bounding rectangle) o RDM. Los nodos intermedios de un nivel dado se pueden sobreponer por lo tanto sus RDMs no representan regiones disjuntas.

La cantidad de sobreposición de los nodos que pertenecen a un mismo nivel, depende directamente del número de rutas que deben de ser atravesadas por una consulta espacial (spatial query).

En el caso extremo, si cada rectángulo de los nodos intermedios cubre completamente el espacio de búsqueda entonces cada nodo tendrá que ser inspeccionado al hacer una consulta. En general es difícil generar un árbol que tenga pocas sobreposiciones entre los nodos hermanos en los niveles superiores, a no ser que se lleve a cabo un rearreglo de la estructura de índices o que la inserción de datos se lleve a cabo en un orden óptimo.

Si al insertar un nuevo objeto en una hoja, la hoja se llena, entonces es dividida y esta división origina que la página de su predecesor se sobreexceda, entonces la operación de división se propaga hacia los niveles superiores. El algoritmo de división es crucial para determinar la cantidad de sobreposiciones en los rectángulos existentes.

Así por ejemplo, si al conjunto de objetos de Figura 3-4 le agregamos un nuevo objeto, el H, entonces lo deberíamos de registrar en el subárbol B, pero como la hoja ya esta llena, entonces se origina una división y se generan las hojas O y P (figura 3-5), readaptándose los RDMs de acuerdo a la figura 3-6.

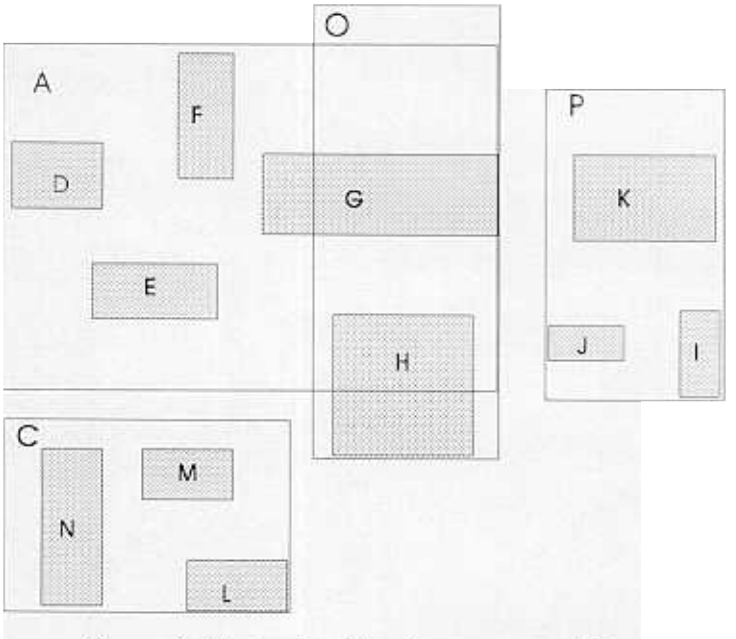


Figura 3-5 introducción de un nuevo objeto

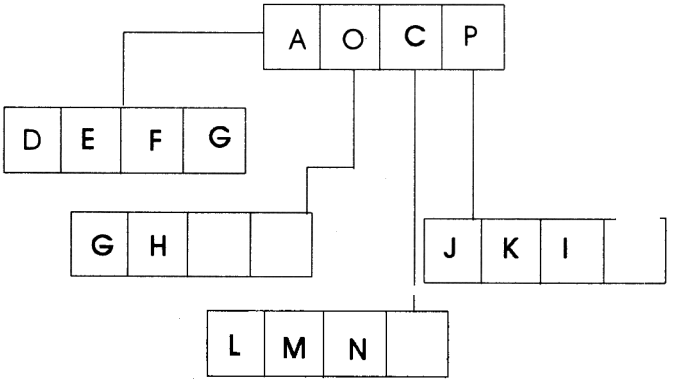


figura 3-6 árbol correspondiente a la figura 3-5

Considerando la eficiencia de la búsqueda en R-trees, los conceptos de cobertura y sobreposición son importantes. La cobertura de un nivel de un R-tree está definida como el área total contenida dentro de todo el conjunto de nodos en un nivel dado y sobreposición es el área común entre 2 o más nodos. Obviamente, una búsqueda eficiente en un R-tree requiere que tanto la cobertura como la sobreposición sean minimizadas. Una cobertura mínima reduce el promedio de espacio muerto (i.e. espacio sin objetos) cubierto por los nodos y una sobreposición mínima reduce el número de accesos que son necesarios para acceder un objeto. Llevar a cabo una sobreposición mínima es más difícil que una cobertura mínima. Para buscar una ventana que está en el área de K nodos que se sobreponen en el nivel h-l, siendo h la altura del árbol, en el peor de los casos se tienen que recorrer K caminos (i.e. uno por cada uno de los nodos sobrepuestos), reduciendo la búsqueda de l a lk accesos. Por ejemplo, para buscar la ventana W mostrada en la figura 3-7 tanto el subárbol con raíz A como el subárbol con raíz B deben de ser recorridos, aunque solamente el último regresará un rectángulo apropiado. Claramente, debido a que es muy difícil controlar la sobreposición durante la división dinámica de R-trees, la búsqueda se degrada y puede incluso degenerar a tal grado que su costo ya no es logarítmico sino lineal.

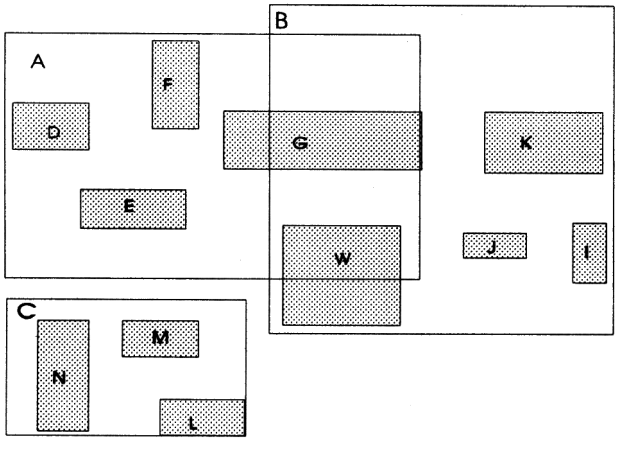


Figura. 3-7 Organización de un R-tree

3.2.2 R^+-trees

Consideremos primeramente uno de los problemas que los R-trees tienen, se ha demostrado [Rous85] que sólo se puede evitar tener regiones que se sobrepongan cuando se tienen puntos que son conocidos de antemano. En la misma referencia se muestra que siempre se van a tener rectángulos que se sobrepongan en el caso de que se manejen regiones y no puntos. Sin embargo, si se permite que se dividan los rectángulos que encierran a las regiones, entonces sí es posible generar un árbol en el que los nodos intermedios no se sobrepongan. En la figura 3-8 se ejemplifica un agrupamiento diferente de los rectángulos de la figura 3-7 y en la figura 3-9 se muestra el R^+-tree correspondiente.

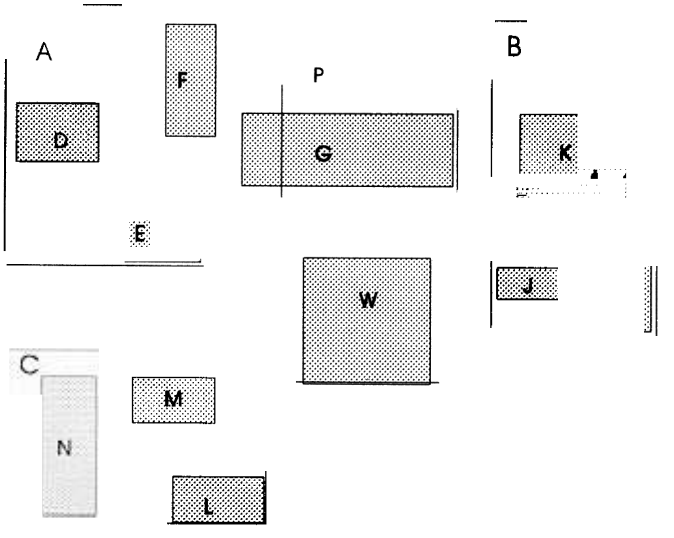


Figura3-8 Otro ordenamiento para los rectángulos de la figura 3-7

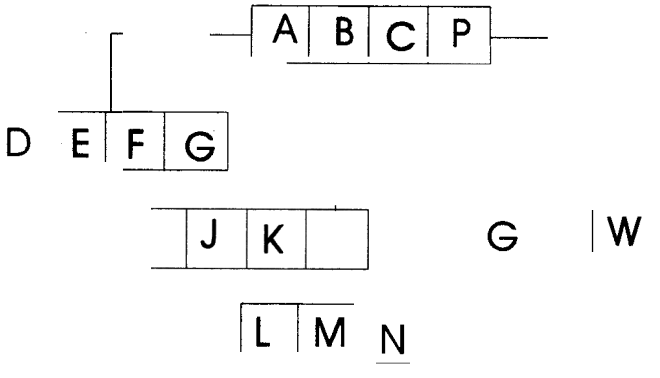


Figura3-9. R+-tree correspondiente a la figura 3-8.

Nótese que el rectángulo G ha sido dividido en 2 subrectángulos, el primero contenido en el nodo A y el segundo en el nodo P. Esto es, cuando un rectángulo delimitaste mínimo situado en un nivel bajo se sobrepone con otro, se debe de descomponer en una colección de subrectángulos que no se sobrepongan de cuya unión se obtiene nuevamente el rectángulo original.

3.3 ESTRUCTURA DE DATOS DEL SISTEMA

Debido a la necesidad de la búsqueda de objetos, es deseable tener un índice que tome en consideración las relaciones de lugar entre los objetos. Normalmente, las técnicas clásicas de indexado, usadas en bases de datos de una dimensión no son apropiadas para la búsqueda espacial multidimensional. Por ejemplo, las estructuras basadas en la técnica de matching exacto de valores, tales como las tablas de hash, no son útiles, debido a que los datos que se buscan están en un rango de valores y no en un valor único. Las estructuras que usan el ordenamiento de valores llave, como por ejemplo los B-trees no funcionan si se aplican de la misma forma que para bases unidimensionales. Para resolver entonces el problema del acceso espacial a objetos se han propuesto diferentes técnicas.

3.3.1 Por qué no usamos quadtree o R-tree?

En las bases de datos convencionales se han desarrollado varios tipos de modelos de estructura espacial mencionados en la sección anterior para el almacenamiento de datos espaciales. La efectividad de tales modelos ha sido confirmada desde un punto de vista teórico[SAME90].

Pero ellos tienen sus defectos particulares. En la representación de quadtrees o R-trees la información nominal es almacenada en los nodos del árbol. Sin embargo, en muchas bases de datos espaciales, tales como los GIS's, las entidades espaciales se asocian con información nominal, que generalmente consta de estructuras de datos muy complejas. Y la información descriptiva de las entidades puede estar vinculada con unas u otras entidades. Es claro que el almacenamiento de tal información es difícil en los nodos de los quadtrees o R-trees. Mas aún, cuando la cantidad de gestión descriptiva varia de entidad a entidad o cuando varios objetos están cubiertos, provoca que la administración sea difícil y compleja.

El objetivo de tener ambos enfoques de organización de datos, consiste en integrar las estructuras nominales y espaciales. Por un lado, requerimos el manejo de relaciones topológicas que presentan las entidades en dos o tres dimensiones. Por el otro, requerimos del manejo de las relaciones dadas por los atributos asignados a dichas entidades. Unificándose dentro de una arquitectura general es posible tener un sistema con capacidad de resolver problemas de naturaleza geográfica.

3.3.2 estructura general

Del planteamiento general de requerimientos, llegamos a un diseño en donde las entidades son objetos que tienen una representación espacial codificada en coordenadas, en algunos casos imágenes, con la asignación conceptual de sus nombres y un conjunto de atributos. La idea es combinar la estructura de B-árbol y multilistas para el almacenamiento de la información.

En el sistema de base de datos espacial, por lo general, los requerimientos plantean la necesidad de realizar operaciones que involucran tanto datos nominales como datos con una representación gráfica. Estas operaciones comprenden, entre otras, el despliegue en pantalla o una consulta gráfica (interactuar con la imagen en pantalla).

Los requerimientos de datos nominales de la base de datos implican en la mayoría de los casos un registro (e.g. parámetro de una línea) y en algunas ocasiones a varios registros que por término general no pasan de un par de decenas (e.g.. la columna geológica de un pozo). Para dar respuesta a estos requerimientos es necesario una recuperación rápida y selectiva en la base de datos.

En la manipulación de datos gráficos como puntos, líneas o polígonos, se requiere de una estructura diferente a la utilizada en los datos nominales. La gran cantidad de información que se involucra en un despliegue gráfico que en general comprende varios y en ocasiones miles de registros de la base de datos, demanda el uso de una estructura que permita la recuperación masiva y selectiva.

Bajo los conceptos anteriores se plantearon y desarrollaron dos tipos de archivos de datos:

a) Archivo para datos nominales.

b) Archivo para datos gráficos.

Usamos estructuras de Arbol B para manejar los datos nominales y estructuras de Multilistas para los datos gráficos. Las dos estructuras se conectan como se muestra en la figura 3-10.

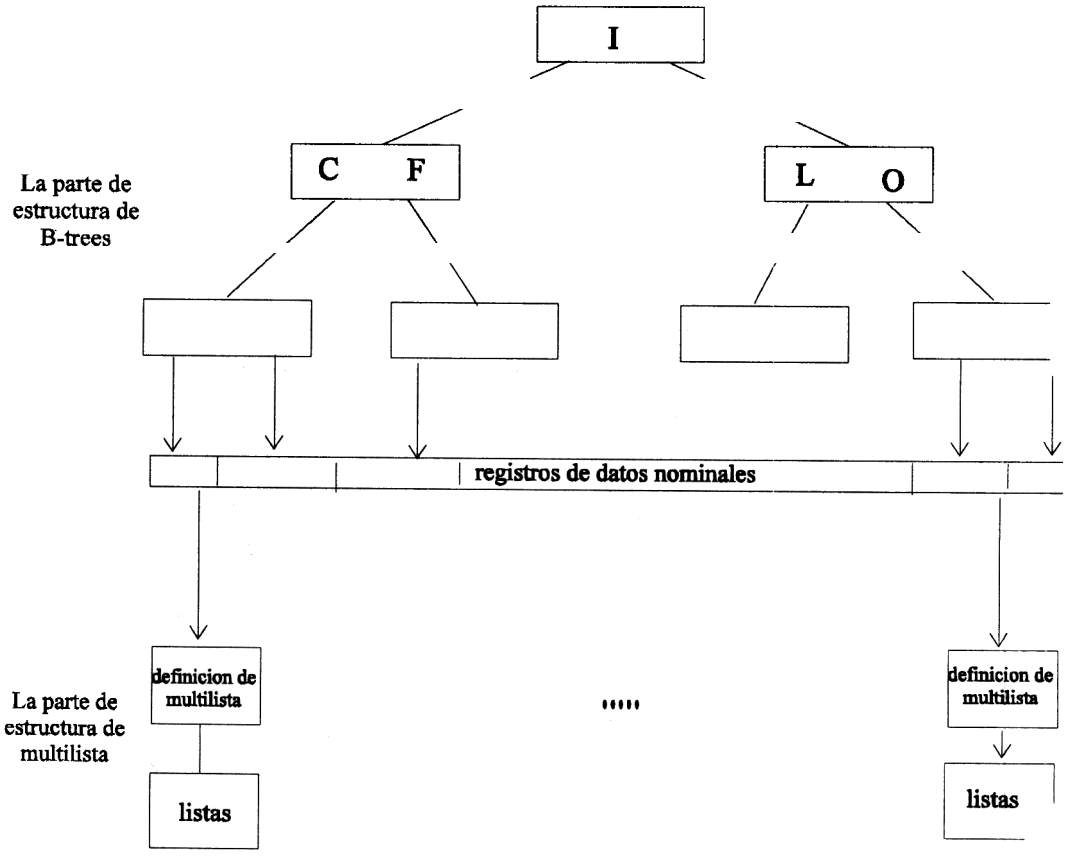


Figura 3-10 La combinación de dos estructuras lógicas.

En la figura 3-10, la parte de arriba es una estructura de Arbol B que tiene indexado los archivos de datos nominales. Si en un archivo usamos dos campos, tenemos dos llaves a cada una de las cuales se le asigna una estructura de Árbol B cuyo Archivo se nombra *.X01 para la primera llave, *.X02 para la segunda, y así sucesivamente, según el número de llaves utilizadas, como se muestra en figura 3-11. Los datos nominales se almacenan en un archivo con extensión *.DAT.

En el caso especial en que los archivos con extensión *.DAT tengan registros que contengan archivos anexos, se tiene para cada registro un apuntador a los archivos con extensiones *.DEF y éste a su vez tendrá apuntadores a los archivos con extensión *.GRA o *.BIT.

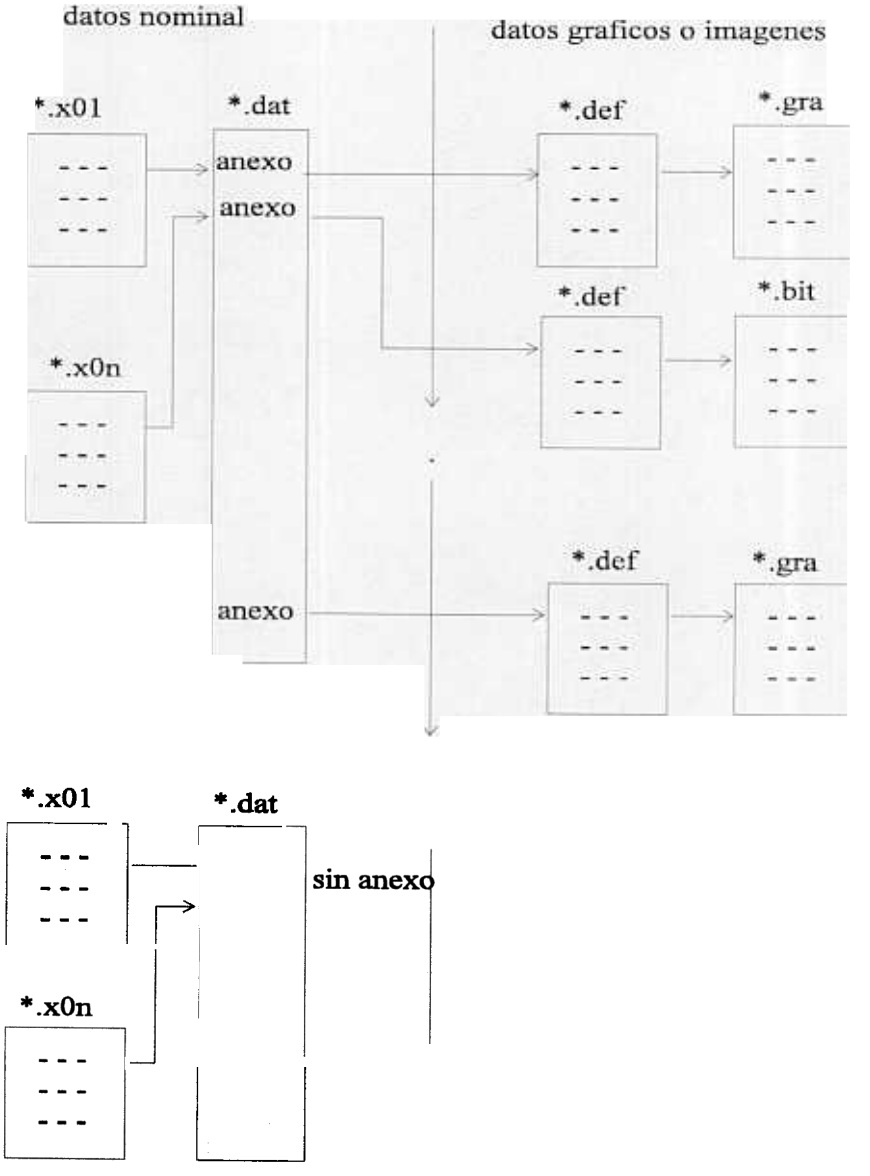


Figura 3-11 Muestra la relación entre los archivos de 5 tipos.

Los archivos *.X01,...,*.X0n son los archivos donde se almacena la información del árbol B, *.DAT son los archivos donde se definen los datos para gráficos o imágenes y *.GRA son los archivos donde se almacenan los datos gráficos. De estos dos últimos se hablará en la siguiente sección.

3.4 ARCHIVOS DE DATOS GRÁFICOS.

3.4.1 relaciones entre archivo definición y gráficos

La información gráfica se mantiene en dos archivos: uno que contiene todos los registros con representación gráfica (NOMBRE.DEF), y otro, para mantener la descripción de los datos gráficos almacenados(NOMBRE.GRA) y su posición relativa dentro del archivo (figura. 3-12).

El archivo de datos gráficos contiene únicamente registros con datos (o sea, el archivo de datos es una colección de registros de datos), la longitud (fija) de cada registro es la suma de las longitudes (fijas) de sus elementos de datos (campos). Todos los registros en el archivo están en forma consecutiva. Puede verse al archivo de datos como un subconjunto de listas y cada una agrupando uno o varios registros.

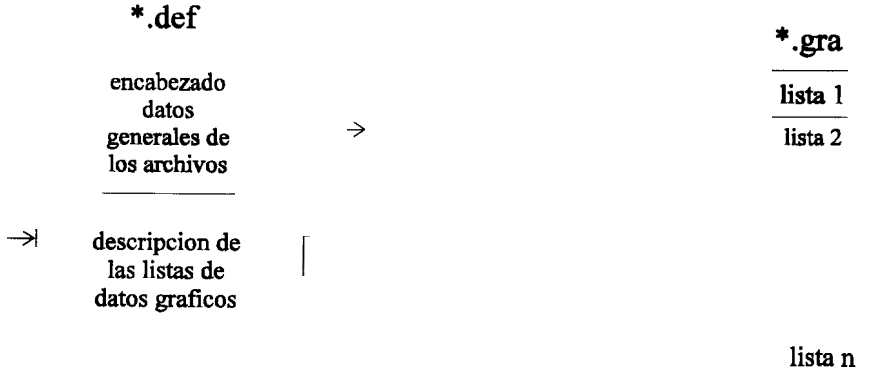


Figura 3-12 La relación entre archivos de definición y gráfica

Los archivos con datos gráficos están compuestos de múltiples listas (archivos de multilistas) donde cada lista en el archivo contiene información de una representación gráfica, que es homogénea desde el punto de vista de lo que representa como imagen (e.g. ríos, líneas, perímetros geográficos), donde cada una de estas imágenes estará contenida en una lista.

El acceso a una lista en el archivo es aleatorio. Posterior el almacenamiento de la lista en memoria principal, el acceso a cada registro de la misma es secuencial, y la descripción del archivo de las multilistas se mantiene por separado en otro archivo (NOMBRE.DEF). En este archivo está toda la información relevante de ambos archivos (NOMBRE.DEF y NOMBRE.GRA).

El archivo de definición está compuesto de dos partes, la primera contiene información general sobre los dos archivos y en la segunda parte, información de cada lista almacenada. La estructura y contenido de la información dependerá del tipo de lista manejada. Como se verá más adelante, hay dos tipos de estructuras para almacenamiento y manejo de las listas, y las cuales son definidas como "estándar" y de "detalle".

```
struct header_descriptor {
        char   nom_arch[40];
        int    tipo_reg;
        int    libres;
        int    med_reg;
        int    num_campos;
        int    campo_llave;
        int    arch_padre;
        char   tipodesc;
        float  corxmax;
               corymax;
        float  corxmin;
        float  corymin;
      } hdlis;
```

La estructura "header_descriptor" define la información que se almacena en el "encabezado" del archivo de definición, esta información que es la misma para los dos tipos de estructuras de almacenamiento de datos gráficos.

.nom_arch.- Cadena original con la que se formó el nombre del archivo gráfico y el de definición (e.g. DETALLE DR COSS" forma a DDC515.GRA y DDC515.DEF).

.tipo_reg.- Tipo de registro almacenado en el archivo de datos gráficos definido en el descriptor de esquemas).

.libres.- Número de lista (inclusive) a partir de la cual están disponibles.

.med_reg.- Tamaño en bytes de cada registro almacenado.

.num_campos.- Número de elementos (campos que contiene el registro).

.campo_llave.- Posición de elemento de datos que se usa como llave.

.arch_padre.- Número del archivo padre (de datos nominales) con el que está vinculado el archivo gráfico.

.tipo_desc.- Tipo descriptor gráfico que se usó (definido en descriptor de esquemas) para el actual archivo de datos ('E' para estándar o 'D' para detalle).

corxmax.- Coordenada (UTM) máxima en las 'x' entre la cual están los datos gráficos almacenados.

.corymax.- Coordenada (UTM) máxima en las 'y'

.corxmin.- Coordenada (UTM) mínima en las 'y'.

corymin.- Coordenada (UTM) mínima en las 'x'

En el campo "nom_arch" se almacena la cadena original con la que se formó el nombre del archivo, esta cadena corresponde al valor de un campo llave de una entidad nominal con un anexo gráfico. Cuando se da de alta un registro en una tabla de datos nominales y esta tabla está declarada en el descriptor con liga a una descripción gráfica, se procede a generar los archivos gráficos (el de datos y el de definición).

3.4.2 Estructura de definición de listas estándar

Para la definición de listas "estándar" se utiliza la estructura "desc_arch", esta contiene los elementos para el control de las listas de registros almacenadas en el archivo de datos gráficos (con un formato "estándar").

```
/* estructura de control listas de estandar*/
struct desc_arch{
        char    id_nombre[name_size]
        int     num_recs;
        int     siguiente;
      }listas[MAX_LISTAS];
```

En la estructura "desc_arch" el elemento "id_nombre[]" se utiliza para almacenar el identificador a la lista (e.g. "L_100" que en el sistema corresponde conceptualmente a la línea denominada 100) y es predefinido por el usuario, en "num_recs" se almacena el número de registros correspondientes a la lista y el campo "siguiente" indicara el número que identifica a la siguiente lista. la figura 3-13 muestra como se relacionan el archivo de definición estándar y el archivo de datos.

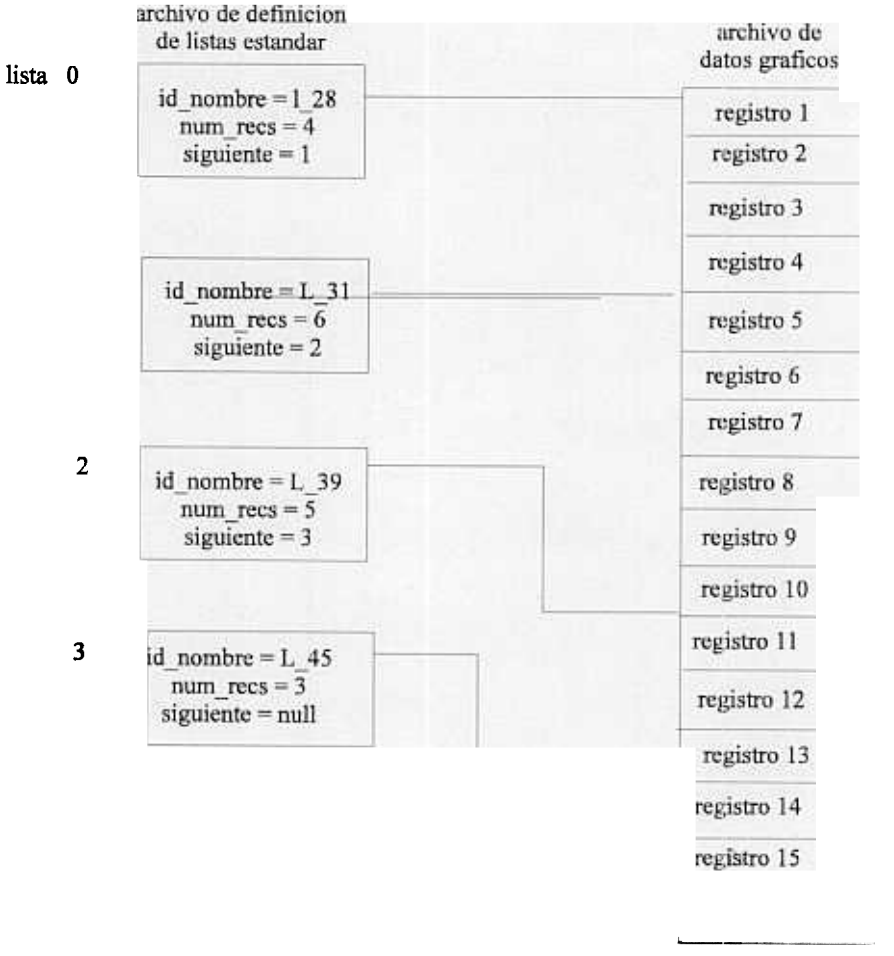


Figura. 3-13 Estructura de listas estándar

3.4.3 Estructuras de listas de detalle.

Si la información gráfica de cierta entidad requiere que se almacene como de "detalle", en la segunda parte del archivo de definición estará la estructura de listas de detalle como se muestra en la figura. 3-17 y donde pueden identificarse los elementos.

```
/*estructura de control listas de detalle */
struct desc_archdet{
    char        id_nombre[NAME_SIZE]  /*identificacion de la lista      */
    int         num_recs;             /*numero de registros en archivo */
    struct det_lis  det[DETGRAF];     /*sublistas de la lista          */
    int         siguiente;            /*siguiente lista en el archivo  */
                listasdet[MAX_LISTASDET]
```

Los elementos anteriores tienen el mismo significado descrito anteriormente para listas "estándar" con la excepción del elemento "struct det_list det[]" que es exclusivo de listas de detalle.

La siguiente figura muestra una representación de detalle del descriptor y como éste se vincula con el archivo de datos gráficos.

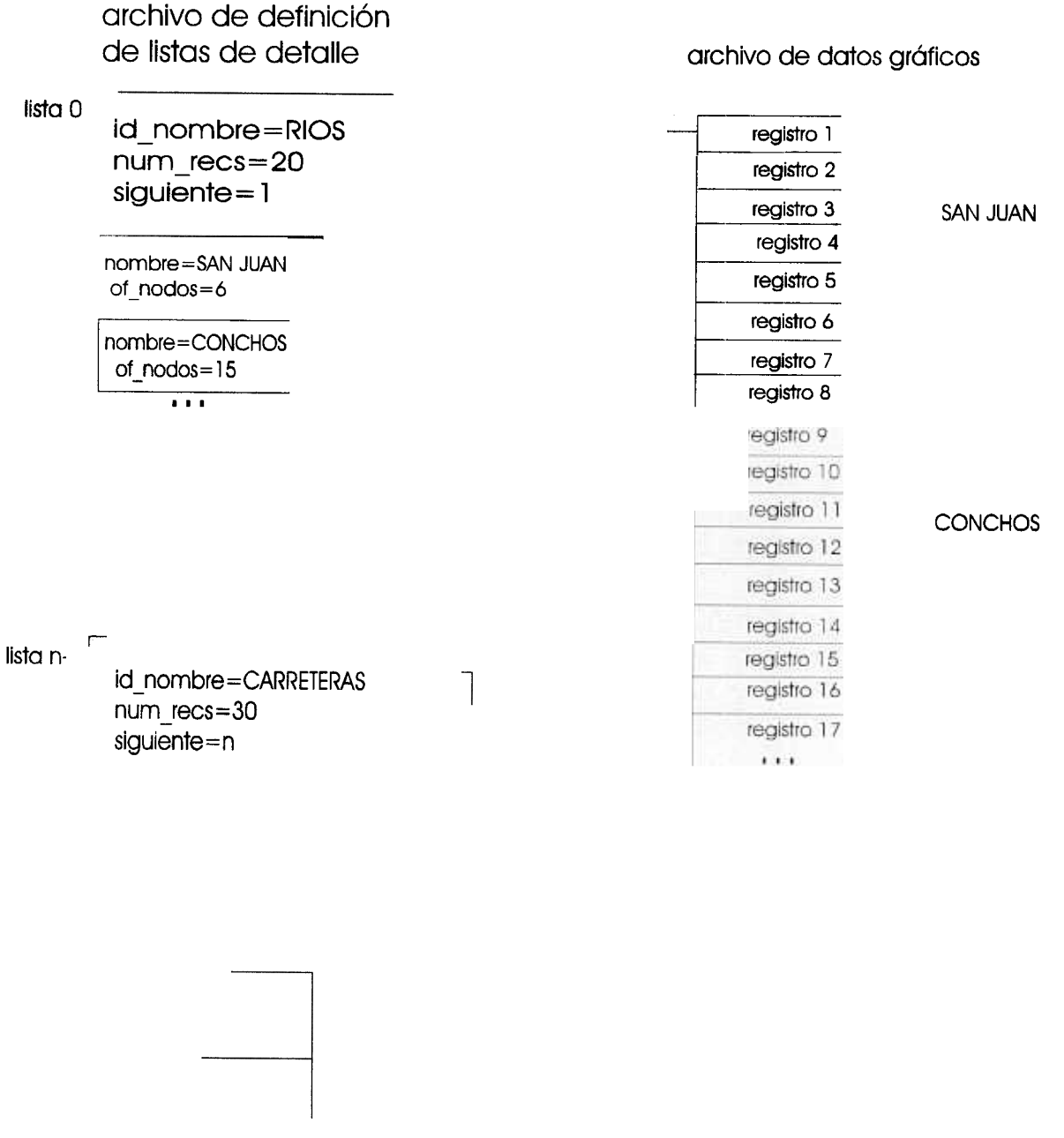


Figura. 3-14 Estructura de listas de detalle.

La estructura "detalle" contiene un elemento más, un arreglo de estructuras que describe las diferentes partes que componen la lista.

```
/*estructura para detalle en listas detalladas*/
struct det_lis{
    char    id_det[NAME_SIZE]; /*identificador de un grupo de nodos de la lista*/
    int     of_nodos;          /*desplazamiento en nodos del inicio al último del
                                 actual grupo*/
```

Lo anterior puede verse como una lista compuesta de sublistas, es decir, la descripción de grupos de nodos que constituyen una lista. En esta descripción se especifica el último nodo de cada grupo (donde este grupo termina) como un desplazamiento desde el inicio de la lista, así como su identificador.

Este tipo de descripción detallada puede usarse en los planos base -como las cartas de DETENAL-, que contienen, por ejemplo, ríos, carreteras, límites estatales o municipales.

En la figura 3-14 puede identificarse la lista que almacena "RIOS", esta lista contiene todos los puntos geográficos de los ríos contenidos en una carta de DETENAL. El nombre de cada río contenido en la lista ("RIOS") está almacenado en cada "id_det" de la lista, así como su desplazamiento en registros del último registro de la sublista en relación al inicio de la lista y el río Conchos tiene su último registro a 15 del inicio de la misma.

Para la manipulación de la lista (e.g para despliegue) se tiene cada identificador de los ríos, el inicio de cada sublista se obtiene sumándole a la dirección del inicio de la lista el desplazamiento "of_nodos" del detalle anterior (o si es el primer detalle), además se puede calcular el número de registros correspondientes a cada sublista restando "of_nodos" del actual detalle al "of_nodos" del detalle anterior (si es la primera sublista, el número de registros es igual al actual "of_nodos").

3.5 ARQUITECTURA DEL SISTEMA VISDA

La arquitectura general del sistema **VISDA** consiste principalmente en los módulos (figura 3-15)

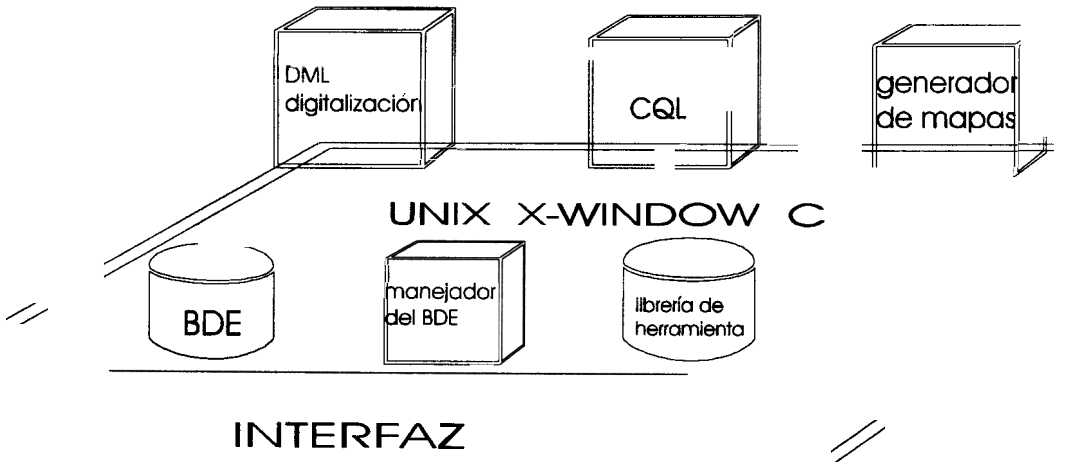


Figura 3-15 Arquitectura del sistema VISDA

Interfaz visual. La parte frontal del sistema es una interfaz visual de uso fácil y amigable para los usuarios. El módulo está construido sobre X-Window y lenguaje C. El usuario puede llevar a cabo diversas operaciones que le permitan: consultar la base de datos con información alfanumérica y gráfica; manipulación y gráficación de mapas; análisis espacial de datos y presentación estadística de la información.

Manejador de la base de datos espacial. Una de las partes más importantes del sistema es el gestor de datos. Mediante éste, los sistemas anteriores pueden acceder a la base de datos de una forma eficiente permitiendo estructurar los objetos espaciales y nominales. Mientras los objetos espaciales: puntos, líneas y polígonos; se determinan en base a una organización de estructuras de multilistas. La información nominal se puede acceder mediante la estructura de árboles B como mencionado anterior.

En el **DML**, un módulo importante es la entrada de datos. Según afirmaciones de especialistas, entre el 80 y 90% del costo de operación de una base de datos pictográfica, radica en el costo de consecución, obtención y actualización de los datos. La entrada de datos cubre todos los aspectos de digitalización de mapas existentes (normalmente desactualizados), observaciones del campo a través de levantamientos topográficos y obtención de imágenes a través de fotografías aéreas, imágenes de satélite, etc. Este proceso además de costoso puede requerir mucho tiempo. Teniendo en cuenta que la confiabilidad y precisión de la base de datos depende en gran medida de la calidad de los datos, es necesario analizar su origen (fuentes de información, técnica empleada y nivel de precisión.) y establecer criterios de control de calidad y procedimientos que garanticen la actualización de datos. En el siguiente capítulo se explicarán más a detalle estos términos.

En el paquete del **CQL**, se incluye el módulo de interfaz de comandos de consultas, el de interprete de consultas. El lenguaje se definirá en el capítulo 5.

Los lenguajes antes mencionados están caracterizados fundamentalmente por un ambiente visual, incluyendo en ellos ventanas e iconos. En general, el espacio de trabajo es la misma información gráfica, mostrada en mapas.

El generador de los mapas que sirven para visualizar los resultados de las consultas. Esta parte se describirá en el capítulo 8.

CONCLUSIONES

En este capítulo se describió la arquitectura del sistema VISDA, y la estructura de datos. Se emplea la estructura que combina estructura de árboles B y estructura de mulitlistas para soportar de manera integrada la manipulación los datos espaciales y nominales sobre las entidades de la base de datos espacial aplicada al área de geográfica.

CAPITULO 4

LENGUAJE VISUAL DE MANIPULACIÓN DE DATOS

Las operaciones del usuario se expresan (usando el DML) en términos de registros externos, y el DBMS debe convertirlas en las operaciones correspondientes sobre los registros internos o almacenados. Estas últimas operaciones deben convertirse a su vez en operaciones al nivel real del hardware, es decir, en operaciones sobre registros físicos o bloques. La componente responsable de esta conversión interna/física se llama método de acceso(figura 4-1). El método de acceso se compone de un conjunto de rutinas cuya función es ocultar al DBMS todos los detalles dependientes de los dispositivos y presentar una interfaz de registros almacenados. De esta manera, la interfaz de registros almacenados corresponde al nivel interno, así como la interfaz con el usuario corresponde al nivel externo.

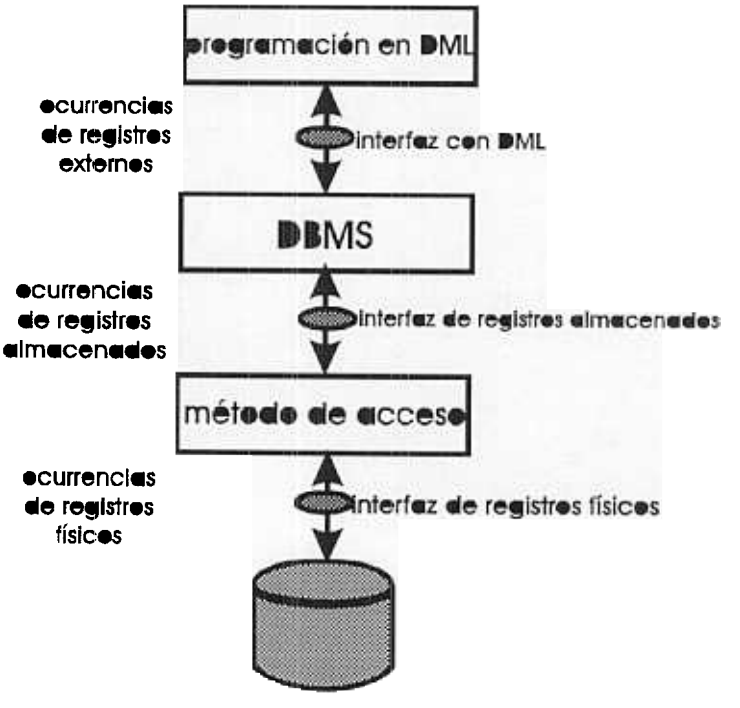


Figura 4-1 Interfaz de registros almacenados

El método de acceso se puede considerar como una extensión lógica del DBMS, y en realidad en algunos sistemas los dos están empacados conjuntamente. En otros casos, el DBMS cuenta con que el sistema operativo subyacente proporciona las funciones del método de acceso.

La interfaz de registros almacenados permite al DBMS ver la estructura de almacenamiento como un conjunto de archivos almacenados, cada uno compuesto de todas las ocurrencias de un tipo de registro almacenado. En términos específicos, el DBMS conoce (a) qué archivos almacenados existen y de cada archivo, (b) la estructura del registro almacenado correspondiente, (c) el campo almacenado, si lo hay, sobre el cual está secuenciado, y el campo almacenado, si lo hay, que pueda usarse como argumento de búsqueda para el acceso directo. Toda la información se especificará como parte de la definición de la estructura de almacenamiento. Adviértase que los puntos anterios equivalen a decir que el DBMS sabe qué proposiciones de acceso puede emitir sobre el archivo almacenado. Nótese también que la unidad que atraviesa la interfaz de registros almacenados es una ocurrencia de registro almacenado.

Desarrollamos la interfaz de un lenguaje de manipulación de datos (DML), que apoya el manejo y procesamiento de los objetos de la base de datos usando una filosofía de visualización.

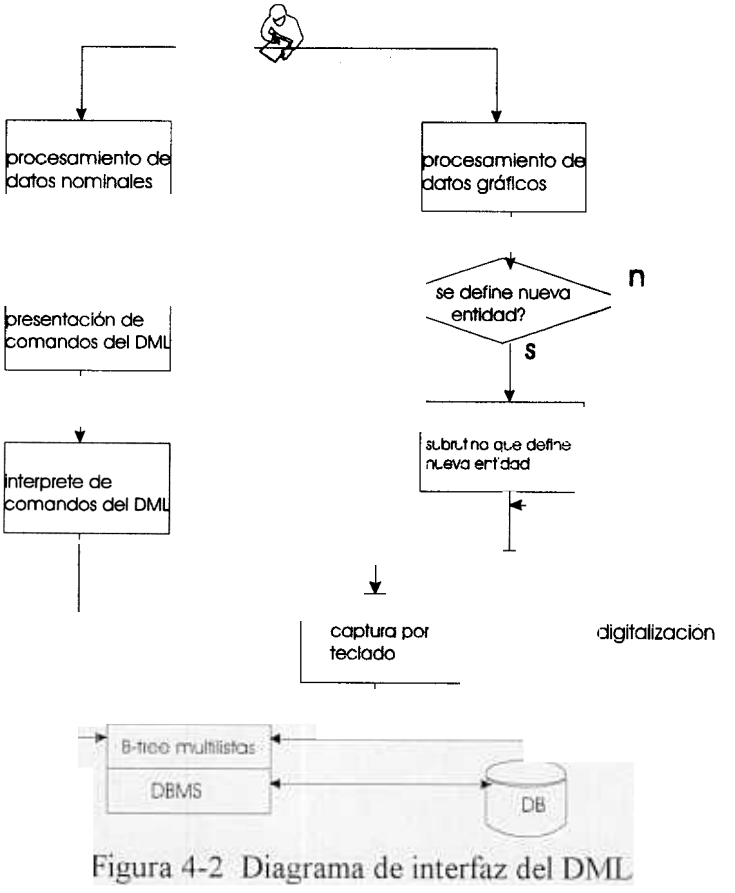


Figura 4-2 Diagrama de interfaz del DML

En el capítulo anterior, ya describimos que el modelo físico de la base de datos está estructurado por B-tree y multilista. Teóricamente podemos decir que el B-tree maneja los datos nominales, y la multilista los datos gráficos. Sin embargo, la información gráfica es representada en forma nominal. Por lo tanto, para manejar la información de una entidad necesitamos ambas estructuras, aunque trabajen en dos líneas independientes.

Como se puede observar en la figura 4-2, el DML se divide en dos líneas: la manipulación de los datos nominales y la de los datos gráficos. En la sección siguiente, describimos el manejo de datos nominales, y posteriormente el de datos gráficos.

4.1 UN AMBIENTE VISUAL DEL DML

La línea de la manipulación de los datos nominales hace las operaciones sobre los archivos nominales. En su presentación, se emplea un ambiente visual. Este ambiente emplea ventanas que incluyen **iconos, botones, imágenes y textos** para una mejor visualización de la estructura de **archivos, registros y comandos** del lenguaje.

4.1.1 Visualización de las características de los elementos de la base de datos

En los capítulos 2 y 3, describimos como se diseñó y estructuró la base de datos, para los usuarios esto es una "caja negra". Lo importante para ellos es la salida de la caja negra, en otras palabras, la interfaz.

Usamos un ambiente integrado para que los usuarios realicen consultas en cualquier momento, por ejemplo, como se muestra en la figura 4-3, hay 4 ventanas: la ventana de ***diccionario,*** la ventana de ***llaves***; la ventana de los ***archivos***; y la ventana de ***ligas de archivos***.

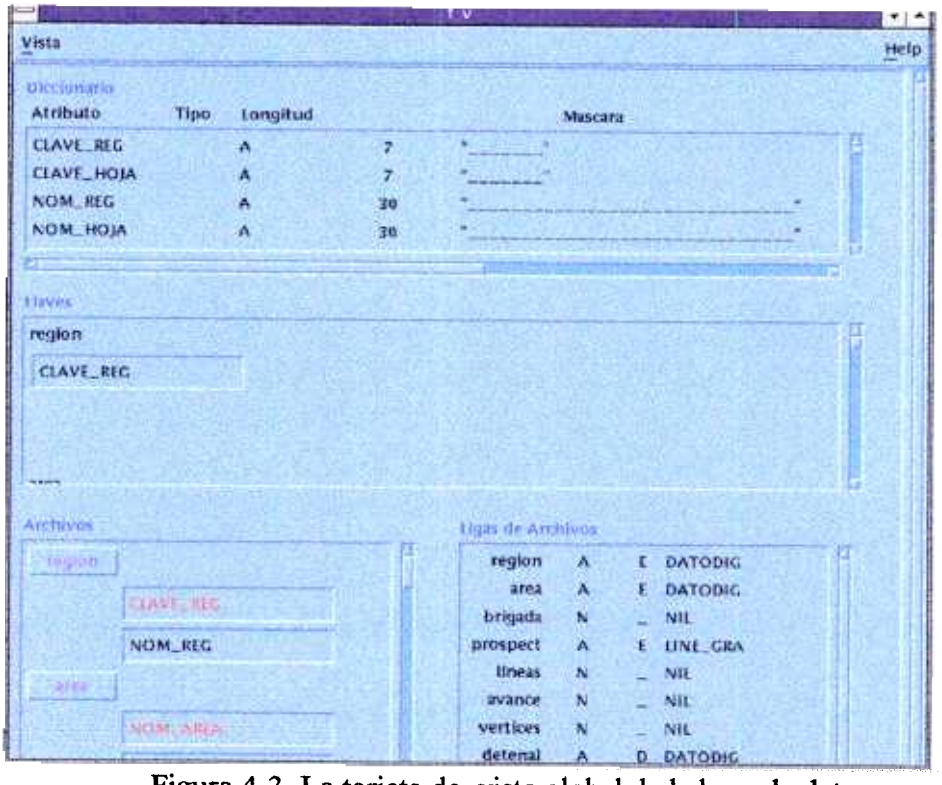


Figura 4-3 La tarjeta de vista global de la base de datos

En la ventana de ***diccionario*** se describen los nombres de atributos, los tipos de datos, las longitudes de cada atributo y las mascaras de despliegue y captura de los atributos.

En la ventana de ***llave***, hay una lista que colecta las llaves para cada archivo, determinando llaves primarias y secundarias.

En la ventana de ***archivo***, se presentan los atributos para cada archivo, identificando con color rojo a los atributos que forman parte de la llave.

En la ventana de ***ligas de archivos***, nos describe los tipos de archivos del sistema, los cuales pueden ser gráficos, nominales o nominales con anexo gráfico. Por ejemplo, en la línea de "prospecto" hay una "E", lo cual significa que el tipo de archivo es "ESTANDAR" (en el capítulo 3 se explicó que la "D" significa "DETALLE").

La tarjeta de vista global muestra todas las características de los elementos de la base de datos, y ayuda a los usuarios a definir y dar de alta los datos de entrada.

4.1.2 Visualización de los comandos del DML

Para programar usando el DML, se inicia con las tarjetas, cada tarjeta representa un archivo de la base de datos, como se muestra en la figura 4-4. Si el usuario se interesa por un archivo, coge y expande la tarjeta usando el ratón. En este momento el archivo físicamente ya está abierto. En la cara de la tarjeta aparece el nombre de archivo, simbolizándose con una imagen, así como los campos del registro. Los campos que integran la llave del archivo se identifican porque tienen asociada la imagen de una llave, como se muestra en la tarjeta expandida de la figura 4-4.

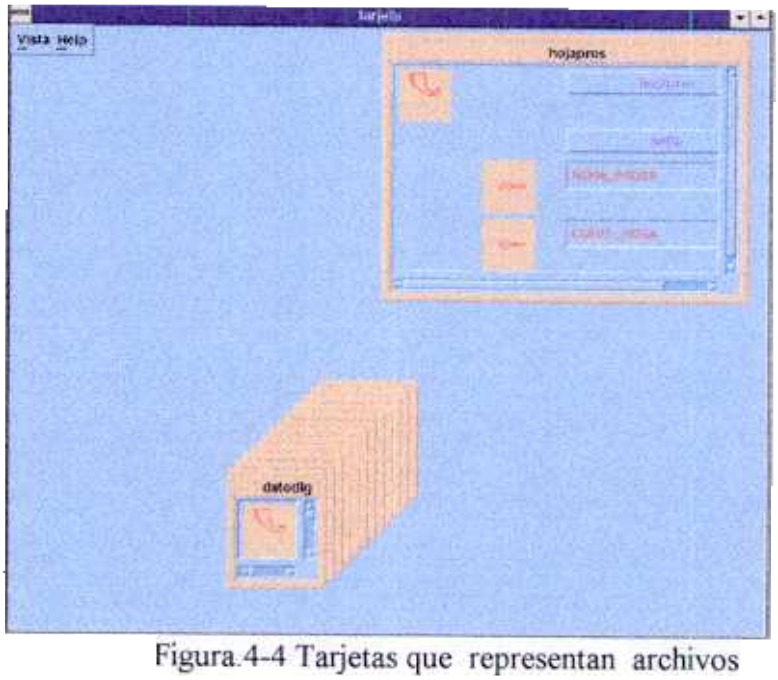


Figura 4-4 Tarjetas que representan archivos

Cuando el usuario quiere hacer las operaciones sobre el archivo, basta con seleccionar el botón que contiene su nombre y se desplegará automáticamente una ventana llamada

"Actualiza y Consulta" en la que se pueden realizar las operaciones del DML sobre el archivo seleccionado. Ver la figura 4-5.

Figura 4-5 Tarjeta: "Actualiza y Consulta" para actuar sobre el archivo

La tarjeta de "Actualiza y Consulta" tiene dos áreas: la primera sirve para la captura y el despliegue de los atributos del archivo; la segunda, tiene una serie de botones, que llevan a cabo las operaciones de: **alta, baja , búsqueda**,...

Se pueden abrir varios archivos simultáneamente. En otras palabras, simultáneamente se pueden activar varias tarjetas.En la figura 4-6 están activadas las tarjeta de "pozos" y "hojapros". Cada tarjeta tiene asociada su propia tarjeta de "Actualiza y Consulta".

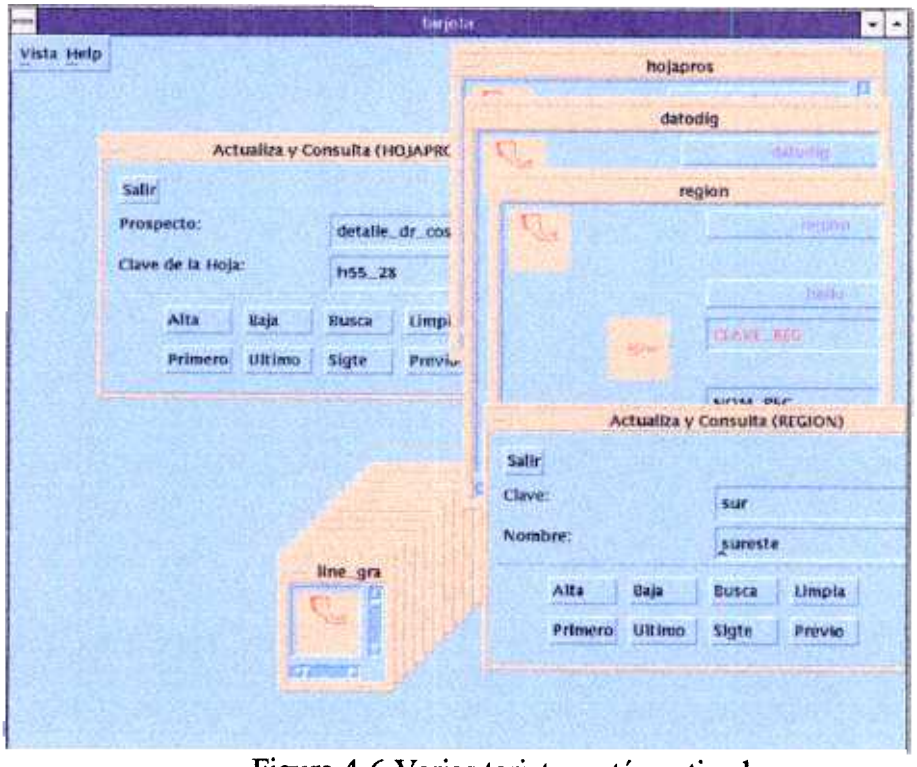


Figura 4-6 Varias tarjetas están activadas

4.2 INTERPRETE DE LOS COMANDOS DEL DML

4.2.1 Agregando un nuevo registro.

(1). Programación en DML.
Cuando se quiere agregar nueva información a la base de datos; por ejemplo, cuando el usuario quiere dar de alta la información de un nuevo pozo, abre la tarjeta de "Actualización y Consulta" de pozos. En los espacios correspondientes se escribe la información que se quiere añadir .Ver figura 4-5.

Después se oprime el botón "Alta". En la tarjeta correspondiente al archivo de "Pozos" un nuevo registro se adicionó. Si se desea agregar más información a la base de datos se repite la operación.

(2). Interpretación del comando "ALTA" de DML.
Cuando el usuario oprime el botón "Alta"en el DML, el sistema llama al intérprete del DML. El interprete utiliza la función "add_rcd()" (en el programa database.c) la cual recibe como parámetros el número del archivo y el apuntador al buffer que contiene el nuevo registro, se verifica por medio de "relate_rcd()" si algún valor del registro está relacionado con otros archivos de la base de datos y si este no viola la consistencia, por medio de la función "new_record()" se agrega fisicamente el registro del archivo de datos.

```
/*----------agrega un registro a un archivo ----------*/
```

```
int add_rcd(f,bf)
int f;
char *bf;

 if((rtn=relate_rcd(f,bf)!=ERROR)  {
        ad=new_record(curr_fd[f],bf);
        if((rrtn=add_indexes(f,bf,ad)==ERROR)
          errno=D_DUPL;
          delete_record(curr_fd[f],ad);
```

La función "new_record()" (en datafile.c y el código mostrado abajo) se verifica si hay espacio disponible en la lista de registros borrados (if(fh[fp].first_record)), si es así se usa para el nuevo registro. En caso contrario, el registro se agrega al final del archivo utilizando la variable next_record y la función "put_record()", y por último se actualizan las variables del encabezado del archivo (en fh[fp]).

```
/*-----------------------crea un nuevo registro-
RPTR new_record(fp,bf)
int fp;
char*bf;

    if(fh[fp].first_record)  {
        redno=fh [fp}.first_record;

        get_record(fp,redno,c);
        fh[fp].first_record=c->next_record;

}else
       redno=fh[fp].next_record++;
put_record(fp,redno,bf);
return redno;
```

4.2.2 Búsqueda de un registro

Para datos nominales, el acceso a los registros en el archivo es en forma aleatoria(directa). Con el acceso aleatorio se puede recuperar un registro directamente en cualquier posición sin tener que recorrer el archivo.

Programación en DML.

Cuando se quiere recuperar un registro del archivo que está abierto con la tarjeta de "Actualiza y consulta", se capturan los campos que correspondan a la llave del registro en el espacio de la presentación de la tarjeta. Si el registro tiene varios campos que son llaves, nada más se le da una llave. Por ejemplo, en la figura 4-7 un archivo de "pozo" tiene como llave el campo de nom_pozo. Si el usuario quiere ver información sobre un pozo determinado; por ejemplo, el pozo de "el_plan", en el espacio de nom_pozo se teclea el nombre del pozo, en este caso "el_plan" y luego se activa el botón "Busca".

Figura. 4-7 Se busca la información de pozo "el_plan"

En la tarjeta, se muestra la información completa del pozo llamado "el_plan'

(2). Interprete del comando "Busca'

Las funciones que proporciona el manejador de archivos del sistema para recuperación y borrado utilizan el registro "lógico" (el número de registro) como argumento, el cálculo de posicionamiento ("lógico") se realiza internamente en la función . A continuación se describe el posicionamiento por byte y por registro "lógico".

A la acción de mover el apuntador a otra posición en el archivo, se le llama posicionamiento (seeking). El posicionamiento por byte requiere del nombre lógico del archivo así como de la nueva posición del apuntador de lectura_escritura, es decir, el número de bytes que deben ser saltados desde el inicio del archivo. La forma que tiene esta operación es:

SEEK (Archivo-lógico, posición)

En las funciones de librería de entrada/salida en 'C' estándar, un registro en el archivo es direccionado por la localización del primer byte del registro, relativo al primer byte del archivo (inicio del archivo). Por ejemplo, si los registros tienen una longitud de 75 bytes, y se requiere el tercer registro, este se localizará en la posición del carácter 150. El primer registro está en la posición 0, el número dos está a 75 bytes de distancia o sea en la posición 75.

Por lo anterior se puede ver que la posición de un registro en el archivo es un múltiplo de 75 (según el valor del ejemplo), y para pasar de un registro a otro se llevan a cabo saltos de longitud constante. El parámetro de registros de longitud fija da como resultado que el archivo de datos se vea como una colección de registros "lógicos"(registro #1, registro #2,..) más bien que de caracteres. Bajo el concepto de registro lógico, la dirección de un registro se considera un entero (número de registro) relativo a la posición que el registro tiene en el archivo. El primer registro es el número 1, el segundo registro es el número 2, y así sucesivamente. El cálculo del posicionamiento "lógico" es realizado mediante las funciones del SMBD, por ejemplo en la recuperación de un registro de la base de datos.

```
/*----------------------toma un registro de un archivo--
static int getrcd(f,ad,bf)
int f;
RPTR ad;
char *bf;

 get_record(curr_fd [f],ad,bf);
 curr_a[f]=ad;
 return OK;
```

La función "getrcd()" (en database.c) llama a la función "get_record"(en datafile.c), esta última recibe como argumentos:
a) el número del archivo.
b) el número del registro a recuperar y,
c) el buffer (bf) donde se va a depositar el registro a recuperar.

```
#define flocate(r,1)   ((long)(sizeof(FHEADER)+((r)-1r()*(1))))
static int handle [MXFILS]; /*MXFILS en cdata. Máximo # de archivos en la BD*/
FHEADER fh [MXFILS];   /*encabezados de los archivos de la base de datos     */
/*-------------------recupera un registro-------------------------*/
int get_record(fp,redno,bf)
int fp;                    /*número de archivo         */
RPTR redno;               /*número de  registro        */
char *bf;                  /*buffer para almacenar el registro */

 if(redno>=fh [fp].next_record)
         return ERROR;
 Iseek(handle [fp], flocate(redno,fh[fp]. record_legthn),0);
 readl(hadlen [fp],bf,fh [fp]. record_length);
 return OK;
```

La posición del primer carácter de un registro en el archivo para su recuperación lógica se calcula con la macro "flocate()" dentro de la función. La función se apoya en "handle[fp]" que mantiene los apuntadores de los archivos abiertos de la base de datos y "fh[fp]" que mantiene los encabezados de los archivos. Por último, la función "getrcd()" actualiza "curr_a[fp]" que mantiene la dirección del último registro accesado para cada archivo de la base de datos.

4.2.3 Eliminar un registro de un archivo.

En la tarjeta de "Actualización y Consulta" ,existen dos opciones muy similares:"baja" y "limpia", pero que tienen diferentes funciones. mientras "baja" sirve para eliminar la información anterior, ó sea, eliminar un registro de un archivo; "limpia" se utiliza para limpiar las presentaciones de la tarjeta. En otras palabras éste comando sólo elimina la información en la pantalla, nunca la información ya almacenada en la base de datos.

Si por alguna razón es necesario borrar algún registro en el archivo; primero se debe mostrar el registro en la pantalla a eliminar, después se activa el botón "baja" y así el registro queda eliminado.

El borrado de un registro de un archivo presenta ciertas consideraciones. Si por alguna razón es necesario borrar algunos registros en el archivo esto genera espacios libres en disco sin usar, y si al agregar nuevos registros no se re-utilíza el espacio libre dejado por los registros borrados, el archivo crecería enormemente, por lo tanto, es necesario re-usar el espacio.

Cualquier estrategia de eliminación permite reconocer los registros borrados. Por ejemplo un método simple consiste en colocar una marca especial al inicio del registro suprimido. Ahora el problema es la re-utilización del espacio.

Para la recuperación de espacio, algunos sistemas proporcionan como herramienta un algoritmo de compactación. Estos algoritmos reconstruyen el archivo dejando fuera todos los registros borrados. La decisión para ejecutar el algoritmo puede basarse en el número de registros borrados o realizarse a periodos regulares.

Como puede verse, la solución antes descrita es poco práctica en sistemas interactivos o que manejan datos muy volátiles (que constantemente se modifican). De aquí la idea de utilizar la reclamación dinámica de espacios libres para almacenamiento. El método usado en el sistema de información geográfica es el de una lista ligada de registros borrados.

Con la función "del_rcd()" se inicia el borrado de un registro, primero se borra la llave (en el archivo de índices "del_indexes()", para posteriormente borrar el registro de datos del archivo de datos "delete_record()".

```
/*----------borrado del actual registro en un archivo-----
int del_rcd(f)
int f;

 if (curr_a[f]) {
        del_indexes(f,curr_a[f]);
        delete_record(curr_fd[f], curr_a[f]);
        curr_a[f]=0;
        return OK;
        }
 errno=D_PRIOR;
```

```
return ERROR;


/*----------------------borrado de un registro----------------------*/
int delete_record(fp,redno)
int fp;
RPTR redno;

 FHEADER*bf;
 extern char*malloc():

 if (redno>fh[fp].next_record)
       return ERROR;
 if (bf=(FHEADER*) malloc (fh[fp].record_length))==NULL) {
       errno=D_OM;
       dberror();

 set_mem(bf,fh[fp].record_length, '\0');
 bf->next_record=fh[fp].first_record;
 bf->first_record=-1;
 fh[fp].first_record=redno;
 put_record(fp,redno,bf);
 free(bf);
 return OK;
```

La lista de borrados se maneja como una pila en donde todas las inserciones y supresiones tienen lugar en el extremo o tope. Cuando un registro es borrado, se utiliza el mismo espacio que ocupaba el registro, grabando en el mismo una estructura con el formato del encabezado del archivo "bf=(FHEADER*) malloc (.....)". Para escribir la marca (o bandera) de borrado "bf->first_record=-1" y el número del registro que encabezaba la lista de borrados antes del borrado "bf->next_record". La bandera puede ser usada por un programa que necesite leer el archivo secuencialmente, y brincar los registros marcados.

4.3 DAR DE ALTA UN ARCHIVO GRÁFICO.

La otra línea corresponde al manejo de las operaciones de los datos gráficos. En esa parte, podemos tener acceso a los datos gráficos por medio del teclado o del digitalizador. Los datos gráficos se agregan en multilistas, usando los archivos con extensiones *.DEF y *.GRA.

Ya que por el momento no tenemos digitalizador conectado con la estación de trabajo, hemos usado un paquete en PC para agregar los datos del digitalizador a los archivos. Por ejemplo H55_28.DEF y H55_28.GRA que conforman las estructuras de datos que describimos en el capitulo 3. Después se usa "FTP" comando de red para pasar los archivos a la estación de trabajo, al mismo tiempo de agregan los nombres, llaves, etc., a los archivos relativos en la parte de datos nominales.

Por ejemplo H55_28.DEF y H55_28.GRA contiene los datos de una carta, se abre la tarjeta de "Actualización y consulta" de "hojapros" y se llena con la información requerida.

Figura.4-8 Se abre la tarjeta "Actualización y Consulta" de Hojapros y Prospecto para meter los datos nominal para generar un nuevo plano base

También se abre la tarjeta de "Actualización y consultas" de "prospecto" y se agregan los datos necesarios (ver figura 4-8). De esta manera se crea una nueva relación entre archivos nominales y gráficos.

En seguida vamos a describir como se genera un plano base y lo que es un prospecto. Un plano base está formado por varias clases de entidad real. Entonces primero definimos la clase de las entidades reales.

4.3.1 Las clases de entidad real

Esta base de datos maneja objetos del mundo real, que en su mayoría son objetos de la superficie de la tierra. Podemos decir que los objetos del mundo real son las entidades de base de datos que requieren una representación o ubicación georeferenciada. Un objeto del mundo real es nombrado de diferentes formas según el observador como se muestra en la figura 4-9. Por ejemplo, el diseñador de la base de datos lo considera como una

archivo de la base de datos lo considera un registro y en la representación del mapa se le considera como un símbolo.

Figura. 4-9 Diferentes puntos de vista sobre un objeto del mundo real en la base de datos.

El archivo de datos gráficos consiste en una serie de registros de entidades, mientras que la carta topográfica que corresponde a lo que es el mapa físicamente consiste en una serie de símbolos de objetos.

Aunque no se siguió un enfoque orientado a objetos, consideramos que los desarrollos de esta naturaleza son sumamente prácticos, ya que las entidades que se encuentran en un mapa o dibujo, se adaptan a las características de un objeto.

El contexto de la base de datos es un conjunto de información y relaciones entre entidades. En él, hay cuatro tipos de entidades geométricas:

* entidad P formada por un punto

* entidad L formada por un línea
* entidad A formada por una área
* entidad V formada por un volumen

Estas entidades tienen las siguientes características: atributos que los describen, relación con otros objetos desde el punto de vista semántico, representación geométrica, tamaño y relación con otros objetos desde el punto de vista espacial.

Entidad P (puntual)

En el caso de exploración petrolera, los objetos son, por ejemplo: pozos y campamentos. La representación geométrica de una entidad puntual tiene el formato como figura 4-10.

Nom_p	x	y	z	parámetros
.. ...	... .	... .	...	...
...	...	...	..	.. ..

Figura 4-10 Los atributos lógicos de almacenamiento de entidad puntual

donde: Nom_p es el nombre de la entidad; x,y,z son las coordenadas tridimensionales en el espacio Euclidiano tipo de coordenadas UTM, y parámetros depende de la necesidad de la entidad.

Entidad L (lineal)

Las entidades lineales son , por ejemplo: ríos, carreteras y líneas sismologías. Se puede representar por un segmento de línea con un par de puntos conectados:

$$P_1P_2$$

De esta manera, la representación de una línea quebrada, un arco o cualquier otra linea la podemos representar usando un conjunto de segmentos, de la siguiente forma:

$$L = (P_1 , P_2 , ..., P_n)$$

Entidad A (área)

Las entidades A, que representan áreas son, por ejemplo: pueblos, lagos y bosques. Para representar la geometría y tamaño de una área se usa como su limite de periferia a un polígono. Este, lo definimos como:

$$A = (P_1 , P_2 , ..., P_n , P_1)$$

donde P_1 , P_2 ,..., P_n, son los vértices del polígonos.

Entidad V (volumen)

Las entidades V, que representan volúmenes son, por ejemplo: yacimientos de petróleo. Para representar este tipo de objetos, usamos una red de triángulos. El conjunto de vértices de esta red tiene la siguiente estructura:

vértices $\{ N_i \ X_i \ Y_i \ Z_i \}$

donde N_i es el número del vértice i

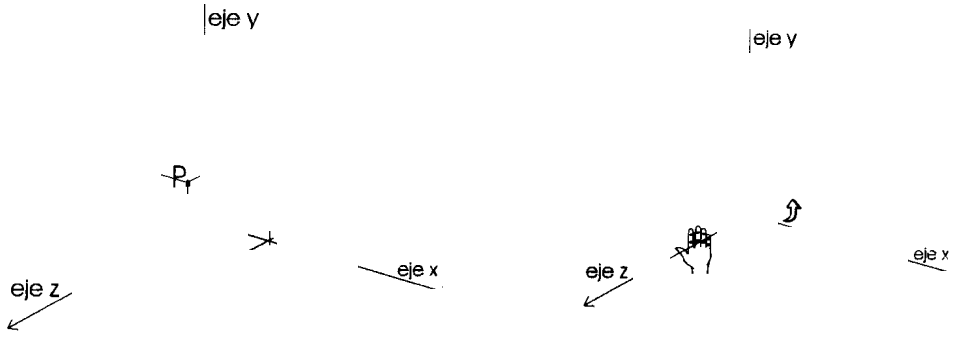
$X_i \ Y_i \ Z_i$ son las coordenadas tridimensionales del vértice

Cada triángulo se representa como:

$\{ N_i , N_j , N_k \}$

donde N_i , N_j , N_k son los vértices que forman al triángulo.

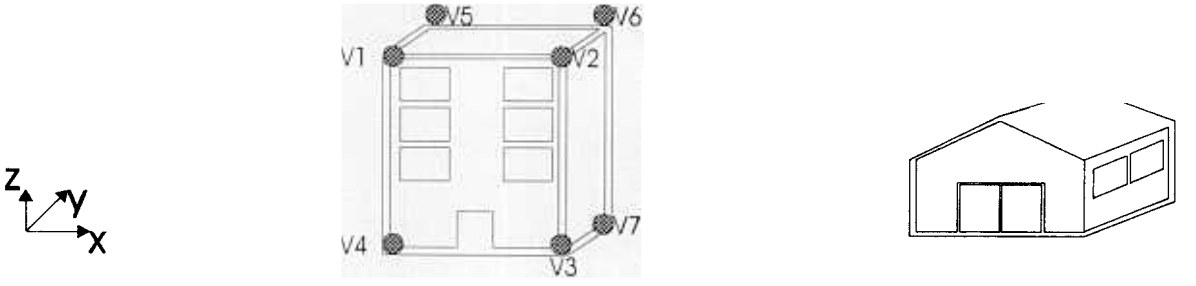
Estos vértices deben especificarse en el sentido contrario a las manecillas del reloj con respecto al punto de vista del observador como se ilustra en la figura 4-11(b).



(a) (b)

Figura 4-11

En dicho sistema, el usuario puede diseñar de cualquier forma la presentación de la base con diferentes puntos de coordenadas. En la figura 4-12(a) se muestran un ejemplo de modelos y sus archivos correspondientes con las coordenadas correspondientes a los vértices.



(a) (b)

Figura 4-12 Objetos de mundo real

Tabla 4-1 Coordenadas de vértices

V1	0	0	1
V2	1	0	1
V3	1	0	0
v4	0	0	0
V5	0	1	1
V6	1	1	1
V7	1	1	0
V8	0	1	0

Tabla4-2 Tabla de triángulos

s1	v1	v3	v2
s2	v3	v1	v4
s3	v1	v2	v5
s4	v2	v3	v6
s5	v6	v3	v7
s6	v5	v4	v1
s7	v5	v8	v4
s8	v5	v6	v7
s9	v5	v7	v8
s10	v5	v2	v6

En la tabla de coordenadas de vértices, se guardan los coordenadas (x,y,z) de los vértices del objeto. Por ejemplo, el objeto que se muestra en la figura 4-12(a) consiste de ocho vértices. La coordenada del vértice 1 es **V1(0,0,1)**; vértice 2 es **V2(1,0,1)** como se muestra en la tabla 4-1. Los tres puntos cercanos pueden definir un plano triangular. En la tabla de triángulos, se guarda la información de cuales vértices construyen un plano triangular. Como se muestra en la tabla 4-2, los vértices **V1,V3,V2** forman el triángulo **s1**. Entonces, para representar una entidad volúmen podemos simplemente usar un conjunto de puntos relacionados.

4.3.2 Como definir una nueva entidad?

En el sistema, hay tres tipos de diccionarios de tarjetas: diccionario de tarjetas de entidades reales, diccionario de tarjetas de entidades conceptuales y diccionario de tarjetas de procesos. Los tres diccionarios se estructuran en una tabla 4-1 con un encabezado que se determina enseguida:

tabla 4-1 Estructura del diccionario de tarjetas:

No. de tarjeta	Apuntador al Bitmap para la representación de la tarjeta:Xi	Nombre de la tarjeta: Xm	Descriptor de funcionalidad:Xf
. .	. .	. .	. .

El número de tarjeta y el descriptor de funcionalidad son usados para control interno, el apuntador al Bitmap y nombre de la tarjeta proporcionan la información para la representación de la entidad en la interfaz de usuario.

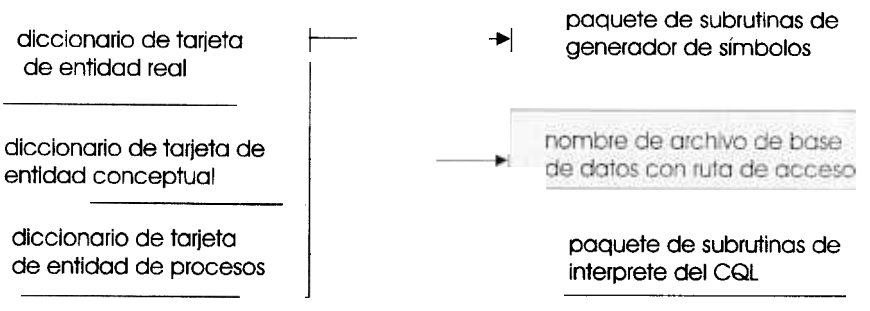
Los números de tarjeta se asignan automáticamente con el formato siguiente:

Se usa "1_*" para las tarjetas de entidad real, por ejemplo: 1_1 para POZO, 1_2 para RÍO.
Se usa "2_*" para las tarjetas de entidad conceptual, por ejemplo: 2_1 para REGIÓN, 2_2 para HOJAPROS.
Se usa "3_*" para las tarjetas de procesos, digamos:3_1 para OPEN, 3_2 para INSIDE OF..

El descriptor funcional Xf depende del tipo de diccionario. Para un diccionario de tarjetas de entidades reales, Xf corresponde a una subrutina que dibuja el símbolo de la entidad.

En el caso del diccionario de tarjetas de entidades de proceso, Xf es la invocación a una función predefinida del interprete del CQL (como se explica en sección 5.5 del capítulo 5).

En el caso de las tarjetas de entidades conceptuales, Xf es un nombre de archivo de base de datos incluyendo su ruta de acceso.



Debido a que las tarjetas de entidad conceptual y entidad de procesos representan la parte de núcleo de la base de datos si se necesita añadir una nueva entidad, es necesario que el diseñador de la base de datos haga los cambios necesarios, en base al modelo conceptual de la base de datos. No se permite los usuarios finales a cambiarlo.

La tarjeta de entidad real es de una parte de aplicación de la base de datos. Los usuarios finales pueden añadir una nueva tarjeta de entidad real según su necesidad. Debido a que el sistema VISDA está orientado a la exploración petrolera, la mayoría de las entidades ya están definidas. Pero en el caso de necesitar una nueva, el usuario la puede definir mediante un proceso determinado por el diagrama según figura 4-13.

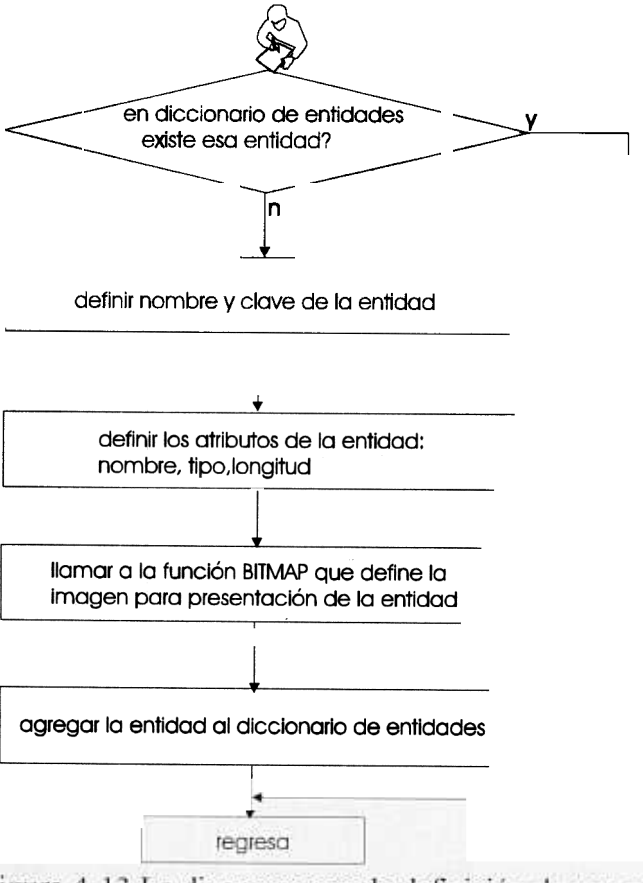


Figura 4-13 La diagrama para la definición de una nueva entidad

Primero, se verifica la existencia de la entidad en el diccionario. Si existe, termina el algoritmo. En el caso contrario, pide: el nombre, la llave y las características de los atributos de la entidad; tales como el nombre, el tipo y la longitud.

En el sistema BITMAP, para definir la imagen Xi de la nueva entidad. La ventana de BITMAP se ve como en la figura 4-14. El usuario puede diseñar interactivamente la imagen de la entidad de cualguier forma sobre una ventana. En el caso del proyecto que

maneja los objetos de topográficos, usamos la misma forma de los símbolos de ingeniería civil, como se muestra en la figura 4-14, el símbolo que representa la entidad "POZOS".

Al final, se agrega la entidad nueva al diccionario. El número de tarjeta se incrementa automáticamente dependiendo de su tipo.

La representación de la imagen Xi, es diseñada por el usuario, de la misma manera que su nombre Xm.

La parte Xf de la tarjeta de entidad real, es una función que toma la imagen Xi y la coloca en el lugar apropiado según las posiciones que tengan en el mapa.

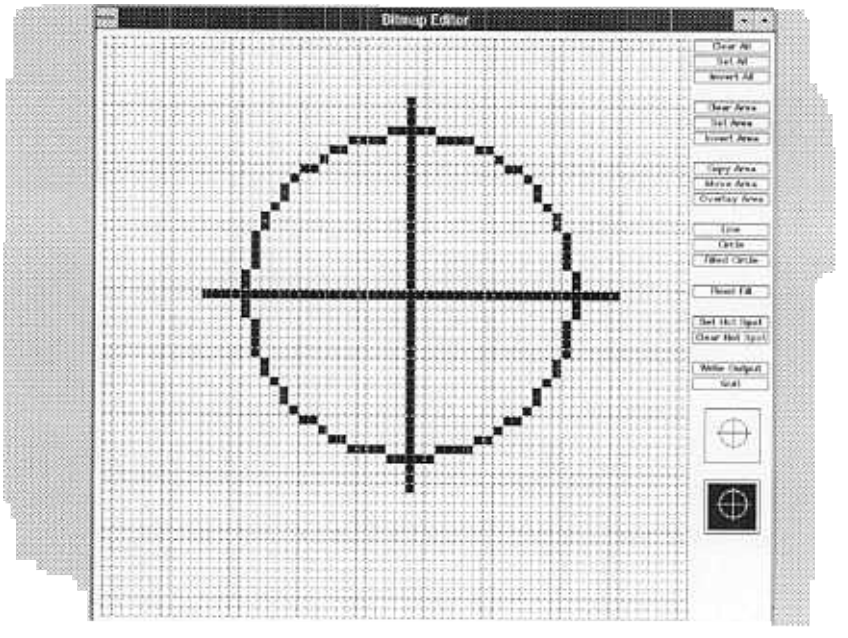


Figura 4-14 El usuario interactivamente diseña
la imagen de la entidad con BITMAP

4.3.3 Procedimientos para generar un plano base en el sistema

Regularmente, en los sistemas de información geográfica, existen diferentes dispositivos de entrada para actualizar la base de datos, como son: digitalizadores, scanners, captadores de imágenes de satélite, pantallas de captura, entre otros; para este sistema se utilizo un digitalizador. Este dispositivo nos permite tener la información vectorizada de tal manera que podemos manejar los atributos inherentes a las diferentes clases de entidades que existen en la hoja y también manipular los datos como puntos, líneas y polígonos.

Un plano de base está formado por diferentes clases de entidades (tipos de datos) como son carreteras, y ríos, también cada clase o cierto tipo de datos a su vez puede estar compuesta por uno o varios identificadores (id_tipo); es decir, en la clase de ríos podemos tener el río Usumacinta, Lerma, y Pánuco.

Como ejemplo de estas hojas están H55_28, y H55_29 según clasificación del cartas 1:50000 del INEGI. Cada hoja (plano base) está formada por dos archivos uno de definición 'H55_28.DEF', y otro de datos 'H55_28.GRA'. Las estructuras usadas para el manejo de la información son listas que apuntan a otras sublistas.

Para emplear el digitalizador (simulador), seleccionamos del menú horizontal CAPTURA-GRAFICAS y del menú vertical DIGITALIZADOR. Posteriormente se despliega una pantalla de captura en la que se proporciona la siguiente información:

Nombre del plano: Clave de la hoja o del plano, ya dada de alta anteriormente.
Escala del plano: Escala de la hoja o plano a digitalizar, en particular 1:50000.

Coord **x** izq inf,Coord **y** izq inf, Coord **x** der sup, Coord **y** der sup son las coordenadas extremas de la hoja o plano a digitalizar, estas son importantes ya que quedan grabadas en el archivo de definición del plano. Además el sistema los usa en el módulo de graficación cuando se va a desplegar este plano en pantalla.

(1) El caso que tipo de plano base es "DETALLE"

Después de haber pulsado F2, se despliega otra pantalla de captura, donde pide

Tipo de datos a digitalizar. Se refiere al nombre de la clase de entidad que se va a digitalizar, ejemplo: CARRETERAS, RIOS.

Identificador del tipo. Nombre que identifica exactamente a la entidad, es decir el identificador del RÍO o de la CARRETERA. (por ejemplo, el río Usumacinta). Luego se pulsa F6-DIGITALIZADA. Posteriormente se define la acción a seguir, de acuerdo a la tecla que se pulse:

S-> Salir o terminar. Esto nos indica que hemos terminado la sesión.

C-> Continuar. Nos permite definir un nuevo origen de la misma entidad que se esta digitalizando. Por ejemplo, un río con afluentes.

Enter o ***Return***. Para iniciar o continuar la digitalización.

Entonces, al iniciar pulsamos ***Enter,*** y después de cada punto de sus coordenadas, también ***Enter***.

Este proceso es cíclico hasta que terminamos pulsando "***S***". Antes de terminar, o de pulsar (S), se da un punto extra que es la posición donde queremos que se coloque el título de la entidad sobre el plano.

Inmediatamente se despliega la pantalla de captura que define la clase, si todo estuvo correcto se salva la información pulsando F2. Después de haber salvado se puede continuar con la misma clase, pero con otro identificador o cambiar la clase y el identificador lo que implica repetir el proceso "DIGITALIZA" (o tecla F6). Si no quiere continuar salve y pulse tecla ESC para finalizar.

(2) El caso que el tipo de plano base es "ESTANDAR"

Hasta ahora, hemos hablado de Hojas o Planos Base de 'DETALLE', pero también hay datos que son propios del estudio y no de las hojas, como es el caso de las líneas sismológicas, pozos, gasoductos, líneas de distribución eléctricas, retícula, y otros Esta información se guarda en dos archivos: el de definición y el de datos, empleando para ellos estructuras en forma de LISTAS. A esta información le llamamos 'ESTANDAR' ya que las clases no tienen identificadores internos, solo es un "tipo de dato".

Tipo de datos. Se refiere al nombre de la línea sismológica. Como observación del programa "meto_geo.c" está rígido con respecto a que el identificador de la línea sismológica comience con "P_", esto es para localizarlo en las LISTAS y la dibuje;

La información del nodo es un número que identifica ese punto, las coordenadas x, y, z del punto; los datos z y la elevación son opcionales, y se emplean para darle atributos al punto. Debes de salvar cada punto y al final también la lista, como se indica en el menú.

Por medio del digitalizador se usa el mismo procedimiento que se hace para generar una hoja o un plano base, solo que los datos no son de DETALLE son ESTANDAR.

4.3.4 Cómo relacionar los planos bases?

Mediante este procedimiento podemos dar de alta todas las hojas de un prospecto, área o región. Cada hoja puede ser un plano base como se muestra en la figura 4-15.

También podemos pensar que estas hojas son "cuadras o bloques" de una ciudad y que varias de estas "cuadras" forman un prospecto que va estar sujeto a un estudio. Entonces las clases que estas hojas pueden contener otra capa de entidades geográficas como puede ser: el Sistema de Drenaje, sistema de agua potable, distribución de gas o distribución telefónica, entre otras.

Para nuestro estudio, un prospecto es una porción de un área contenida en una región. Este prospecto puede estar contenido en una o varias hojas cartográficas.

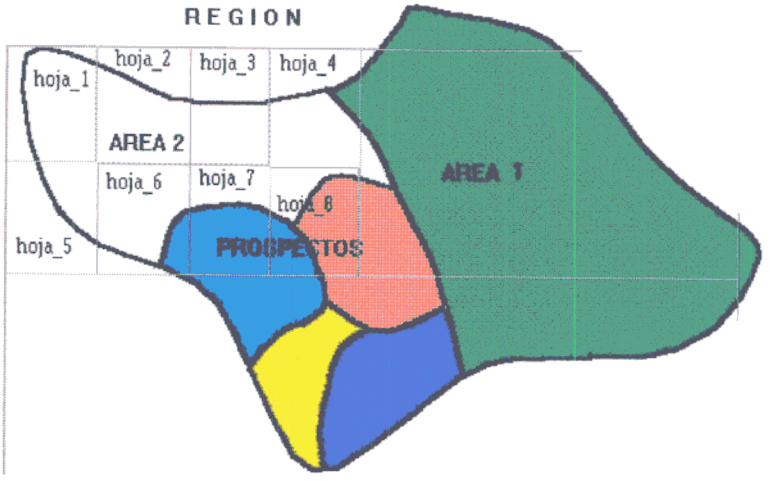


figura 4-15 Una región se puede dividir en varias áreas y una área se puede dividir en varios proyectos y un proyecto se divide en varios hojas.

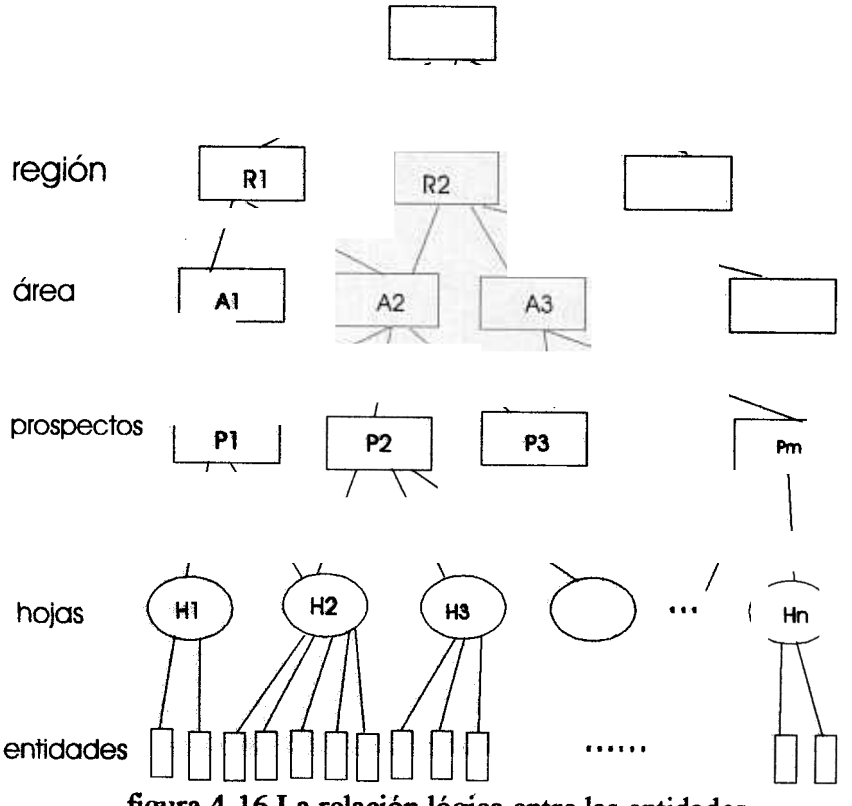


figura 4-16 La relación lógica entre las entidades

Antes de relacionar la(s) hoja(s) que pertenecen a un prospecto, se tuvo que haber empleado el tarjeta de "Actualización y Consulta" que se muestra en la figura 4-8 para dar de alta a la Región, al Área y al Prospecto en donde ahora si relacionamos las hojas que cubren o abarcan el prospecto. Al final en la base de datos se forma una relación lógica entre las entidades como se muestra en figura 4-16.

4.3.5 Algoritmo para manipulación de los datos gráficos.

Sabemos que las estructuras de los datos espaciales que se usan son "multilistas". Cuando hacemos las operaciones de los datos gráficos, se llaman a las funciones correspondientes. A continuación describimos algunos algoritmos de estas funciones.

1) Algoritmo para dar de alta una lista.

Para dar de alta una lista se carga en memoria el contenido del archivo de definición y se procede a buscar el identificador en el arreglo de estructuras de definición, si existe uno con el mismo nombre la alta no ocurre, en caso contrario se lleva a cabo el siguiente procedimiento:

paso 1.- Se asigna en la estructura de definición el espacio que apunta al apuntador LIBRES, se copia el identificador en el elemento **id_nombre** y se actualiza **libres**.

paso 2.- Se carga la lista con el procedimiento de inserción.

paso 3.- En el espacio asignado (en la estructura de definición) se copia en el elemento num_recs el número de registros de la lista.

paso 4.- Agregar al final del archivo de datos la nueva lista localizando esta posición sumando el número de registros de las listas anteriores a la nueva lista y multiplicando este resultado por la longitud de los registros.

2) Algoritmo para borrar una lista.

El borrado de una lista requiere de una serie de pasos para actualizar los archivos de definición y de datos:

paso 1.- Cargar en memoria el contenido del archivo de definición y buscar por su identificador el número de lista a borrar.

paso 2.- Si la lista a borrar se encuentra al final del archivo, ir al paso 3, en caso contrario (la lista a borrar no es la última) mover el contenido del archivo de datos gráficos a partir del primer registro de la siguiente lista a la posición que ocupa el primer registro de la lista que se esta borrando, ir al paso 3.

paso 3.- Mover un espacio (del tamaño de una estructura) el contenido de las estructuras de definición de las listas hacia la estructura que le corresponde a la lista a borrar, este movimiento de información es a partir de la estructura siguiente a la que es borrada y se actualiza el elemento siguiente. Termina procedimiento de borrado.

CONCLUSIONES

En este capitulo se muestra la visualización de archivos, atributos y lenguaje DML, el cual es un lenguaje de manipulación de datos que apoya el manejo o procesamiento de los objetos de la base de datos usando una filosofía de visualización. En esta parte, se explica como creamos una ambiente visual para las operaciones de alta , baja y búsqueda, sobre la base de datos. Y se describe como se visualiza de la estructura de archivos, registros y la base de datos. El usuario puede saber la información completa de la base de datos: el diccionario de datos, los archivos de la base de datos, los campos de cada archivo, longitud de cada atributo y llave de cada archivo, entre otras cosas.

CAPÍTULO 5

EL LENGUAJE VISUAL DE CONSULTA CQL

El campo de las bases de datos es una de las áreas de aplicación más prometedoras de los lenguajes visuales por muchas razones. Una razón es que existe una demanda social para el uso y la expansión de las bases de datos, por lo tanto mucha gente ha tenido oportunidades de manipular sistemas de base de datos por sí mismo. La manipulación de los sistemas de base de datos, sin embargo, pueden crear a veces problemas para el usuario, especialmente para el usuario ocasional no familiarizado con sistemas de este tipo.

Un objetivo del proyecto fue el proporer y crear un nuevo lenguaje visual que pudiera recuperar y entender la información contenida en una base de datos espacial con enfoque geográfico. En este capítulo iniciamos dando un panorama general de estos lenguajes, para particularizar en un lenguaje visual de consulta. En seguida define formalmente junto con la interfaz visual de usuario bajo la cual trabaja.

5.1 LA CLASIFICACIÓN DE LOS LENGUAJES VISUALES

5.1.1 Los tipos de lenguaje visual

El "lenguaje visual" tiene dos formas de despliegue de los elementos operacionales: uno es lineal, es decir, como un lenguaje normal con caracteres que representan a los elementos operacionales. Y otro es aquel, en el que el lenguaje mismo es visual , es decir, un lenguaje de programación con expresiones visuales.

Los objetos manejados por el lenguaje también se pueden dividir en dos tipos: en el primero, los objetos tienen una representación visual inherente. Y en el segundo no, por ejemplo, los arreglos, pilas y archivos.

Con la combinación de las dos formas y los dos tipos podemos clasificar el lenguaje visual en cuatro clases [CHAN 90] como se muestra en tabla 5-1.

En la clase 1, se describen los lenguajes que usan las expresiones lineales (nominales), que manejan los objetos que no tienen una representación visual inherente y usan las representaciones visuales según sus significados lógicos. Esta clase se denomina como los lenguajes que soportan interacción visual.

En la clase 2, los lenguajes que usan la expresión visual, paro manejan los objetos que no tienen una representación visual inherente y usan la representaciones visuales según sus significaciones lógicas.

Tabla 5-1 Los cuatro tipos de lenguajes visuales

Clase	Objetos que se manejan	Despliegue de los elementos operacionales del lenguaje	Característica del lenguaje	Transformación de los objetos
1) Lenguaje que soportan interacción visual	objetos lógicos con representación visual	lineal (normal con caracteres)	lenguajes que soportan interacción visual	$(X_m,e) \rightarrow (X_m,X_i')$
2) Lenguajes visuales de programación	objetos lógicos con representación visual	visual	lenguajes con programación visual	$(X_m,e) \rightarrow (X_m,X_i')$
3) Lenguajes que procesan información visual	objetos visuales con una representación lógica sobrepuesta	lineal	lenguajes para el procesamiento de la información visual	$(e, X_i) \rightarrow (X'_m, X_i)$
4) Lenguaje que procesan información visual icónico	objetos visuales con una representación lógica sobrepuesta	visual	lenguaje para el procesamiento de la informa-ción visual usando iconos	$(e, X_i) \rightarrow (X'_m, X_i)$

En la clase 3, los lenguajes usan las expresiones de caracteres y manejan los objetos que tienen representación visual inherente. Esta clase se clasifica como lenguaje para el procesamiento de la información visual.

En la clase 4, los lenguajes usan las expresiones visuales, por ejemplo, iconos, y manejan los objetos que tienen representación visual inherente.

Como se muestra en la tabla, las cuatro clases de lenguajes son diferentes entre sí, pero manejan el concepto visual general que la unifica.

5.1.2 La evaluación de lenguaje visual

En la introducción y el estudio de lenguajes visuales de Nan C. Shu [SHU 92], se propone un esqueleto de tres dimensiones para caracterizar y comparar los lenguajes visuales, figura 5-1. Al evaluar un lenguaje, se plantean tres cuestiones:

* **Descriptividad. ¿Es suficientemente descriptivo?**
* **Completitud. ¿Qué tan completo es como un lenguaje?**
* **Generalidad. ¿Qué tan general son sus aplicaciones?**

Los tres preguntas determinan tres dimensiones de un esqueleto que permite caracterizar y clasificar los lenguajes visuales según la figura5-1. El eje "extensión visual", evalúa si el lenguaje visual tiene o no suficiente visualización. El eje "nivel del lenguaje" evalúa si el lenguaje visual cubre un rango de nivel para la representación de los procesos del sistema, es decir, desde el punto de vista de un lenguaje si es o no suficiente para la representación de los procesos del sistema. En el último eje se puede escalar el "área" con el fin de evaluar el lenguaje desde el punto de vista de generalidad y su amplitud de aplicación en areas. Por ejemplo, **Xerox Star System** es alto en extensión visual, pero bajo en las otras dos dimensiones. Por otro lado, **Query-by-Example** es bajo en extensión visual y área de aplicación, pero alto en nivel de lenguaje. **LIDA** [CHAP91] es alto en extensión visual y en nivel de lenguaje, pero su aplicabilidad es especifica a base de datos. **PegaSys**[MORI 85], **HI-Visual**[HIRA90] (Hiroshima - Visual) es alto en extensión visual.

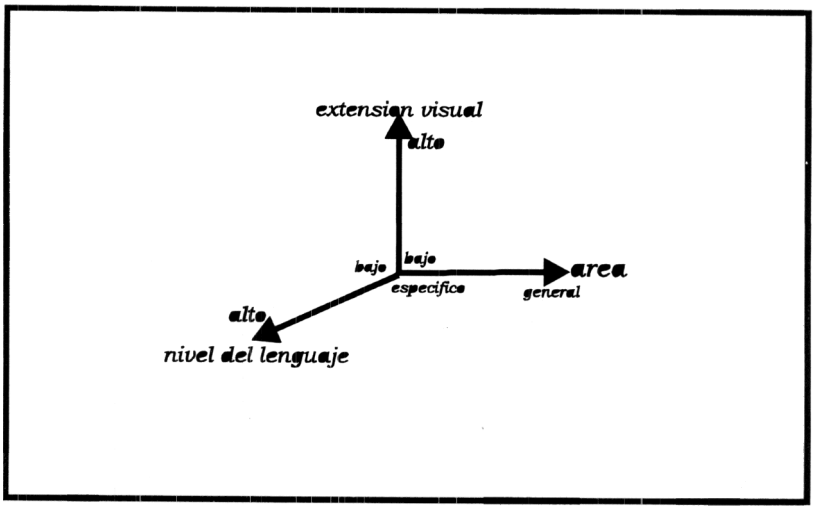


Figura 5-1 Un esqueleto de tridimensiones para caracterizar y comparar a los lenguajes visuales

En este proyecto nos enfocamos en el diseño de un lenguaje de consulta CQL con alto nivel de extensión visual, el nivel del lenguaje es equivalente con SQL, y se aplica a bases de datos espaciales. El lenguaje está basando en las siguientes ideas:

1) Emplear un ambiente gráfico de visualización.
2) Representar a las entidades y a las acciones usando tarjetas sobre la base de datos.
3) Ampliar la construcción gráfica del lenguaje: objeto temporal, a fin de facilitar la expresión de una clase de operación llamada sub-consulta, como almacenamiento temporal de información.

En nuestra base de datos, generalmente existen dos tipos de consulta para localizar una entidad :

* Recuperación de entidades aisladas ignorando las relaciones espaciales que existen entre ellas.
* Recuperación de entidades considerando las relaciones que existen entre los pares formados por ellas.

Recuperación por entidad. se usan diferentes objetos específicos como llaves para representar el conjunto completo de los objetos contenidos en la base de datos. El procesamiento de una consulta es equivalente a la recuperación por un matching parcial que se lleva a cabo en las bases de datos convencionales.

Recuperación por relaciones entre entidades. Para procesar este tipo de consultas existe un método que consiste en grabar cualquier combinación de relaciones espaciales entre cada par de objetos y realizar la búsqueda por un matching parcial.

A fin de tomar en cuenta las características de una consulta en un lenguaje visual y de los datos geográficos, nosotros introducimos varias extensiones a los modelos de una base de datos clásica (las tarjetas, las variables, la superposición de la representación espacial). Para ilustrar los conceptos del lenguaje, proveemos la representación de varias consultas (involucrando un objeto único con el operador, dos o más objetos vinculados con un operador espacial, todos los objetos vinculados con un operador espacial).

5.2 EXPANSIÓN DE ICONOS

Actualmente, los lenguajes visuales usan una forma de icono. Una de las razones por las que el uso de iconos ha resurgido rápidamente, se debe a la facilidad por parte del hombre para capturar un mensaje basado en ellos, ya que la mente, cuando procesa imágenes, infiere relaciones escondidas sin necesidad de incluir texto en estas. Además, existen otras razones que nos invitan a iniciar la utilización de elementos visuales dentro de los medios ambientes de trabajo y desarrollo. Antes de la definición al lenguaje, primero estudiaremos la idea de icono. Se ha dicho que un lenguaje de consulta, será visual cuando la semántica de la consulta se exprese a través de un dibujo. También se ha dicho que será declarativo cuando la consulta especifique las propiedades para ser verificadas por el resultado, pero no la manera de obtenerlas.

5.2.1 iconos generalizados

Icono es una imagen, representación o ilustración, utilizado para representar un concepto, idea, dato u operación. En computación, un icono es una representación de dos partes de un objeto, su imagen y una acción o un proceso computacional.

Un icono es denotado por ($\mathbf{X_m}$, $\mathbf{X_i}$) [CHAN90], donde X_m es la parte lógica (el significado o nombre del objeto) y X_i la parte física (la imagen o su representación). Por ejemplo, en los lenguajes de programación visual (tabla 5-1), los objetos son de la forma **(X_m,e),** es decir, objetos con un significado lógico pero sin una representación visual (**e** = empty). Posteriormente, la representación visual se sobrepone, es decir, el icono se transforma de **(X_m,e)** en ($\mathbf{X_m}$, $\mathbf{X'_i}$) de tal forma que pueda ser visualizado. Los objetos manejados por el lenguaje visual pueden ser entonces considerados como iconos con una parte lógica y una parte física, que representan a un objeto.

Similarmente, las operaciones del lenguaje pueden ser consideradas como iconos de proceso con una parte lógica y una física que representan una operación computacional.

5.2.2 TARJETA -- expansión de icono

Los iconos son más poderosos que las palabras como un medio de comunicación y pueden ayudar a entender y recordar, sin embargo su semántica puede estar determinada por su representación, contenido o abstracción, aspectos que pueden llevar a significados mas complejo.

En nuestro sistema, la mayoría de los elementos pictóricos, pueden ser contemplados a través de su propio significado, imagen, parámetros y características.

Definición 5-1: TARJETA (O CARD)

Es un símbolo pictórico,que describe una entidad primitiva, una acción o un proceso computacional, y es representado por (Xm,Xi, Xt, Xp).

donde Xm es el significado del elemento (nombre),
Xi es la imagen del elemento (representación),
Xt es el tipo del elemento, PROCESO u OBJETO,
Xp es el parámetro del elemento

Usamos tarjetas para representar objetos concretos a fin de facilitar el reconocimiento del significado de objetos y operaciones. Generalmente, la función reconocida de un objeto del ambiente en que el objeto sea empleado. Por lo tanto, las tarjetas deben diseñarse tomando en cuenta estas consideraciones.

En nuestro caso, donde la base de datos se aplica a la exploración petrolera, el diccionario de tarjetas CD, intenta representar tres tipo de entidades: **Tu** (entidades reales), **Ts1** (entidades conceptuales) y **Ts2** (entidades para las acciones y procesos).

Tu

incluye las entidades reales que tienen una representación visual inherente - - objetos pictóricos que se asocian con una interpretación lógica cierta. En el sistema, esas entidades son exactamente los datos espaciales de la base de datos, los cuales son objetos reales de la superficie de la tierra(definidos en el capítulo 4). Por ejemplo: carreteras, ríos, caminos, gasolineras, casas.

Ts1

consiste de las entidades conceptuales que no tienen una representación visual inherente. En este caso se considera iconos el cual tiene un significado abstracto. Esto incluye tipos de datos tradicionales y aplicación de tipos de datos tales como documentos y bases de datos. Nosotros llamamos a las entidades de este tipo como objetos conceptuales, por ejemplo, las entidades: **región, área, proyecto, cinta y brigada**. Para cada uno existe una tarjeta en la base de datos como los definido en el capítulo 2.

Ts2

incluye las entidades que representan las acciones o procesos computacionales del sistema. Por ejemplo, **dentro de, arriba de , abrir**(se va a definir en la sección 5.4).

El objetivo del diseño de las tarjetas es facilidar la comprensión del significado funcional, aprovechando las bondades que ofrece el dibujo del objeto para contener mas información. Para tal fin, el diseño de las tarjetas (figura 5-2) usan la misma forma para los tres tipos de entidades.

Visualmente, sobre la cara de una tarjeta primitiva, hay un espacio para mostrar el nombre de la tarjeta **Xm**, otro para el parámetro **Xp** que es aputando al objeto especial que va a ser usado, y un área para dibujos **Xi** que representa visualmente un grupo de entidades.

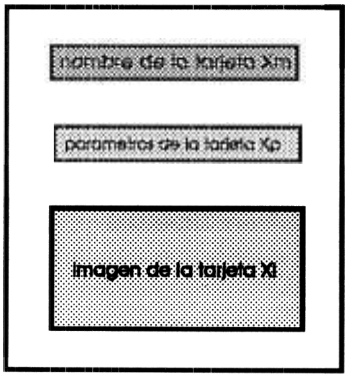


Figura 5-2 Una tarjeta primitiva

Mencionamos que en el sistema hay tres tipos de tarjetas. Para la tarjeta en la cual Xt = PROCESO, diseñamos el cuadro **Xm** de acuerdo al significado inherente a las operaciones. Por ejemplo, para la operación "**OPEN**", usamos la imagen de una persona que esta abriendo un libro; la operación "**INSIDE_OF**" es representado por un objeto dentro de otro.

En muchos casos es difícil y frecuentemente infructuoso encontrar un conjunto de tarjetas universalmente aceptables. Como alternativa, las tarjetas podrían ser definidas por el usuario, su forma adoptaría las necesidades particulares del usuario y su propia representación mental de las tareas y métodos que él desee desarrollar.

Un tipo de representación de tarjetas es la de los objetos del mundo real. Estos objetos son los objetos espaciales dentro de los archivos de datos. En este caso, nosotros dejamos que el usuario defina las tarjetas mediante el empleo de la herramienta generadora de tarjetas del sistema (como se explicó en el capítulo 4).

En este proyecto, proponemos los símbolos **Xi** para los objetos reales, lo cuales están de acuerdo con el sistema estandarizado de símbolos de México, usado por todos los fabricantes de mapas de los departamentos del gobierno, en particular los generados por el INEGI. Esto permite a los ingenieros civiles comprender directa y rápida el significado funcional de las tarjetas, como se ve en la figura 5-3.

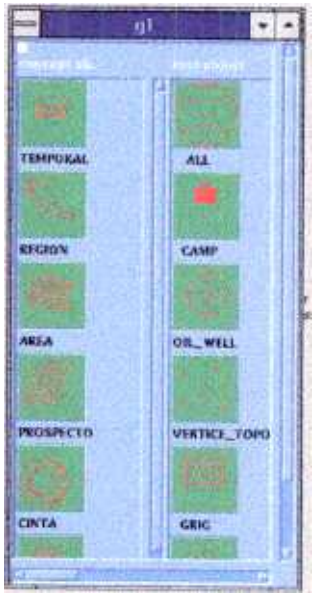
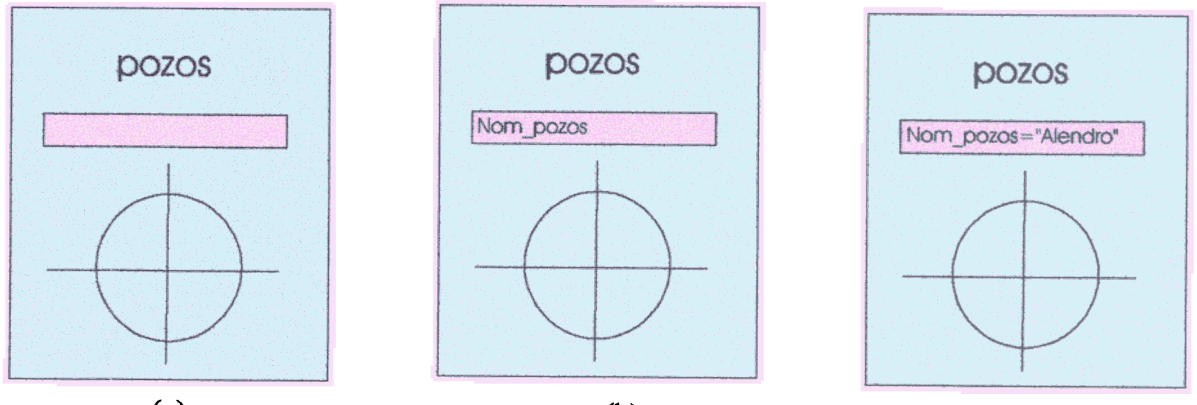


Figura 5-3 Ejemplo de las tarjetas de objeto

5.2.3 Los parámetros de la tarjeta: Xp

La función del elemento **Xp** de una tarjeta es representar el objeto de una forma más precisa. Una tarjeta puede representar un conjunto de entidades, o una entidad, o atributos de entidades dependiendo de los parámetros. Por ejemplo, la tarjeta sin parámetro en figura 5-4(a) representa un conjunto de la entidad de "POZOS". La tarjeta con parámetro(Nom_pozos) en figura 5-4(b) representa un conjunto del dominio "Nom_pozos" de la entidad "POZOS". Y la tarjeta con parámetro (Nom_pozos= "Alendro") en la figura 5-4(c) representa un objeto pozo llamado "Alendro".



(a) (b) (c)

Figura 5-4 Una tarjeta se asigna distintos parámetros indica distintos objetos

5.3 ESPECIFICACIÓN DEL LENGUAJE VISUAL DE CONSULTA (CQL)

Definición 5-2:VQS (Visual Query Sentence)

La estructura formada espacialmente por un arreglo de tarjetas, es llamada Sentencia de Consulta Visual ***(VQS).***

Definición 5-3:CQL (Card Query Langunge)

Un conjunto de VQS's pueden formar un programa del denominado Lenguaje de consulta con tarjetas ***(CQL).***

El **CQL** está especificado por el triple (**CD,G, B**),
donde: **CD** es el diccionario de tarjetas,

G es una gramática libre de contexto,

B es una base de conocimientos. El conteniendo la información que se usa para construir el significado del VQS dado. La información respecto(o refiérese):

a) nombre de evento.
b) relación conceptual.
c) nombre de objeto resultante.
d) referencia al objeto resultante.

El diccionario de tarjetas **CD** se constituye con la unión de las tres clases anteriormente determinadas (**Tu**$\cup$**Ts1**$\cup$**Ts2**),
donde

Tu = { $CARD_{user}$ | CARD= (Xm, Xi, OBJECT_R, Xp) $\in$ CD }

representan objetos aplicados y son definidos por el usuario;

Ts1 = { $CARD_{system}$ CARD= (Xm, Xi, OBJECT_C, Xp) $\in$ CD }

representan los archivos del sistema. Estos son proporcionados por el sistema y;

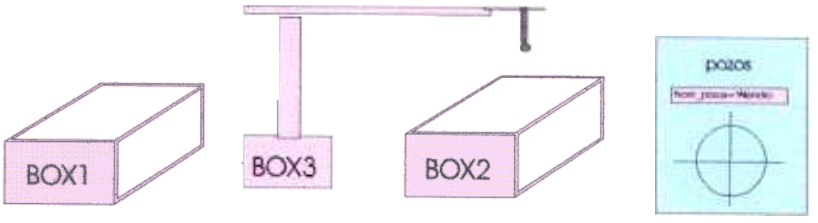
Ts2 { CARD system | CARD= (Xm, Xi, PROCESS,Xp) $\in$ CD }

representan operaciones de consulta. Este es proporcionado por el sistema

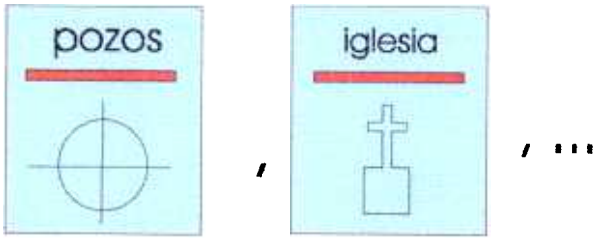
La gramática G puede ser representado por una sexta-tupla, **<N, T, S, P, Q, E>**

donde:

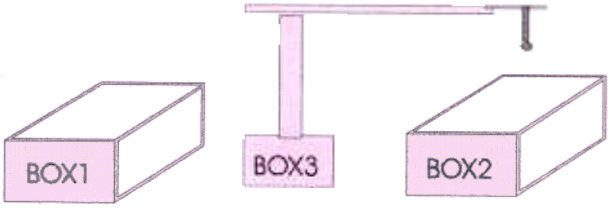
N es el conjunto de símbolos no terminales, es finito y no vacío. Cada elemento del conjunto es una tarjeta no estática. Por ejemplo:



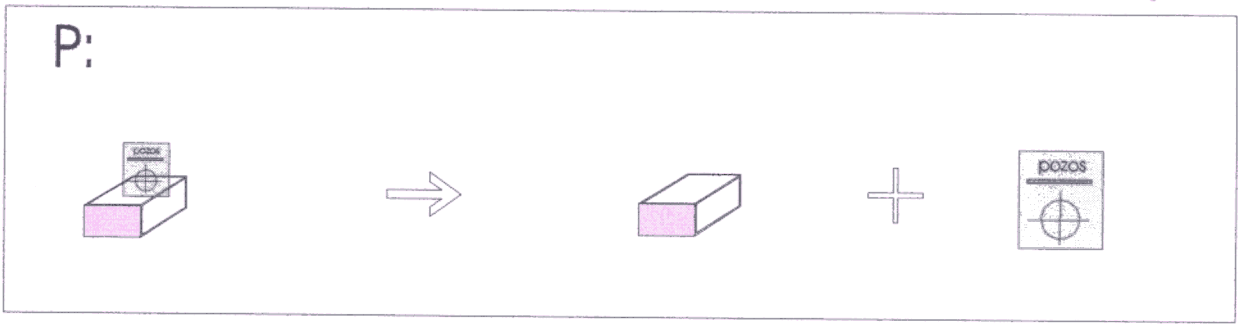
T es el conjunto de tarjetas estáticas, que representan los símbolos terminales de G, y son expresados gráficamente por:



S $\in$ **N** es un conjunto de símbolo inicial compuesto de BOX1, BOX2, BOX3, y expresados gráficamente por:



P es un conjunto finito de construcciones o reglas reescrituras, y expresados gráficamente por ejemplo:



Cada construcción en **P** tiene la forma

$$\Gamma \rightarrow C_1 \Psi_1 C_2 \ldots \Psi_{m-1} C_m, \Delta$$

La regla de inferencia de atributo, Δ, define los valores sintácticos de atributos de Γ que dependen de los valores sintácticos de atributos de C_i.

El símbolo $\rightarrow$ se lee "**puede ser reescrito como**". Γ es una tarjeta no-terminal, cada $C_i \in N \cup T$ y cada de Ψ_i, es una relación compuesta de la forma

$$\Psi i: \quad (R_i^{h_1}, R_i^{h_2}, \ldots, R_i^{h_{i-1}}, \text{NULL}, R_i^{h_{i+1}}, \ldots, R_i^{h_m}$$

Cada $R_i^{h_j}$ denota un par (R_i,h_j) donde Ri es una relación en **Q** que relaciona los valores de los atributos sintácticos de C_i con los de C_{hj} y $1 \leq h_j \leq m$

Q representa un identificador de la relación entre las tarjetas.

E es un parámetro evaluador, el cual convierte representaciones visuales en elementos numéricos o caracteres.

Para entender fácilmente los elementos del gramática G, se usa un ejemplo de construcción Γ que se muestra en la figura 5-5.

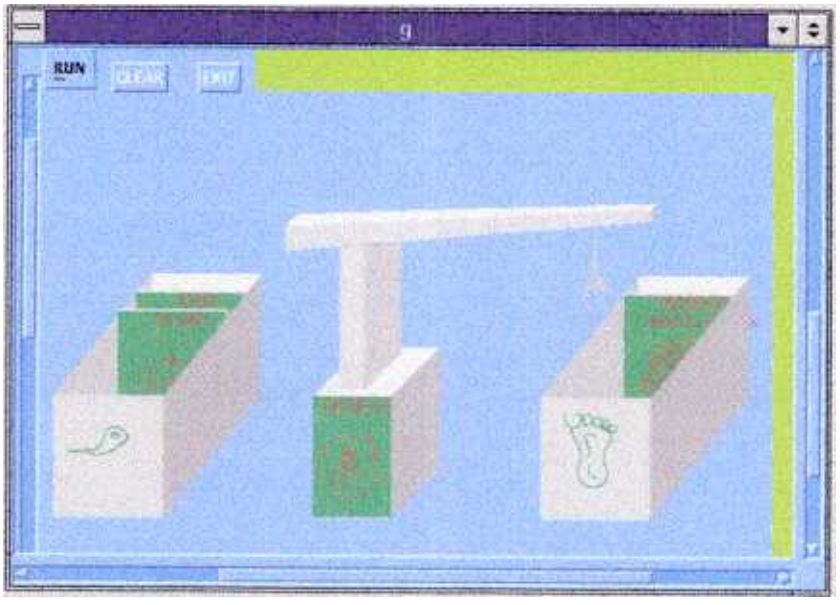


Figura 5-5. Una construcción.

En la construcción incluyendo 7 tarjetas. Las tarjetas de RIO y POZO son los tipos de entidad real. Las tarjetas vinieron del diccionario de entidades reales. Sus números en el diccionario son 1_4 y 1_7. La tarjeta de PROSPECTO vino del diccionario de entidades conceptuales. Su número es 2_5. La tarjeta de INSIDE_OF vino del diccionario de entidades de proceso. Su numero es 3_4.

Se usan dos tablas para representar **E** y **Q** del ejemplo.

E:

No_tarjeta	Nombre de tarjeta	parametro de tarjeta
1_4	POZO	null
1_7	RIO	null
3_4	INSIDE_OF	null
2_5	PROSPECTO	nom_prosp=DETALLE_DR_COSS

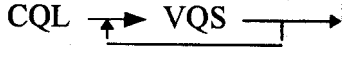
Q:

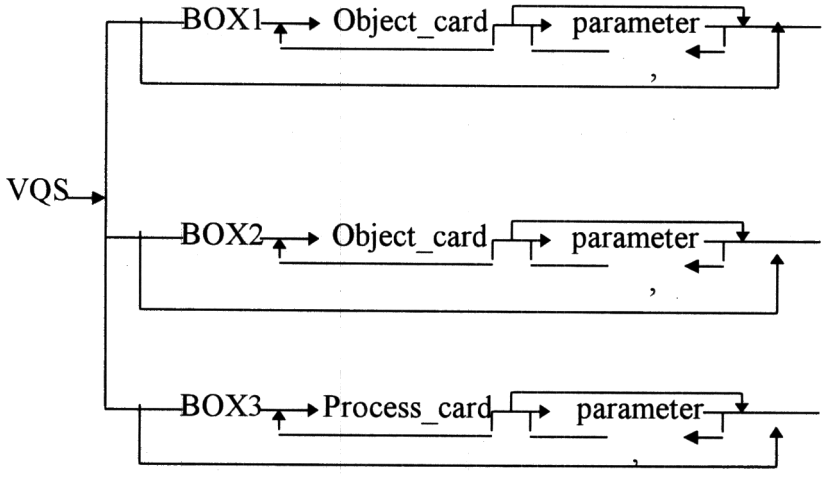
No_tarjeta	box1	box2	box3	1_4	1_7	3_4	2_5
box1	0	0	0	1	1	0	0
box2		0	0	0	0	0	1
box3			0	0	0	1	0
1_4				0	0	0	0
1_7					0	0	0
3_4						0	0
2_5							0

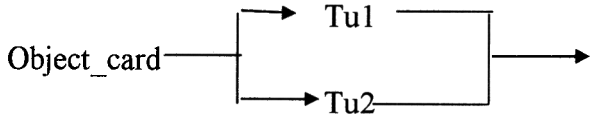
Ψ1: **(0,0,0,1,1,0,0)** **0 representa relación NULL.**
Ψ2: **(0,0,0,0,0,0,1)** **1 representa una relación.**
Ψ3: **(0,0,0,0,0,1,0)**
Ψ4: **(1,0,0,0,0,0,0)**
Ψ5: **(1,0,0,0,0,0,0)**
Ψ6: **(0,0,1,0,0,0,0)**
Ψ7: **(0,1,0,0,0,0,0)**

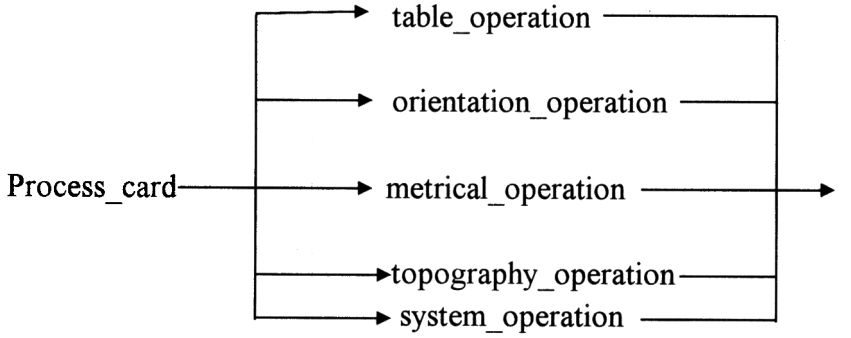
5.4 EL DIAGRAMA DE LA SINTAXIS DEL CQL

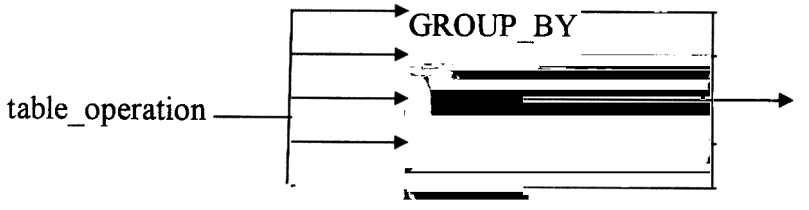
Damos el diagrama de la sintaxis del lenguaje textual, que corresponde al CQL

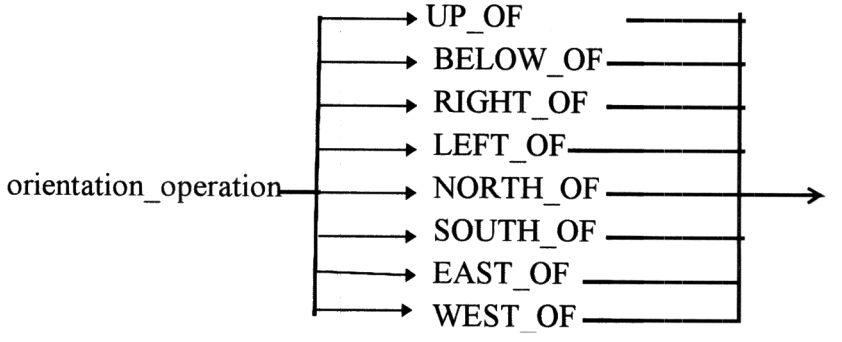


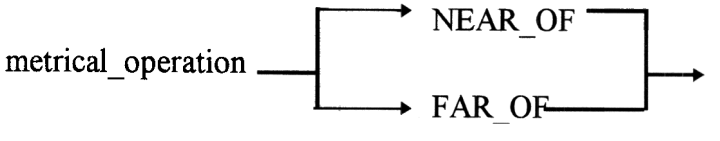


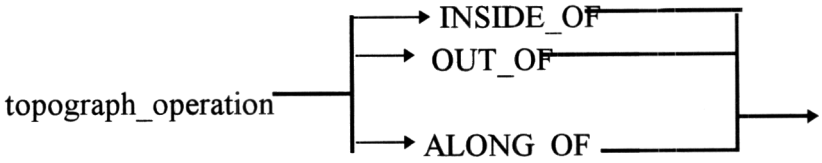

Object_card
Tu1
Tu2

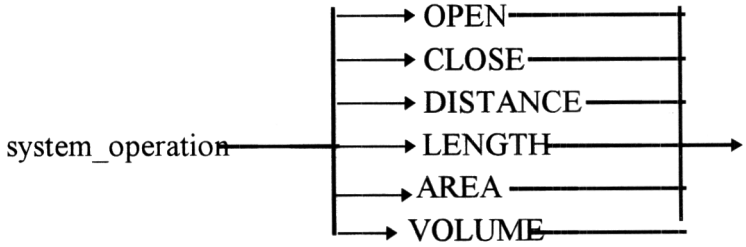

Process_card
table_operation
orientation_operation
metrical_operation
topography_operation
system_operation

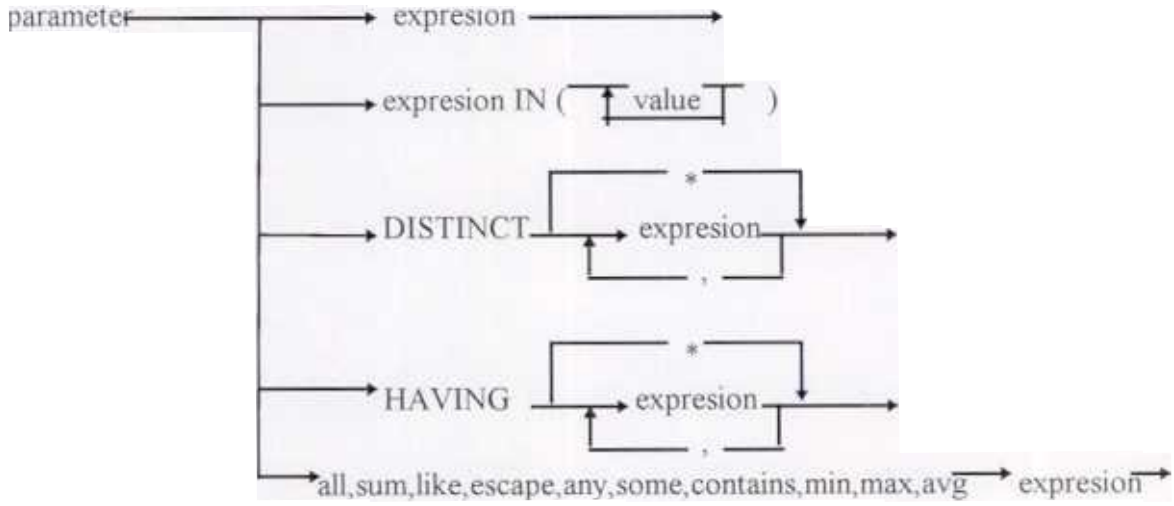

table_operation
GROUP_BY


orientation_operation
UP_OF
BELOW_OF
RIGHT_OF
LEFT_OF
NORTH_OF
SOUTH_OF
EAST_OF
WEST_OF


metrical_operation
NEAR_OF
FAR_OF





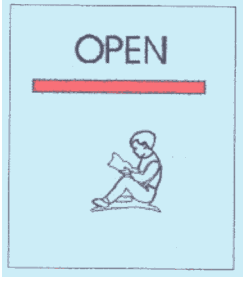


expresion ⟶ (nombre de atributo de la entidad)

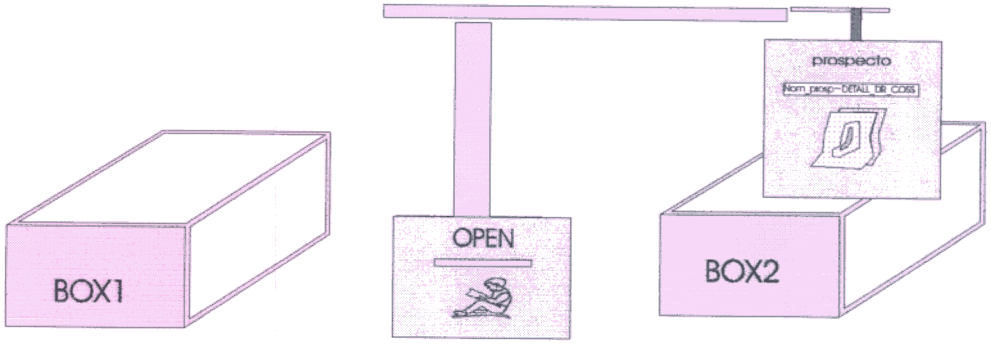
5.5 DEFINICION DE LA SINTAXIS Y LA SEMANTICA DE LAS TARJETAS DE PROCESOS

En esta sección se describen algunos ejemplos que definen las sintaxis y semántica de las tarjetas de procesos además se explican sus algoritmos desde el punto de vista de interpretación del CQL.

Tarjeta **OPEN (similar CLOSE)**:



<sintaxis>



<semántica>

La tarjeta **OPEN** abre una entidad(conceptual) y la activa como un archivo de trabajo.

<ejemplo>

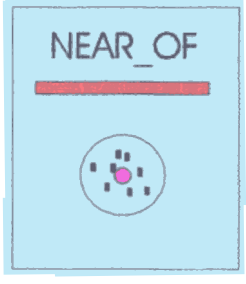
En el BOX2 de la figura anterior hay una tarjeta de "PROSPECTO" que tiene como parámetro "Nom_prosp=DETALLE_DR_COSS", y en el BOX3 hay la tarjeta de proceso "OPEN", lo que abrirá el archivo "DETALLE_DR_COSS" el cual queda como una área de trabajo.

<El algoritmo del procesamiento>:

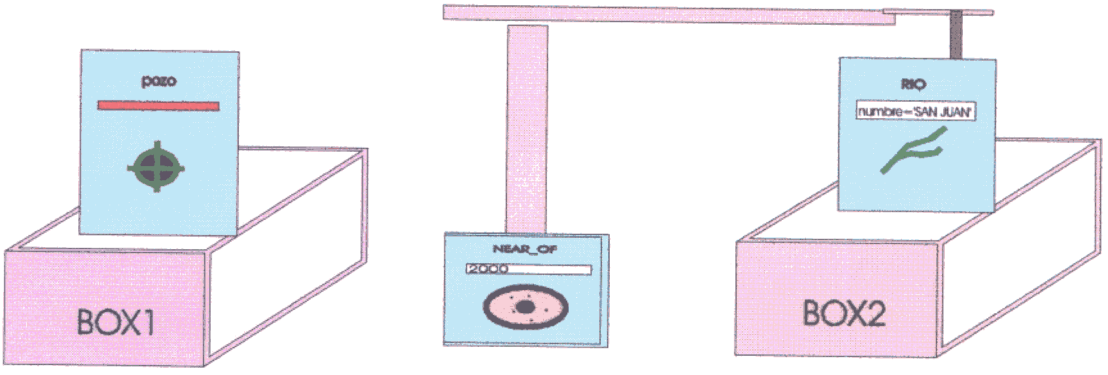
interprete del encabezado del archivo f;
mientras se encuentren más entidades en el archivo
comienza
próximo E(f, ∀s);
poner el E(f, ∀s) a la tuberia de salida
fin
despliegue la tuberia de salida

***nóta: **E(f,s)** denota expresiones lógicas para recuperar la entidad **s** que esta incluida en la entidad **f**.

Tarjeta **NEAR_OF (similar FAR_OF)**:



<sintaxis>



<semántica>

Seleccionar las entidades **As** cercan de la entidad **B** en menos de **r** metros. El **r** es el parametro de la entidad de proceso. Las entidades **As** y **B** $\in$ Archivo de trabajo que se encuentra definido actualmente.

<ejemplo>

El VQS anterior va a recuperar los pozos cerca del río "SAN JUAN". El ancho de la área es 2*2000 metros a todo lo largo del río (2000 a cada lado).

<Algoritmo de procesamiento>:

Caso 1: Si la entidad B es una entidad puntual, y su coordenada es B(x,y). Se buscan las entidades As. Si están dentro de la área circular con centro en el punto B(x,y) y el radio es r que se definió en la parte de parámetro de la tarjeta de NEAR_OF.

Caso 2: Si la entidad B es una entidad lineal, se buscan las entidades As. Si están dentro de la área que a lo largo de entidad B y su ancho es 2.r que se definió en el parámetro de la tarjeta NEAR_OF.

Caso 3: Si la entidad B es una entidad formada por área. Primero se calcula el centro del área:

$$Xc = (X1 + X2 + .. + Xn) / n$$
$$Yc = (Y1 + Y2 + .. + Yn) / n$$

donde (Xi,Yi) son las coordenadas de polígono de frontera del área B.

Luego se hace el procesamiento como en el caso 1.

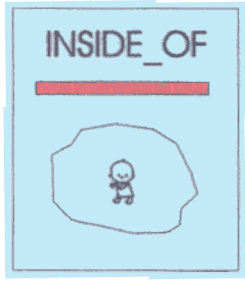
Caso 4: Si la entidad B es una entidad formada por volumen. Se calcula:

$$Xc = (X1 + X2 + ... + Xn) / n$$
$$Yc = (Y1 + Y2 + ... + Yn) / n$$
$$Zc = (Z1 + Zc + ... + Zn) / n$$

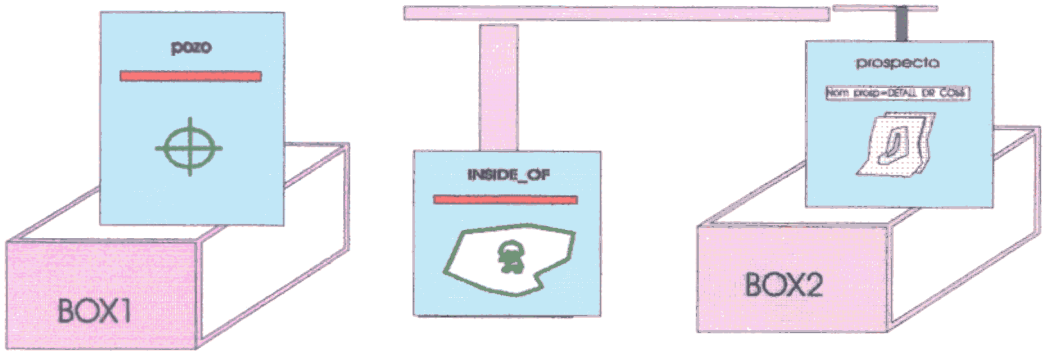
donde {Xi,Yi,Zi} son coordenadas de vértices de las caras de superficie del volumen.

Luego se buscan las entidades As que están dentro de la esfera cuyo el centro está en punto B(Xc,Yc,Zc), y r como se definió en la tarjeta NEAR_OF.

Tarjeta **INSIDE_OF (similar OUT_OF)**:



<sintaxis>



<semántica>

Recupera las entidades, s1, que se definen en la caja 1 la cúal incluye en las entidades, s2 que se definen en la caja 2.

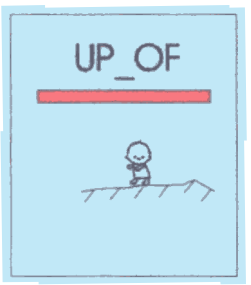
<ejemplo>

El VQS arriba representa la recuperación de los pozos que se ubican dentro del proyecto "DETALLE_DR_COSS".

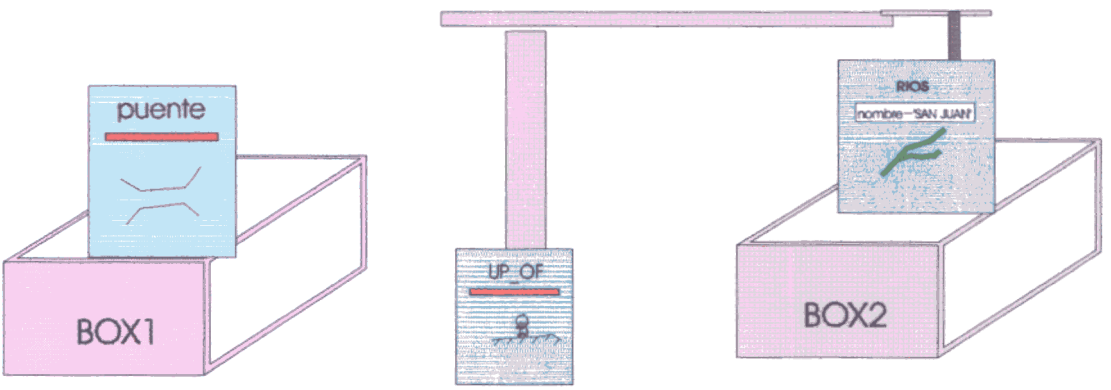
<El algoritmo de procesamiento>:

interprete del encabezado de los archivos f de s2;
mientras se encuentren más entidades en el archivo se circula:
comienza
próximo E(f,s1);
if E(f,s1) esta dentro de ∀E(f,s2)
then insertar E(f,s1) a tuberia de salida
fin
despliegue los datos espaciales del tuberia de la salida;

Tarjeta UP_OF (similar BELOW_OF):



<sintaxis>



<semántica>

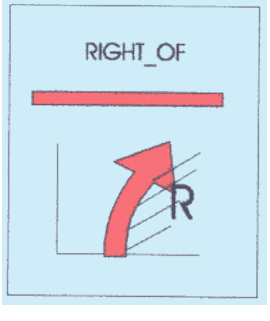
Recupera las entidades, s1, en la caja 1, los cual están arriba de las entidades, s2, en la caja 2. Las entidades s1,s2 $\in$ Archivo de trabajo.

<ejemplo>
El VQS arriba representa que se recuperarán los puentes que se ubican arriba del río "SAN JUAN".

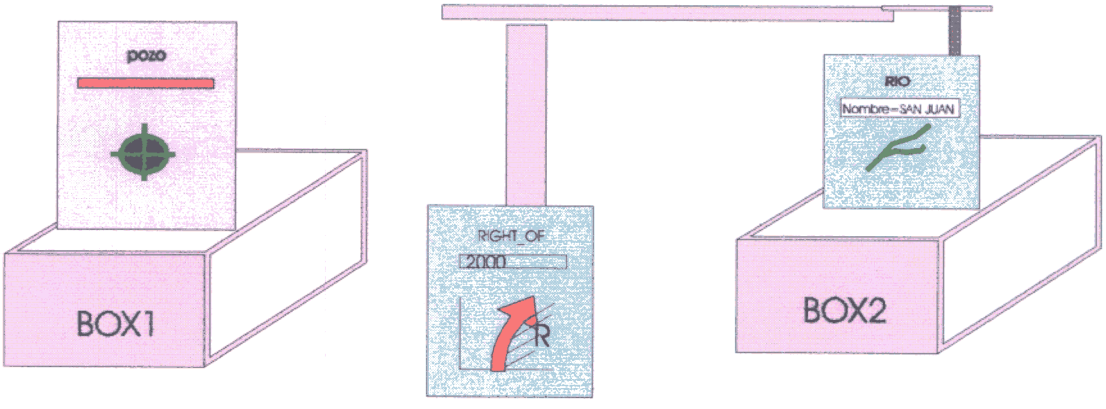
<El algoritmo del procesamiento>:

interpretar encabezado de los archivos en los cuales están los objetos s1,s2;
mientras existan más entidades en el archivo
comenzar
if test E(f,s) ∈ s1 then E(f,s) →T1;
if test E(f,s) ∈ s2 then E(f,s) → T2
fin
mientras existan más entidades en T1
comenzar
próximo E(T1,s1);
if E(T1,s1) esta arriba de ∀E(T2,s2)
then insertar E(T1,s1) a la tuberia de sálida.
fin
despliegue los datos espaciales de la tuberia de salida.

Tarjetas **RIGHT_OF (similar LEFT_OF)**:

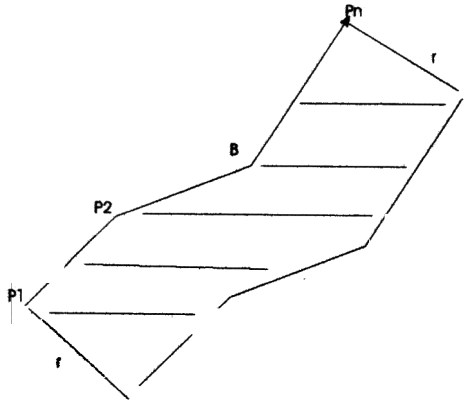


<sintaxis>:



<semántica>:

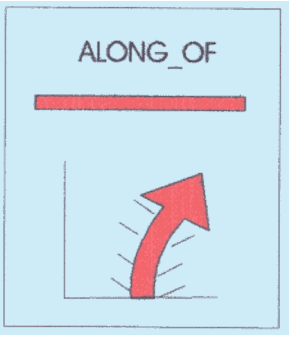
Recupera las entidades puntuales As, definidas en BOX1, que se ubican a la derecha de (izquierda de) la entidad lineal B, a una distancia menor de **r** metros, definida en BOX3. La dirección de B se define según los registros (P_1, P_2, P_3, ..., P_n), es decir, P_1P_2, .., P_{n-1} P_n. El área de búsqueda se define como en la siguiente figura. Las entidades As y B $\in$ Archivo de trabajo que se encuentra definido actualmente.



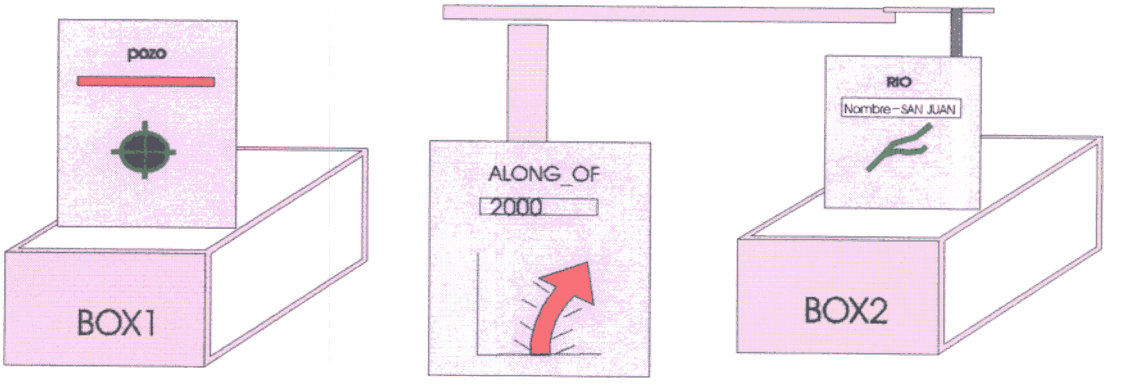
<ejemplo>:

El VQS arriba va a recuperar los pozos que se ubica a la derecha del río SAN JUAN. El ancho de la área es 2000 metros.

Tarjeta **ALONG_OF**:

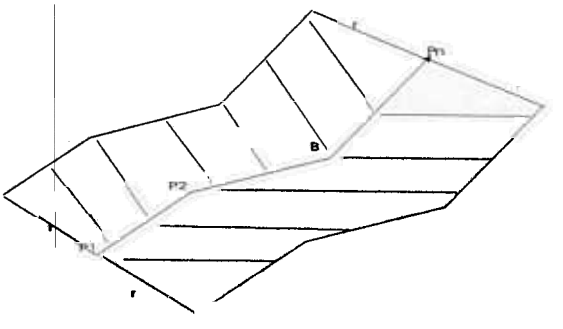


<sintaxis>:



<semántica>:

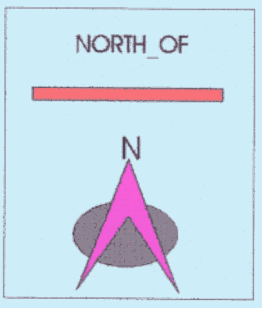
Recupera las entidades puntuales As, definidas en BOX1, que se ubican a lo largo de la entidad lineal B definida en BOX2. El área de búsqueda es de un ancho de 2*r definido como un parámetro de la tarjeta de ALONG_OF, como la figura siguiente. Las entidades As y B $\in$ Archivo de trabajo que se encuentra definido actualmente.



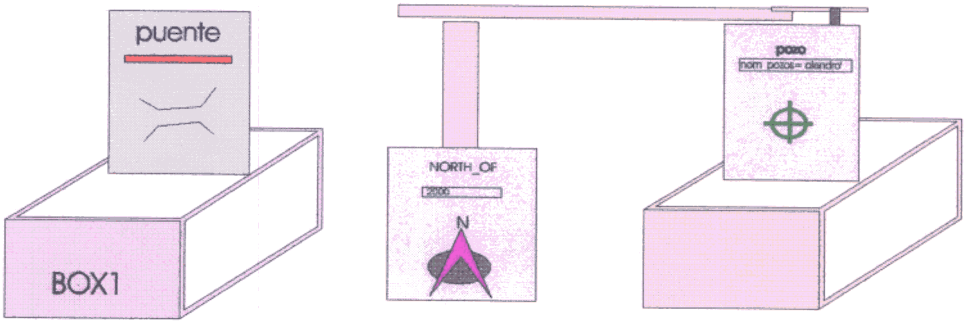
<ejemplo>:

El VQS arriba va a recuperar los pozos que se ubican a lo largo del río SAN JUAN. El ancho de la área es 2x2000 metros a todo lo largo del río.

Tarjeta **NORTH_OF (similar SOUTH_OF, EAST_OF, WEST_OF)**

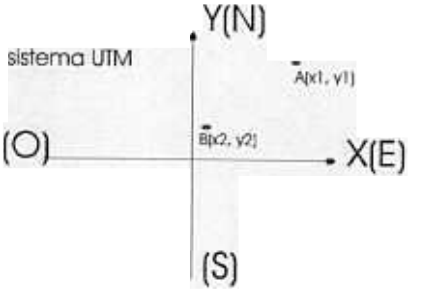


<sintaxis>:



<semántica>:

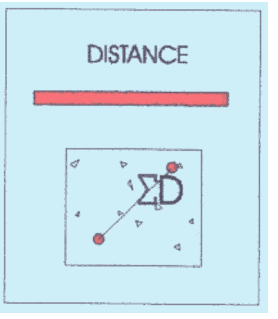
Recupera las entidades puntuales As, definidas en BOX1, que se ubican al norte de (sur de, ...), la entidad puntual B definida en BOX2. Las direcciones corresponden al sistema de coordenadas UTM como se muestra en la figura siguiente. Por ejemplo, tenemos entidad puntual A(x1,y1), entidad puntual B(x2, y2). Si x1 > x2, entidad A se ubica este de entidad B. En el caso contrario, A se ubica oeste de B. Las entidades As y B $\in$ Archivo de trabajo que se encuentra definido actualmente.



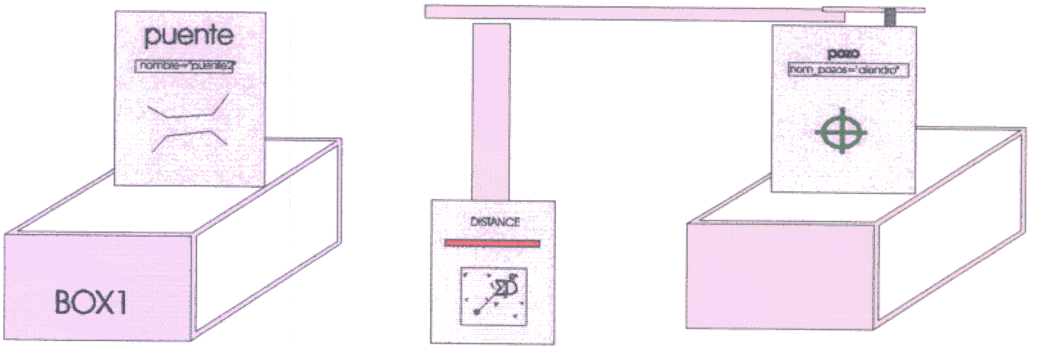
<ejemplo>:

El VQS arriba va a recuperar los puentes que se ubican en la área de la norte del pozo ALENDRO. El ancho de la área es 2000 metros.

Tarjeta **DISTANCE**:

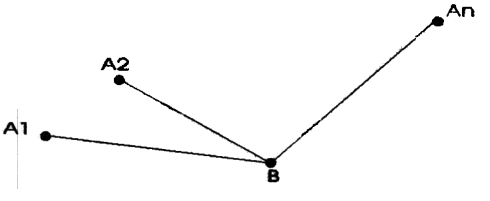


<sintaxis>:

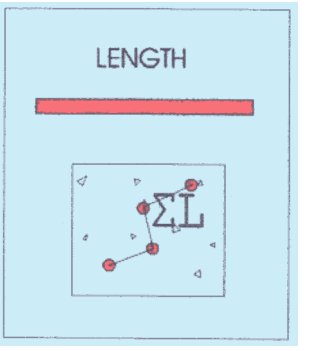


<semántica>:

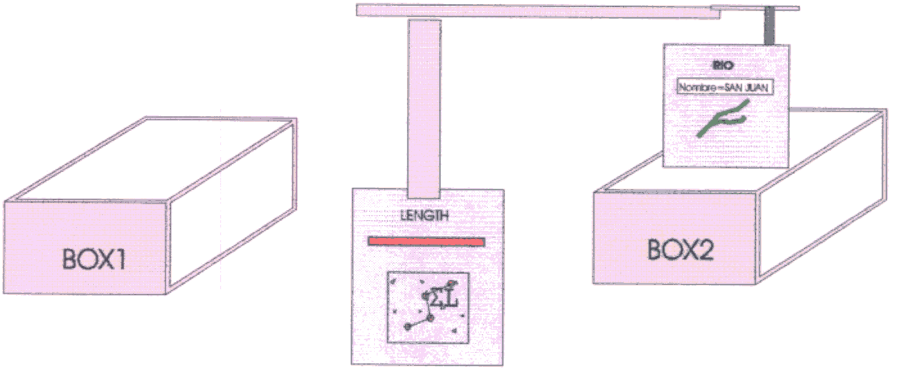
Se calcula las distancias entre las entidades puntuales As definidas en BOX1 y la entidad puntual B definida en BOX2. Las entidades As y B $\in$ Archivo de trabajo que se encuentra definido actualmente. Los resultados son las logitudes de D(A1,B), D(A2,B), ..., D(An,B) como se ve en la figura siguiente.



Tarjeta **LENGTH**:



<sintaxis>:



<semántica>:

Se calcula la longitud de la entidad lineal B definida en BOX2. Los registros de la entidad B son del tipo L, en la forma(P_1, P_2, ... , P_n). Entonces,

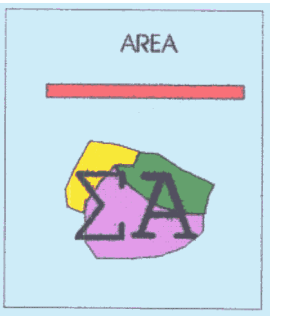
$$\textbf{Longitud} = \sum_{i=1}^{n-1} \mathbf{D(P_i,P_{i+1})}$$

donde $\mathbf{D(P_i,P_j)}$ es la distancia entre dos puntos vecinos $\mathbf{P_i,P_j}$

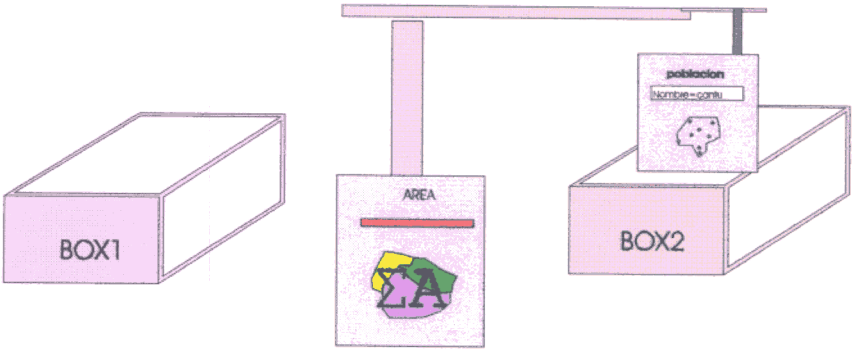
<ejemplo>:

El VQS arriba va a calcular la longitud total del río SAN JUAN.

Tarjeta **AREA**:



<sintaxis>:

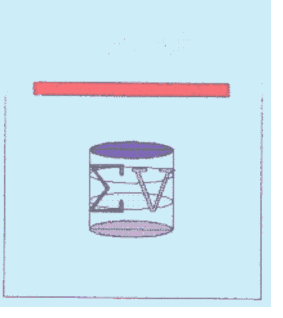


<semántica>:

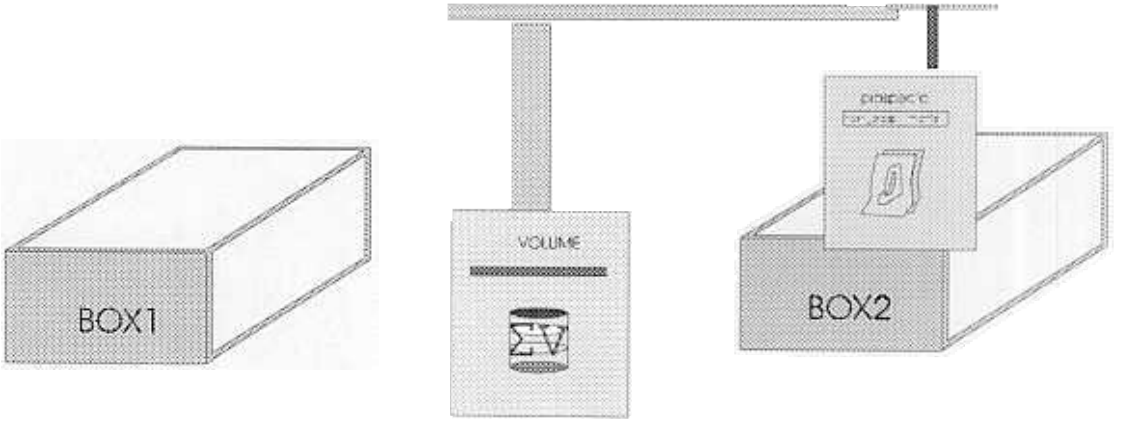
Se calcula el área de la entidad B, tipo A, definida en BOX2 como un polígono.

<ejemplo>:

El VQS arriba va a calcular la área del pueblo de CANTU.



<sintaxis>:



<semántica>:

Se calcula el volumen de la entidad B, tipo V, definida en BOX2.

CONCLUSIONES

En este capítulo, se empleó la tarjeta como elemento primitivo del lenguaje CQL. Como se hace de forma tradicional para definir un lenguaje, se definió la gramática y sintaxis del CQL bajo un ambiente visual. Y para entender fácilmente las funciones de las tarjetas de proceso, se incluyen algunos algoritmos de procesamiento del lenguaje.

CAPÍTULO 6

EL AMBIENTE VISUAL DE PROGRAMACIÓN EN CQL

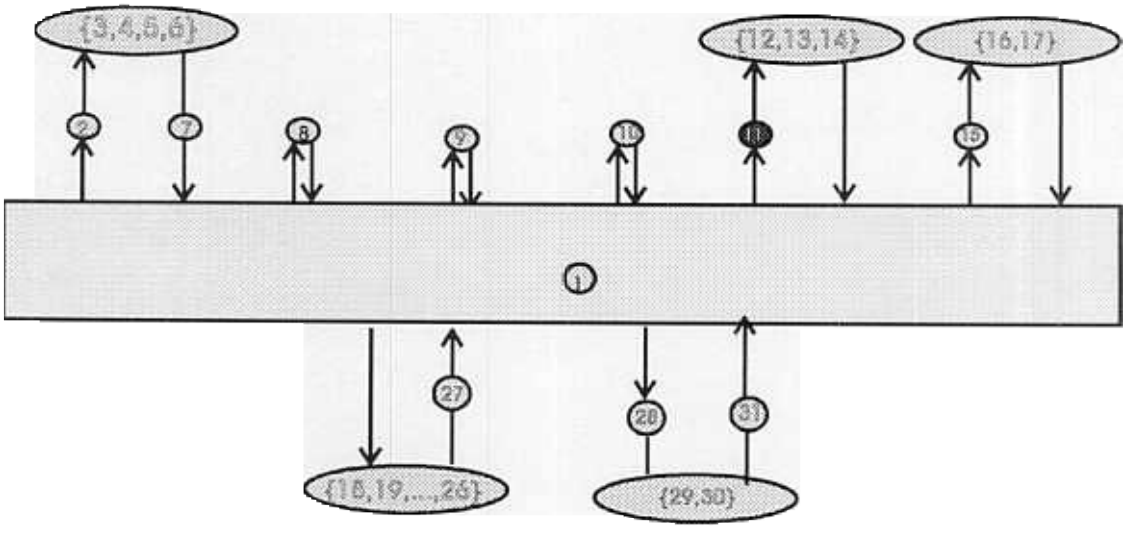
En un sistema de información Espacial la interfaz con el usuario es uno de los aspectos las importantes, la naturaleza misma de la información requiere de un ambiente de visualización, que permita observar los mapas. Esto será cada vez más real, a partir de que la información geográfica sea ahora más afectada en cuanto a la aplicación general tal y como la exploración y todas esas aplicaciones de usuarios no-especialistas.

6. TRANSFERENCIA DE ESTADOS

Es bien conocido que las interfaces de programación del usuario se dificultan. Existen un número de razones por las cuales al software para interfaces modernas del usuario, se les atribuya una mayor dificultad para escribir que otra clase de software, tal y como el diseño iterativo, dificulta la obtención de representaciones en pantalla, las entradas asíncronas, los manejos de errores, abortos y deshacer comandos.

En el sistema de CQL, adoptamos el diagrama de transición de estado [AHO86] a un modelo dinámico de interfaz del usuario. La figura 6-1 es una representación sintética de un autómata de estados finito de la interfaz del usuario.

En la figura 6-1, se muestran los estados que hay en el sistema y las direcciones de las transiciones entre los estados. Por ejemplo, estado 1 representa que en este momento el sistema está estático, después de dar comando de ejecución, el sistema va a uno de cuatro estados: tuberías de salidas de gráfico, texto, objeto temporal o tres direcciones. Los estados del 18 hasta el 26 son operaciones de recuperación de entidades de la base de datos.



Cada a —o→b es una transición automática

Para funciones globales:

Estado 1. estado neutral (estático)
Estado 2. ejecución
Estado 3. salida gráfica
Estado 4. salida de texto
Estado 5. salida de objeto temporal
Estado 6. salida de tres direcci
Estado 7. administración de resultado de consulta.
Estado 8. limpieza
Estado 9. comenzar programación en CQL
Estado 10. salir

Para editar textos:

Estado 11. edición textos
Estado 12. cortar
Estado 13. salvar
Estado 14. nuevo

Para editar gráficos:

Estado 15 edita gráficos
Estado 16. zoom out
Estado 17 zoom in

Para recuperar entidades:

Estado 18. arriba de
Estado 19. abajo de
Estado 20. dentro de
Estado 21. fuera de
Estado 22. conectado con
Estado 23. abrir

Estado 27. modelo físico de la base de datos

Para ventana temporal:

Estado 28. validación de ventana temporal
Estado 29. poner parámetros
Estado 30. definición de la caja que lo inserta
Estado 31. validación de tarjeta activa

Figura 6-1 Autómata de Estados Finitos para la interfaz de usuario

Los generadores de interfaces dividen la pantalla en varias áreas de diálogo, como se muestra en la figura 6-2.

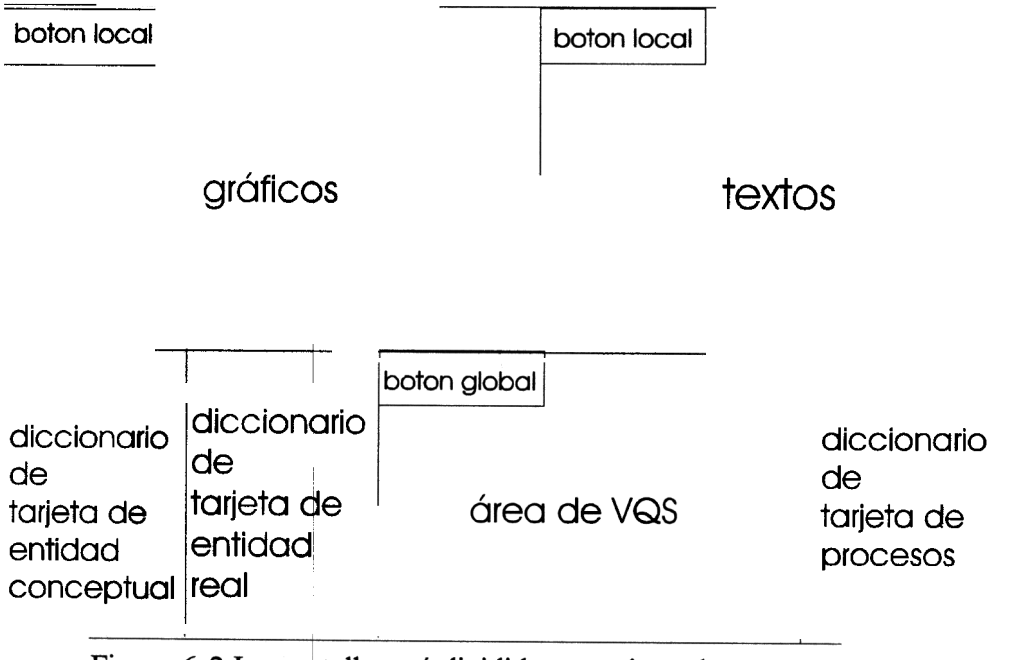


Figura 6-2 La pantalla está dividida en varias sub-ventanas

En el sistema las sentencias de las consultas se dán en una área de trabajo, la máquina de consulta y el editor VQS permite que se declaren visualmente las consultas para ser gestionadas por la máquina de base de datos. En las tres sub-ventanas del diccionario de tarjetas se despliegan tres tipos de tarjetas respectivamente para la selección por el usuario final, cuando éste programa en lenguaje CQL. El resultado de la consulta puede ser presentada en modo gráfico, en modo de texto o en todos los modos, por lo que existen dos áreas de consulta. También puede ser respaldado en tarjeta de objeto temporal. El botón funcional tiene dos clases: global y local. Una es para funciones de ventanas globales y el otro es para funciones de ventanas locales.

Un usuario final puede definir diferentes subconsultas independientes, antes de definir una interacción entre uno (o varios) objeto(s) de una subconsulta y uno (o varios) objeto(s) de otra subconsulta. Si una consulta final tiene un alto nivel de complejidad, el área de consulta necesitará proporcionar bastantes subáreas, una para cada subconsulta independiente en el área de consulta. Puesto que el número de subconsultas es dinámico, este puede ser incrementado y/o decrementado durante el proceso de definición de consultas. Varias soluciones son posibles, entre las cuales: el re-dibujamiento del área de consulta en cada modificación del número de subconsultas, la administración de las páginas de subconsultas con una página para cada subconsulta mediante un mecanismo de barra de desplazamiento, o la limitaste de un número fijo

de subáreas para permitir más consultas del usuario final. Nosotros proponemos un mecanismo, llamado objeto temporal, como una tarjeta conceptual. El usuario final puede poner el resultado de cada subconsulta, en una tarjeta de acuerdo a un orden sucesivo para efectuar la consulta, de esta manera el área de consulta no necesita proveer más subáreas.

6.2 CONSULTA MÉDIATE EL LENGUAJE CQL

La figura 6-3 presenta el editor de sentencia visual de consulta de el lenguaje CQL. Este editor proporciona dos clases de estructuras, correspondientes al área de consulta y al área de diccionario. El área de consulta es usada para visualizar las relaciones espaciales involucradas en los objetos básicos de una consulta visual. El área de diccionario incluye los tres tipos de tarjetas que son vinculadas a los dos siguientes conceptos:

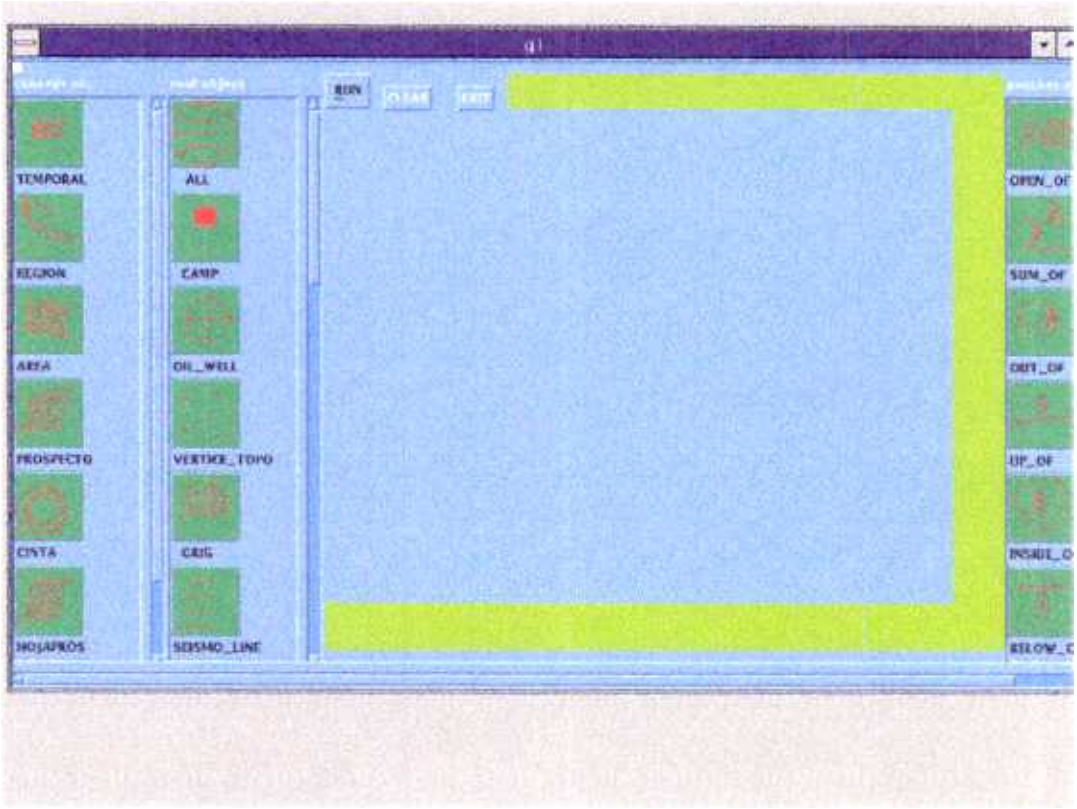


Figura 6-3 Editor de CQL

(a) . La semántica de las tarjetas, representada tanto por una imagen dentro de la tarjeta y una etiqueta de texto de parámetros.

(b) . Las acciones generadas por la activación de la tarjeta y una etiqueta de texto de los parámetros.

En la columna izquierda se muestran las tarjetas de tipo de objeto conceptual, que se definieron en el capítulo 2. Cada tarjeta corresponde a un archivo físico en la base de datos. En la segunda columna, se muestran las tarjetas de tipo de entidad real, lo cual se explicó en capítulo 4. Estas tarjetas representan los objetos del mundo real. Y en la columna derecha se muestra las tarjetas de procesos. Cada una corresponde a una acción o un procesamiento del sistema, lo cual se definió en CQL. Las tres columnas se pueden mover para mostrar todas las tarjetas del diccionario con el barra de desplazamiento vertical.

Posteriormente, describiremos estas estructuras y explicaremos como construir una sentencia de consulta visual.

6.2.1 Transición de estados para la recuperación de tarjetas.

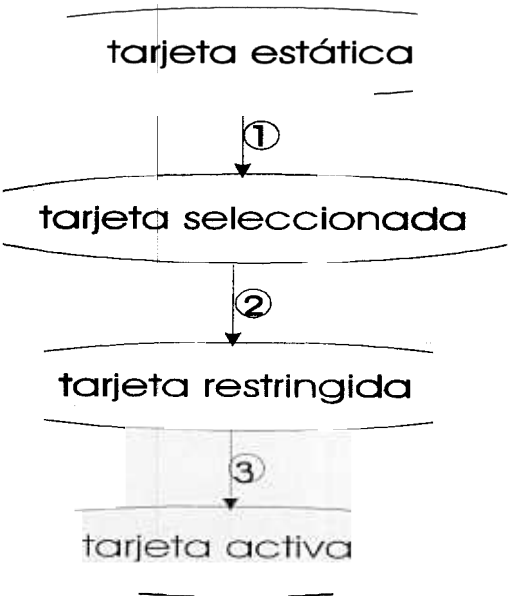
En este sistema, las tarjetas son empleadas para la representación de entidades del mundo real, mediante el significado de imágenes abstractas, y también por la aclaración del objetivo de las consultas.

Una tarjeta tiene cuatro estados:

- **Tarjeta estática** -- si una tarjeta está en el diccionario, le llamamos tarjeta estática.
- **Tarjeta seleccionada** -- cuando el usuario selecciona una tarjeta con Mouse, y sale una ventana temporal para pedir los parámetros, en este momento, la tarjeta se llama tarjeta seleccionada.
- **Tarjeta restringida** -- una que tiene definidos parámetros se llama restringida.
- **Tarjeta activa** -- la tarjeta que está en la máquina de consulta, se llama tarjeta activa.

Para especificar una consulta, primero seleccionamos una tarjeta apropiada indexando objetos para recuperarlos. Posteriormente, especificamos las condiciones para la validación de los valores de las variables en cuestión, finalmente, ya podemos obtener aquellos objetos que satisfagan nuestros requerimientos.

Esta manipulación es llevada a cabo de acuerdo con la transición de estado mostrado en la Figura 6-4. Los círculos y líneas representan estados y transiciones respectivamente. Los números asignados a las líneas representan el tipo de operación, la cual podría invocarse para efectuar la transición del un estado a otro. La restricción de estado esta representada por una ventana temporal que pide los parámetros.



(1) invoca la tarjeta desde el área de diccionario.
(2) alta de parámetros
(3) colocarse en la máquina de consulta
Figura 6-4 Transición de estados de una tarjeta.

El usuario final puede colocarse en la restricción de las condiciones por medio de la tarjeta seleccionada y elegir una Caja de maquina de consulta en donde se puede insertar la tarjeta. La caja es una parte del mecanismo de la consulta. Nosotros explicaremos esto en la siguiente sección. Por ejemplo, el usuario final selecciona una tarjeta, nombrada "POZOS", desde el área del diccionario de objeto real. Entonces, automáticamente una ventana temporal aparece en la pantalla, como se muestra en la Figura 5-10, el usuario coloca dentro la cadena "name=alondra" y presiona el botón "box 1". De esta manera en la Caja 1 se inserta una tarjeta activa .

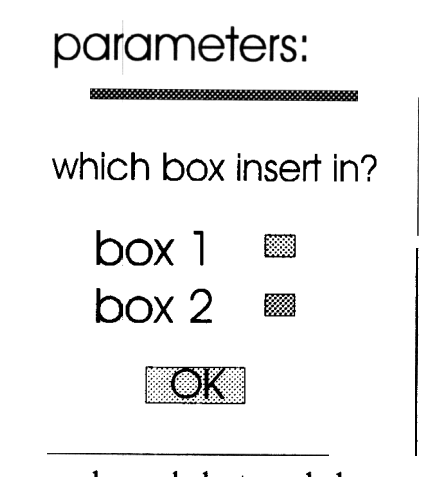


Figura 6-5 Ventana temporal para la lectura de las condiciones de restricción.

6.2.2 *Mecanismo de consulta*

El mecanismo de consulta, editor del CQL, consiste de una máquina de consulta y tres diccionarios de tarjetas. La maquina de consulta consiste de tres partes, Caja 1, Caja 2 y elevador, como se muestra en la figura 6-6.

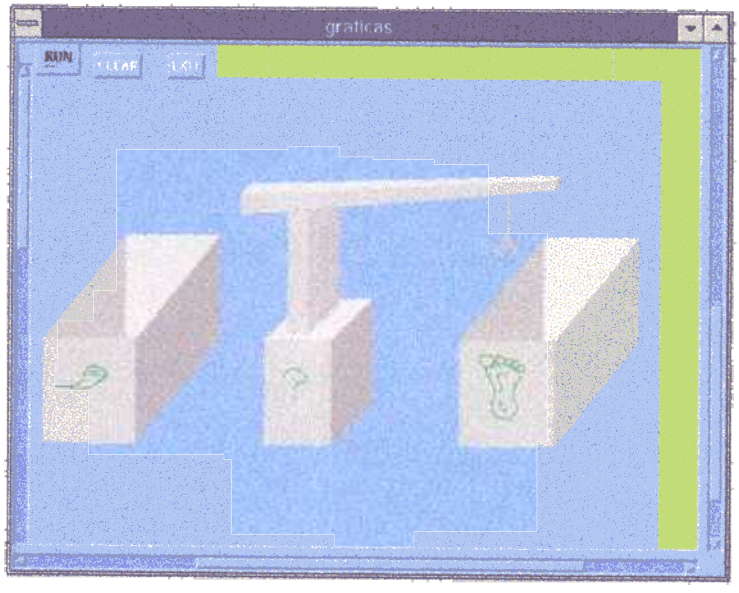


Figura 6-6 Máquina de consulta.

Una de las caras de la caja 1, tiene una imagen que expresa el significado "¿**Sobre qué trata**?" en un lenguaje de los Aztecas de México. Esta caja es empleada para colocarle tarjetas activas, las cuales son representaciones de entidades, las que el usuario ha seleccionado. Las tarjetas activas pueden provenir de las librerías de tarjetas conceptuales o de las librerías de tarjetas de objetos reales.

Sobre la cara de la caja 2, observamos otra imagen que expresa el significado "¿**Apatir de qué**?" en el lenguaje de los Aztecas. Esta caja es usada para colocarle tarjetas activas que son representaciones de las entidades desde las cuales se seleccionan las entidades que son definidas en la caja 1. Las tarjetas activas también pueden provenir de las tarjetas conceptuales o de tarjetas de objetos.

La tercera parte, simula un elevador que trabaja seleccionando algun objeto de la caja 2. Esta imagen significa "¿**Qué hacer**(acción)?". Sobre la cara del elevador pueden asignársele tarjeta activa de procesos cada vez. Ellos comienzan desde la librería de la tarjeta de procesos. La tarjeta

define la manera en la cual los objetos podrían ser seleccionados en la caja o procesados con los objetos en la caja 2.

6.2.3 Cómo editar una consulta en CQL

Los usuarios finales solo usan el ratón para seleccionar las tarjetas las cuales representan las entidades que pueden ser procesadas. Por ejemplo, si el usuario final quiere conocer que pozos petroleros se encuentran dentro del proyecto "DETALLE_DR_COSS"; el puede buscar la tarjeta "POZOS" en el área de librería de objetos reales. El puede mover la barra de desplazamiento con el ratón hasta que la tarjeta sea desplegada en la pantalla. El usuario podría hacer clic sobre la tarjeta con el ratón, una ventana temporal aparecerá pidiendo los parámetros de esa tarjeta. Si el quiere consultar toda la información acerca de los pozos petroleros, el no deberá introducir algún parámetro en el espacio. Después de que el ha seleccionado los parámetros e indicado en cual caja deben ser insertados, la tarjeta asignada a tarjeta de pozos petroleros se transformará en una tarjeta activa. Presionando sobre el botón "Ok", la tarjeta activa "OIL_WELL(pozos petroleros)" puede permanecer en la caja 1, como se muestra en la figura 5-12

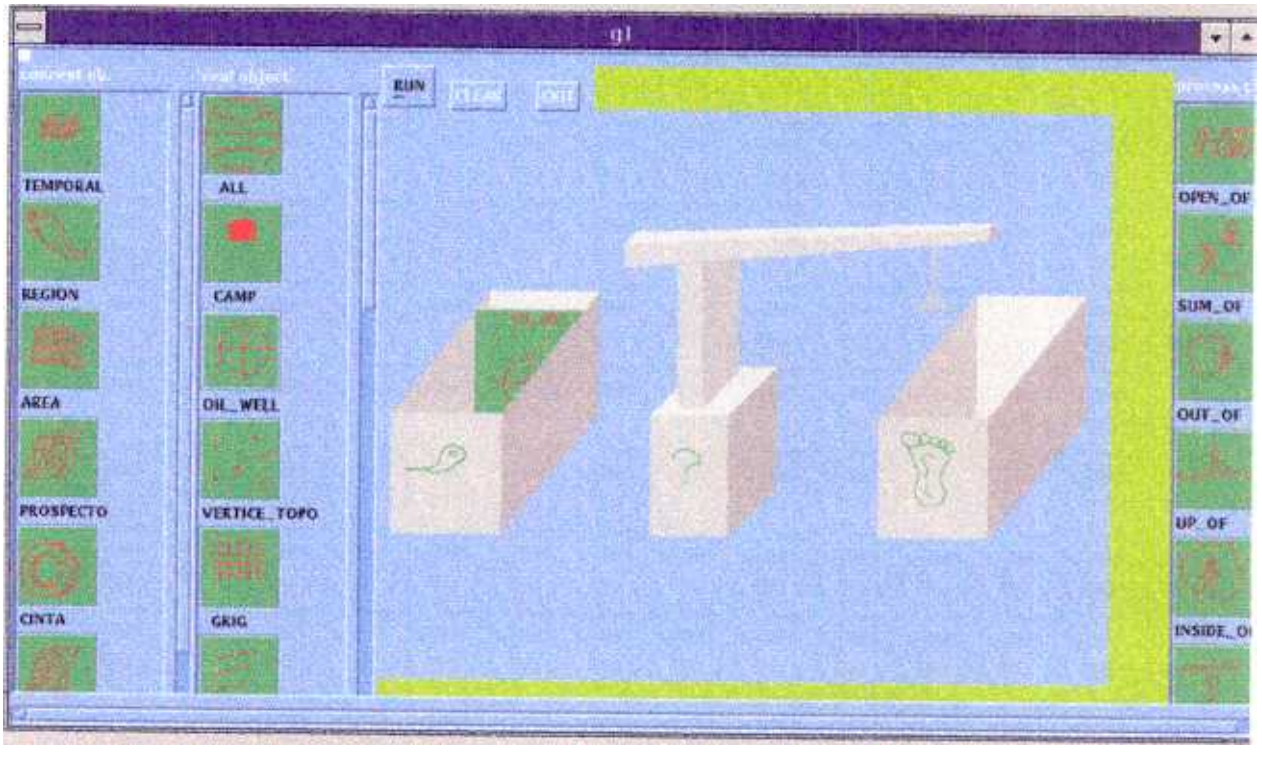


Figura 6-7 La tarjeta activa "OIL_WELL" fue insertada en la caja

Entonces para la definición desde dónde se seleccionan los pozos petroleros, el usuario busca la tarjeta de objeto "PROJECT" en el área de librerías de objetos, llamado este, coloca encima la cadena "name=DETALLE_DR_COSS", y presiona el botón "box 2", la tarjeta activa quedaría en la caja 2 como se muestra en la figura 6-8. El significado de esta acción junto con lo anteriormente descrito, es la selección del pozo petrolero del proyecto "DETALLE_DR_COSS".

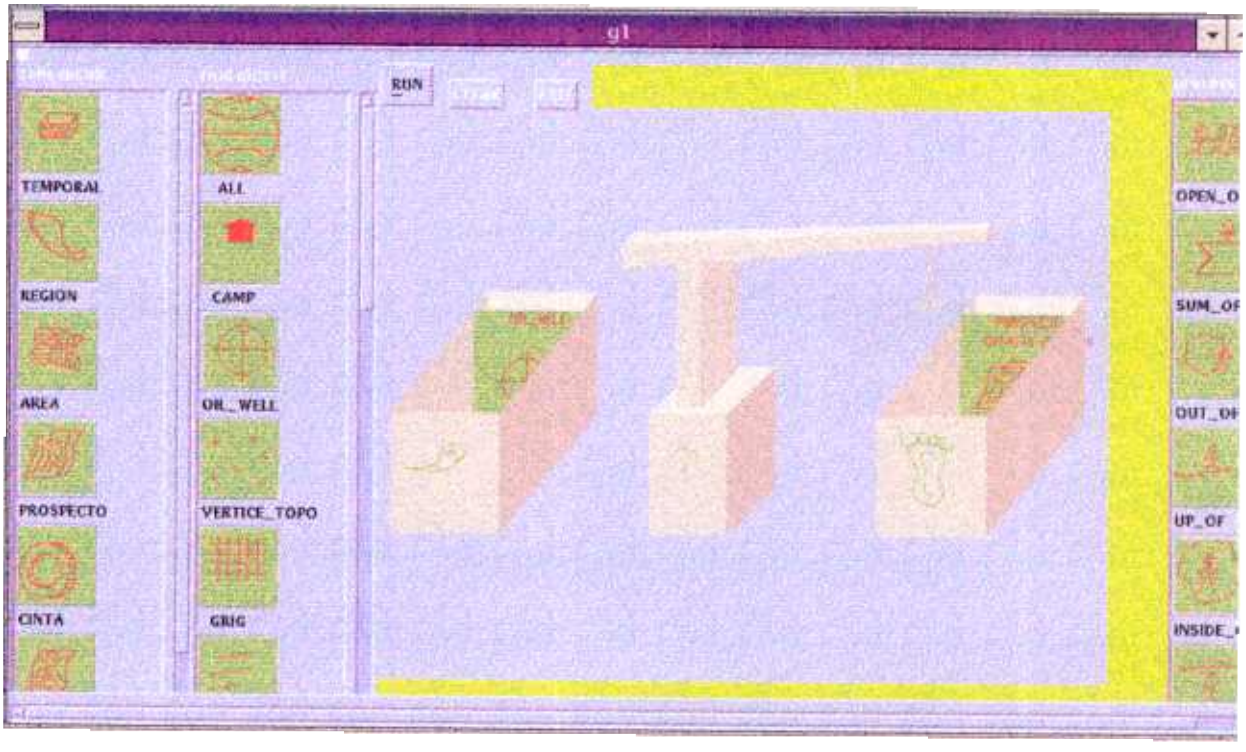


Figura 6-8 Una tarjeta activa fue insertado en la caja 2.

En la definición de cómo seleccionar, el usuario necesita encontrar la tarjeta apropiada de operación en la librería del área de procesos, entonces el podría llamar la tarjeta apropiada "INSIDE_OF". Si esta necesita algunos parámetros para la operación, aparecerá una ventana temporal en la cual el usuario puede llenar la información. Después, presionando el botón "Ok" sobre la ventana temporal, una tarjeta activa "INSIDE_OF" se le asigna al elevador, como se muestra en la figura 6-9

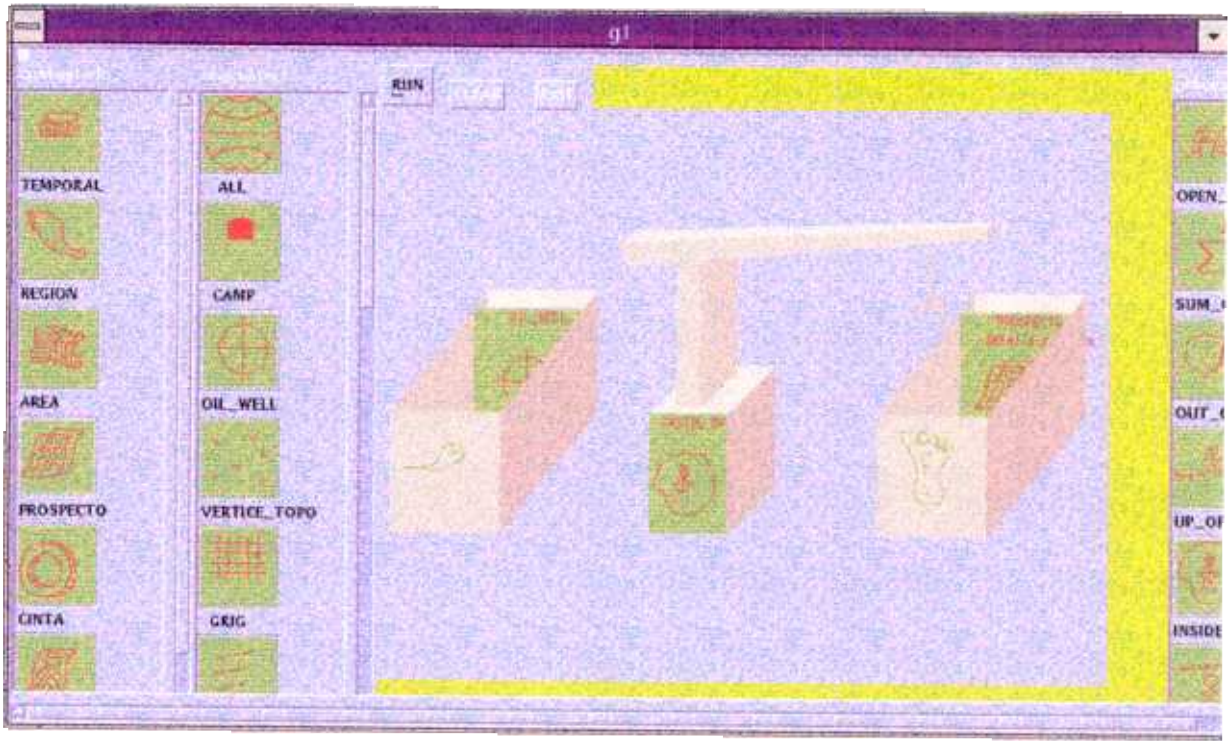


Figura 6-9 Sobre la cara del elevador, se le asigna un objeto activo "INSIDE OF". Y una sentencia de consulta visual : ¿Qué pozos petroleros se encuentran en el proyecto "DETALLE_DR_COSS"?

Ahora, nosotros tenemos construido un VQS en el área de consulta con el lenguaje CQL.

Después de programado, podemos ejecutar el programa mediante el clic el botón "RUN" que llama un menú como el que se muestra en la Figura 6-10.

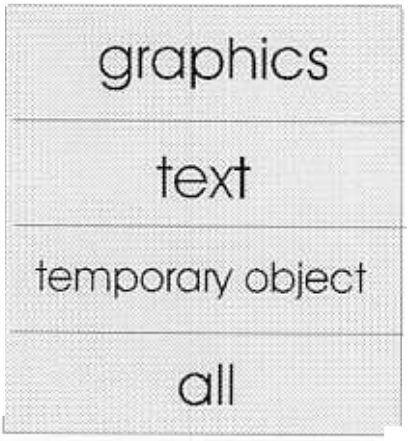


Figura 6-10 el menú de salidas

En el menú, se observan cuatro opciones para cada uno de las cuatro tuberías que se despliegan en los resultados de las consultas.

- **GRAPHICS** .- Significa que el resultado de la consulta podría ser desplegado gráficamente en el área de gráficas.
- **TEXT.-** Significa que el resultado se presentará textualmente.
- **TEMPORARY OBJECT**.- Significa que los resultados de la consulta se salvará en memoria como un objeto temporal. En la siguiente sección se discutirá más sobre los objetos temporales.
- **ALL**.- Significa que el resultado de la consulta se desplegará de las tres maneras descritas anteriormente.

En una palabra, nuestro propósito es proporcionar un sistema de consulta gráfico, con una combinación de tarjetas obteniendo un ambiente integrado.

- Cada entidad esta representada por una tarjeta
- Propiedades (i.e. atributos) de entidades son indicadas por nombres y parámetros de las tarjetas.
- Cada relación esta representada por una combinación de tarjetas.

6.3 ACERCA DE LA ENTIDAD "OBJETO TEMPORAL"

En relación a la solución de los problemas de sub-consultas en el lenguaje CQL empleamos una tarjeta llamada "**OBJETO_TEMPORAL**".

Supóngase que **Y** es una consulta. Para obtener esa consulta es necesario hacer **n** subconsultas. Se expresa matemáticamente como:

$$Y = f^n(x) = f^n(f^{n-1}(..f^1(x)))$$

Los subconsultas se pueden presentar en la forma:

$$y_1 = f^1(x)$$
$$y_2 = f^2(y_1)$$
$$y_3 = f^3(y_2)$$
$$\vdots$$
$$y_{n-1} = f^{n-1}(y_{n-2})$$
$$Y = y_n = f^n(y_{n-1})$$

En **CQL**, se usa la tarjeta **objeto_temporal** para representar los resultados de subconsultas, y_1, y_2, ..., y_{n-1}. Y se usan los **VQS1, VQS2, ..., VQSn** para construir n subconsultas, como:

VQS_1: $f^1(x) \longrightarrow$ **objeto_temporal$_1$**
VQS_2: f^2 (**objeto_temporal$_1$**) $\longrightarrow$ **objeto_temporal$_2$**
VQS_3: f^3 (**objeto_temporal$_2$**) $\longrightarrow$ **objeto_temporal$_3$**
$\vdots$
VQS_{n-1}: f^{n-1} (**objeto_temporal$_{n-2}$**). **objeto_temporal$_{n-1}$**
VQS_n: f^n (**objeto_temporal$_{n-1}$**) . **Y**

La tarjeta **objeto_temporal** puede representar un numérico, una cadena, un registro, un archivo, o un conjunto de cada uno de ellos. El contenido fisico de **objeto_temporal** es un subconjunto de los datos originales de las tarjetas que se definieron en BOX3 dentro de un VQS.

objeto temporal $\subseteq$ los datos originales que definieron de las tarjetas en BOX3

Por ejemplo, el usuario final quiere seleccionar la entidad **río** que se llama "SAN JUAN" del proyecto "DETALLE_DR_COSS". El archivo original del proyecto DETALLE_DR_COSS se muestra en la tabla 6-1.

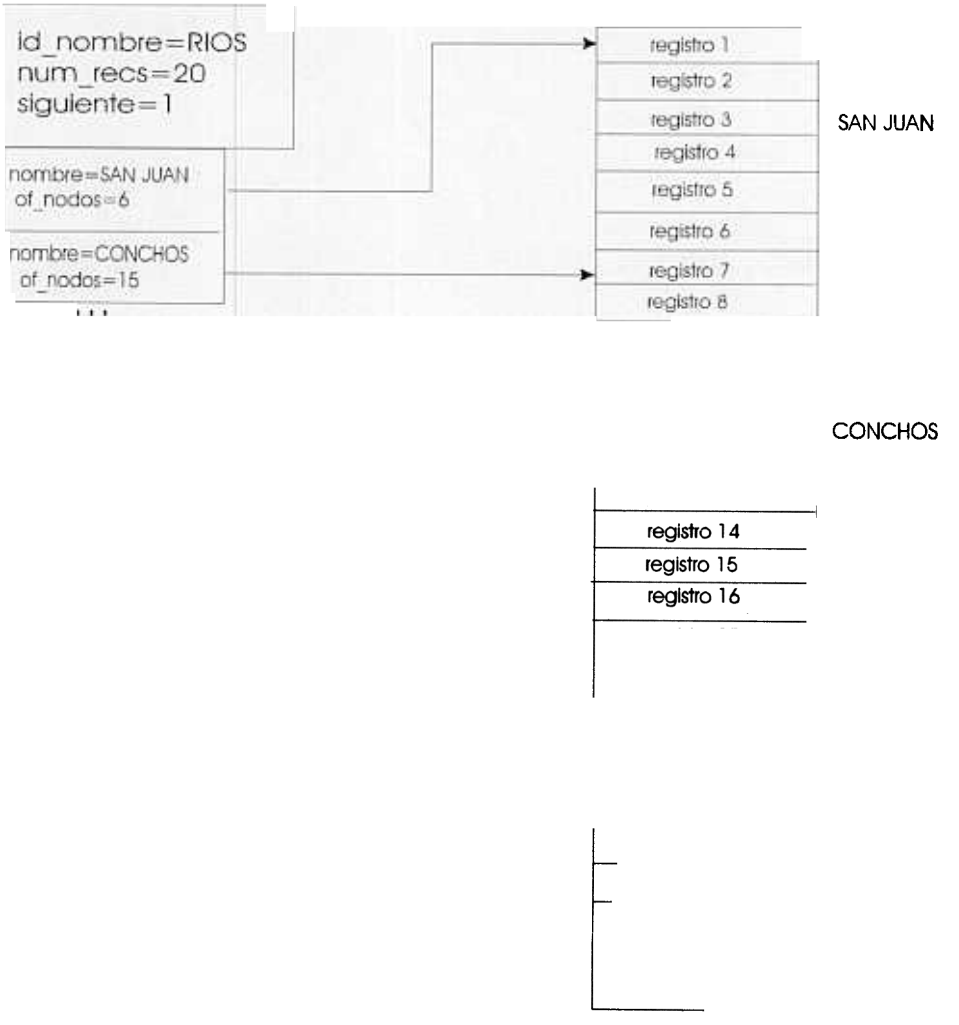


Tabla 6-1 Archivo original de DETALLE_DR_COSS

Después de ejecutar la sentencia de consulta, la cual se define como la dirección de salida del resultado de la consulta para el **objeto_temporal**, el archivo de objeto temporal podría ser llenado en los registros de río de SAN JUAN de el proyecto DETALLE_DR_COSS como se muestra en la tabla 6-2.

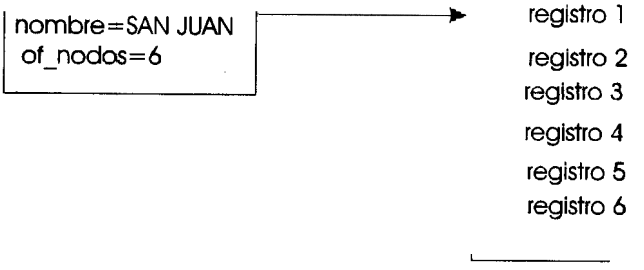


Tabla 6-2 Registros en el objeto temporal

6.4 IMPLEMENTACIÓN DEL CQL

En esta sección vamos a explicar más en detalle las estructuras de procesamiento del CQL

En la figura 6-11 se presenta una visión general del propósito del sistema del procesamiento llevado a cabo en una consulta, mostrando las principales funciones del sistema, la información sobre flujos y las librerías principales.

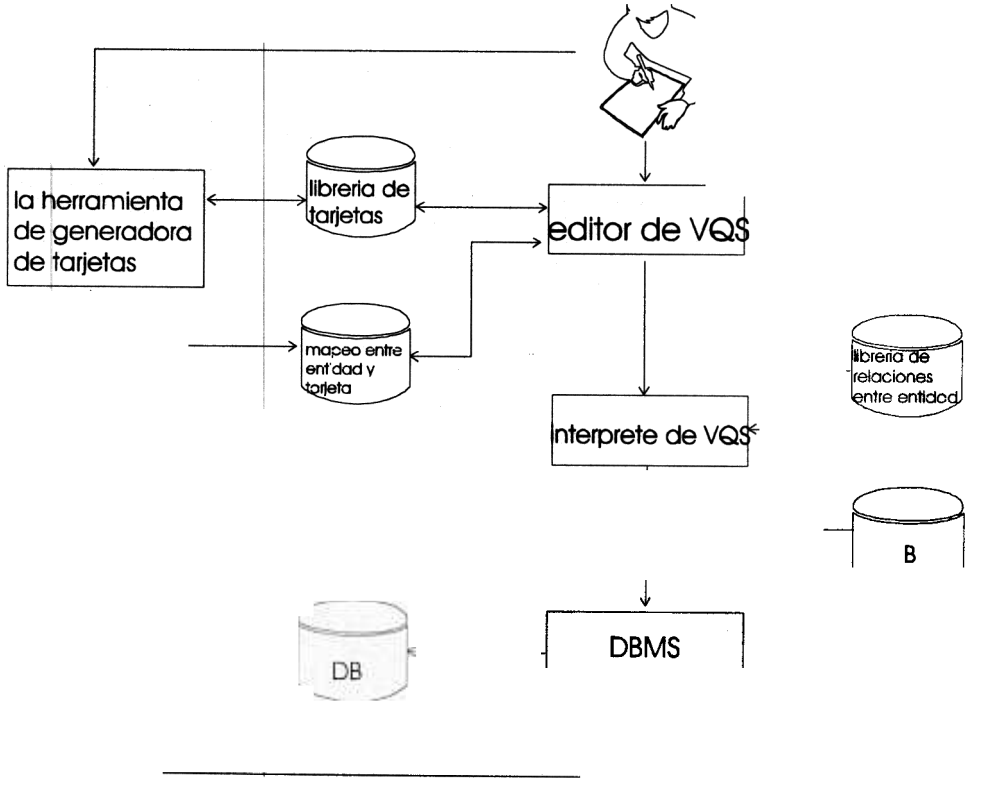


Figura.6- diagrama de procesamiento de una consulta

La herramienta generadora de tarjetas es empleada por las entidades y el mapeo entre entidades y tarjetas. Esta herramienta permite al usuario definir o modificar las tarjetas, así como actualizaciones a las librerías de las tarjetas y mapeo de tarjetas-entidades. Cuando introducimos los datos del archivo, el sistema automáticamente examina si estas entidades existen en las librerías de las tarjetas, en caso de no ser así una nueva tarjeta es invocada para ser generada. Durante el mapeo, el usuario puede observar que la entidad anterior tiene una relación con la actual.

Durante la interpretación , una estructura que represente una consulta en VQS es analizada y segmentada en sus elementos primitivos con ayuda de G y CD. Para cada primitiva se busca su parte lógica(significado) en B y se conjugan siguiendo un conjunto de reglas propias del interprete para construir la parte lógica de la consulta. Finalmente, usando esta información se construye el significado de la estructura original y se realizan las operaciones pertinentes.

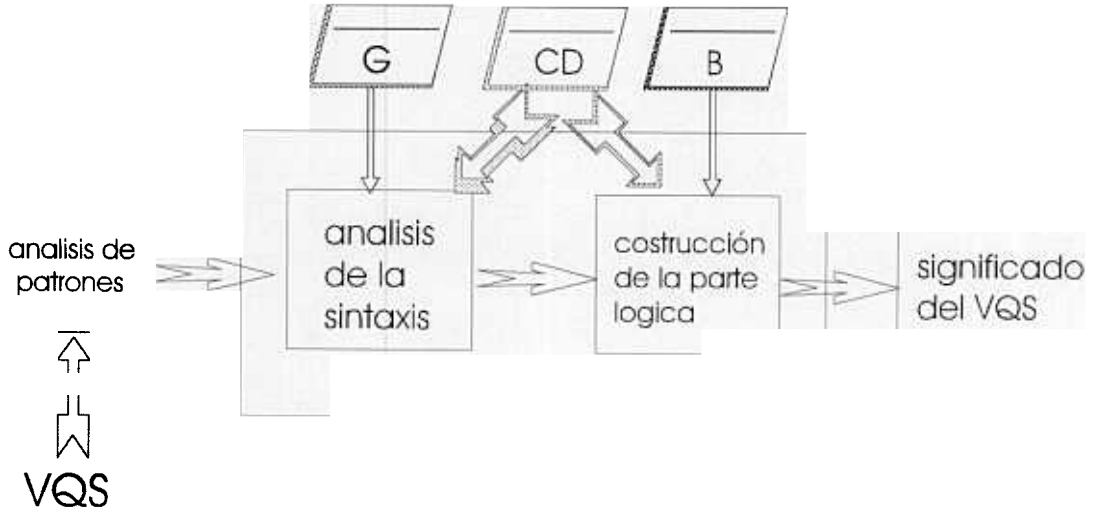


Figura 6-12 Interprete de VQS

En la figura 6-12, G es la gramática gráfica del sistema de tarjetas, B es base de conocimiento, y CD es el diccionario de tarjetas. El diccionario de tarjeta de entidad almacena las informaciones como tabla 6-3.

Tabla 6-3: estructura de diccionario de tarjeta de las entidades

No. de tarjeta	apuntador de imagen de la tarjeta Xi	nombre de la tarjeta Xm	apuntador de la tabla o archivo
2_1	→	región	A→
2_5	→	prospecto	B→
..	..	..	..
1_7	→	río	C→

A→ archivo de REGION
B→ archivo de PROSPECTO
C→ tabla de nombre de archivo que tiene la entidad

Todas las tarjetas se incluyen en el diccionario. Cuando el usuario da de alta las entidades, el sistema llena automáticamente esta tabla.

Para las tarjeta de proceso , el diccionario tiene algunas diferencias. La estructura se muestra en la tabla 6-4.

No.de tarjeta	apuntador de imagen del tarjeta Xi	nombre de la tarjeta Xm	apuntador de subrutinas de proceso
3_1	→	OPEN	→
3_2	→	NEAR_OF	→
...	...	...	...
3_n	→	INSIDE_OF	→

Tabla 6-4: estructura del diccionario de tarjeta de procesos

6.5 EJEMPLOS DE CONSULTA

En esta sección, presentamos algunos ejemplos de consultas con CQL.

Q1: solicita la recuperación de la información del usuario: ¿Cuáles son los pozos petroleros del proyecto DETALLE_DR_COSS?

El VQS en CQL tal como se muestra en la figura 6-13. La consulta esta definida por la combinación de un proceso de tarjetas (DENTRO_DE) sobre la cara del elevador, una tarjeta de objeto real (OIL_WELL (pozo petrolero)) en la caja 1 y una tarjeta de objeto conceptual (PROYECTO con parámetro "name=DETALLE_DR_COSS") en la caja 2. Si definimos la salida como texto, al mismo tiempo, en el área de texto la información de los pozos petroleros será listada.

Después de ejecutar el VQS, el cual define la salida a las gráficas, el resultado de la consulta será desplegado en forma de un mapa topográfico en el área gráfica de la pantalla, como se muestra en la figura 6-13. Dentro del proyecto "DETALLE_DR_COSS", se encuentran cuatro pozos petroleros, el mapa es dibujado de acuerdo a las coordenadas de cada pozo petrolero en el archivo de datos.

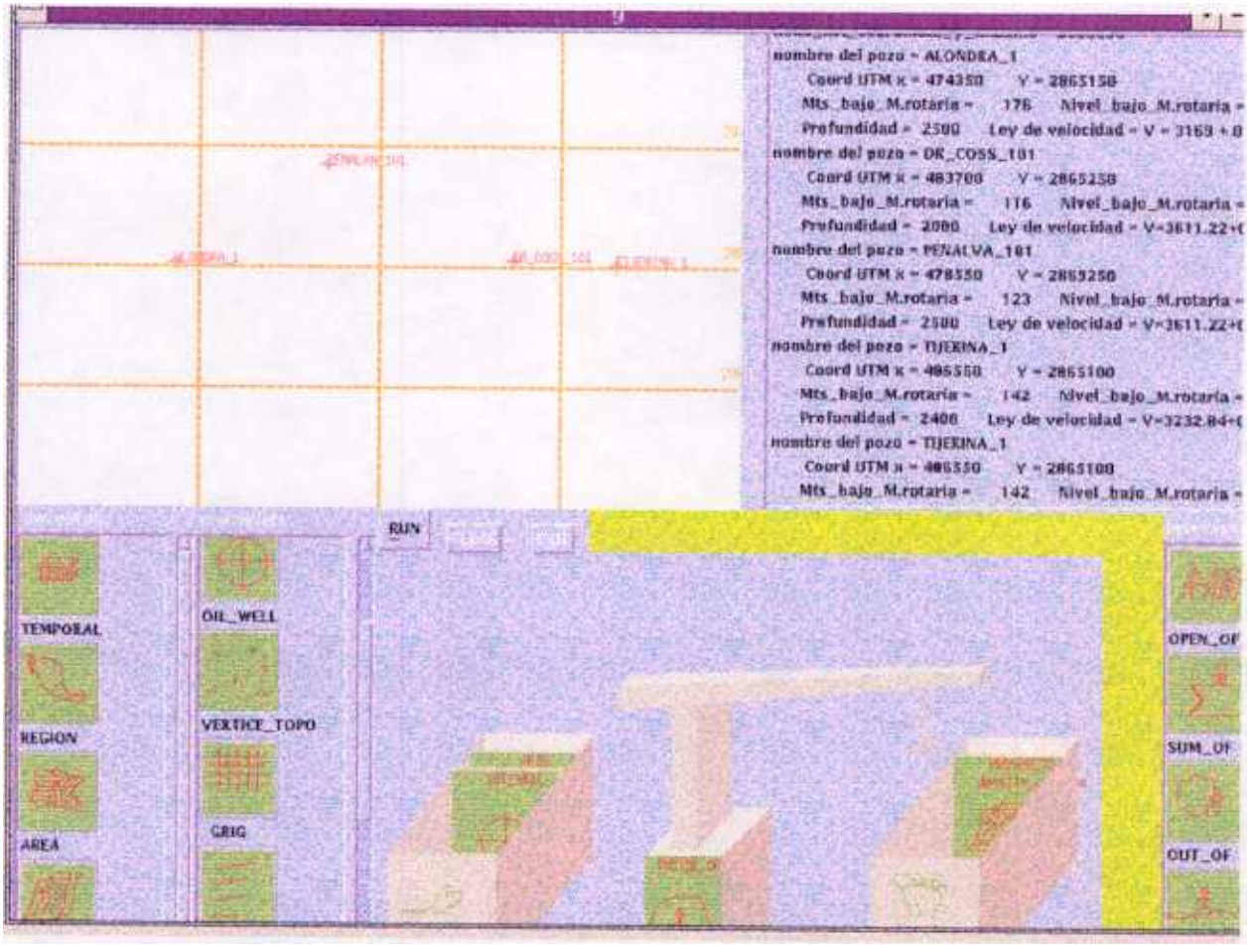


Figura 6-13 Q1 Muestra cuales son los pozos petroleros del proyecto DETALLE_DR_COSS

Q2: solicitud de la recuperación de información del usuario:
¿ Cuáles son los pozos petroleros y aldeas dentro del proyecto DETALLE_DR_COSS?

La consulta esta definida por la combinación de tarjetas de procesos (DENTRO DE) en el elevador, dos tarjetas de objetos reales (OIL_WELL, POPULATION) en la caja 1; y una tarjeta de objeto conceptual (PROYECTO con parámetro "name=DETALLE_DR_COSS") en la caja 2.

Después de presionar el botón "RUN" y definir la salida como ALL, el resultado de la consulta es desplegado en forma de mapa topográfico en el área de gráfica, y en el área de texto aparece una lista, como se muestra en la figura 6-14.

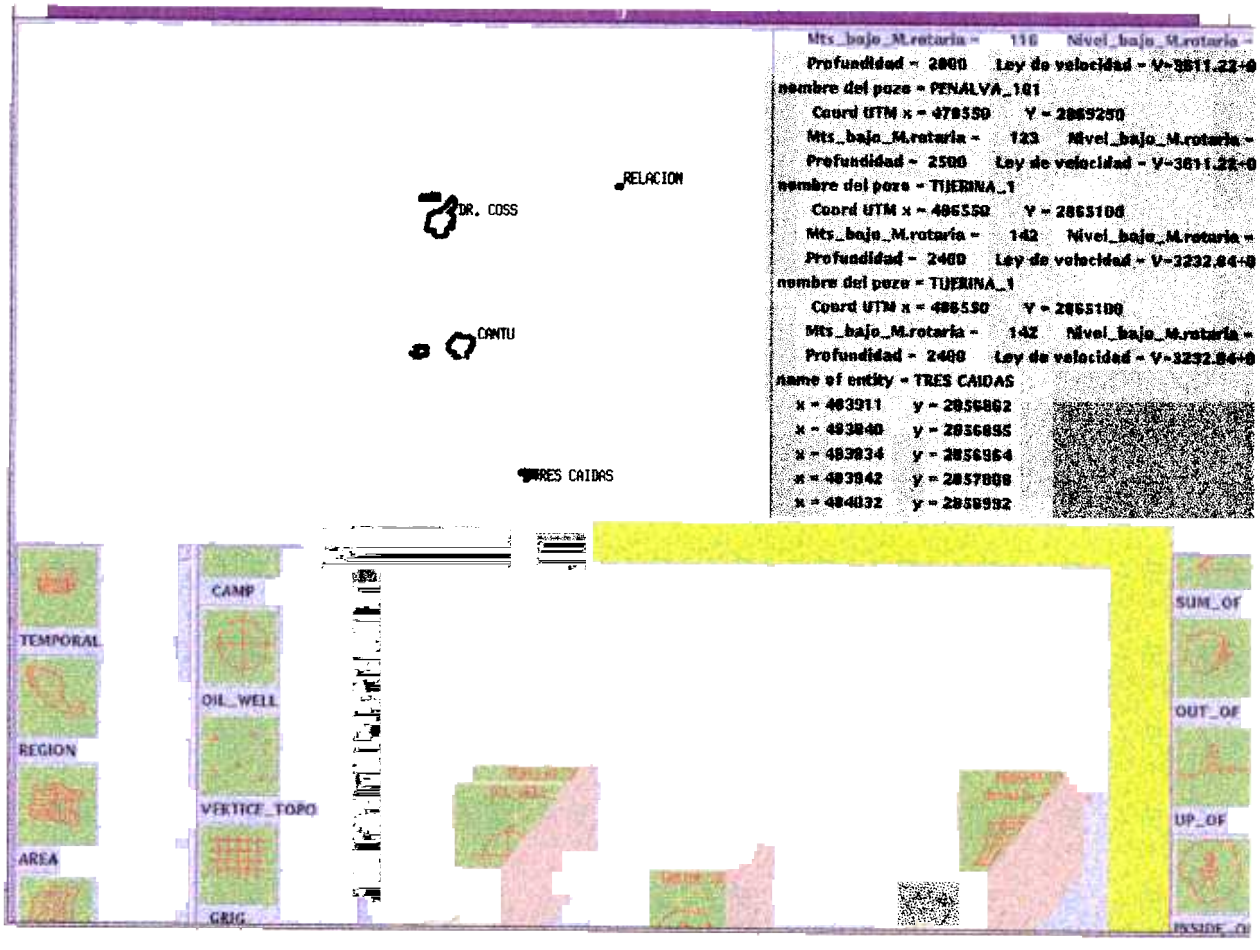


Figura 6-14 Q2: ¿Cuáles son los pozos petroleros y aldeas que se encuentran dentro del proyecto DETALLE_DR_COSS?

Q3: Solicitud de recuperación de información por el usuario:
¿Cuáles son las aldeas, carreteras y ríos que se encuentran dentro del proyecto DETALLE_DR_COSS?

La consulta fue definida por una combinación de tarjetas de proceso (DENTRO DE) en el elevador, las tarjetas de objetos reales (POBLACIÓN, CARRETERAS Y RÍOS) en la caja 1; y una tarjeta de objeto conceptual (PROYECTO con parámetro "name = DETALLE_DR_COSS) en la caja 2.

El resultado de la consulta es desplegado en forma de mapa topográfico en el área gráfica de la pantalla, tal como se muestra en la figura 6-15.

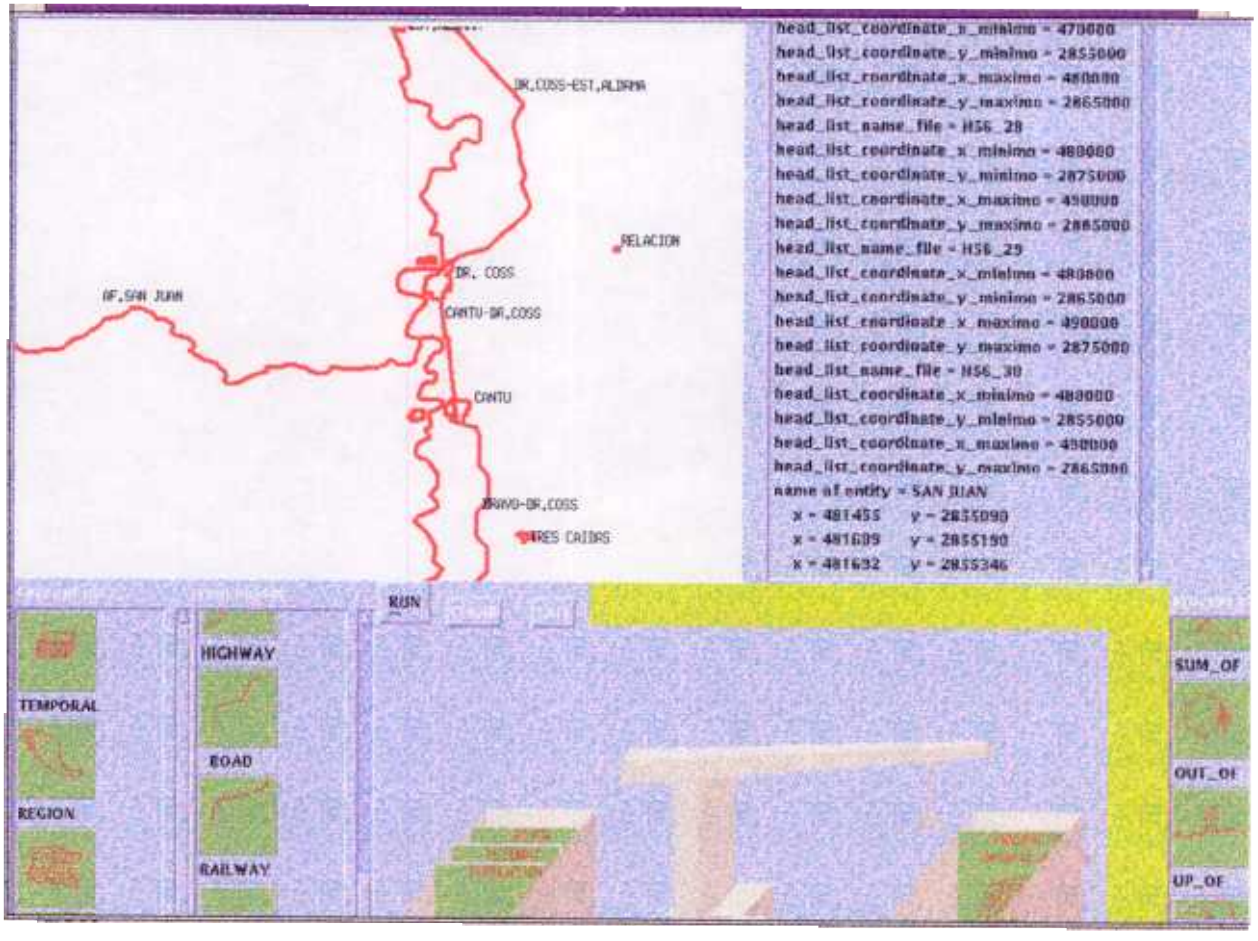


Figura 6-15 Q3: ¿Cuáles son las aldeas, carreteras y ríos
que se encuentran dentro del proyecto DETALLE_DR_COSS?

Q4: solicitud de obtención de información por el usuario:
Despliega el mapa del proyecto "DETALLE_DR_COSS"

El VQS en CQL es programado por la combinación de tarjetas de proceso (ABRIR) en el elevador, una tarjeta de objeto real (TODO) en la caja 1; y una tarjeta de objeto conceptual (PROYECTO con parámetro "name= DETALLE_DR_COSS") en la caja 2; El resultado de la consulta se despliega en modo gráfico. El mapa incluye todas las entidades de la región, como se muestra en la figura 6-16.

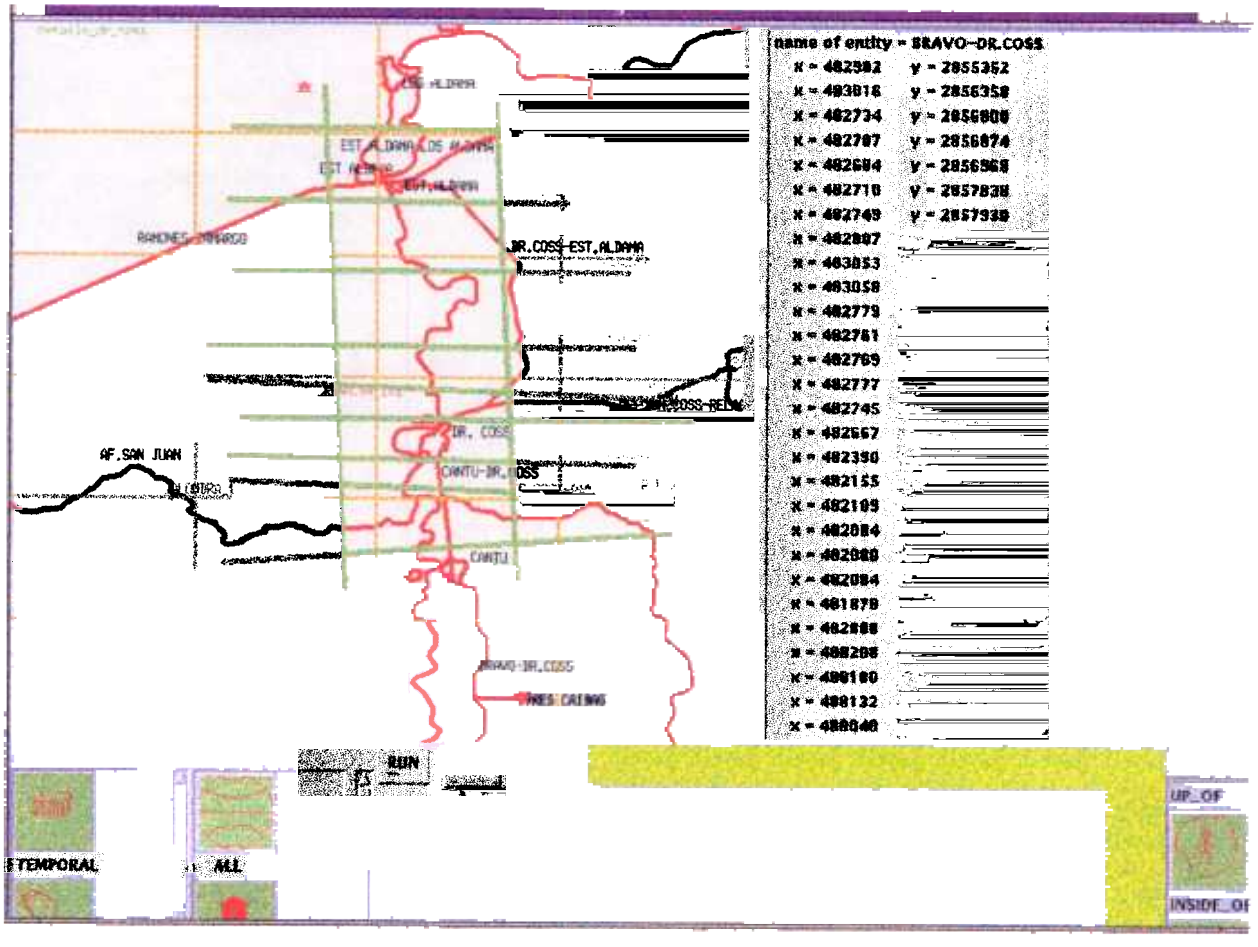


Figura 6-16 Q4: Despliega el mapa del proyecto "DETALLE_DR_COSS"

CONCLUSIÓN

En este capítulo se ha presentado un ambiente gráfico completamente para consulta a la base de datos con CQL. El propósito de CQL es proporcionar las siguientes ventajas para el usuario final:

(1) Determinar fácilmente las tarjetas de las consultas usando la imagen Xi.
(2) Sin estar familiarizado con la base de datos y su estructura, se puede manejar la base de datos rápidamente.
(3) El lenguaje tiene varias maneras de tratar la representación de los resultados de la consulta.
(4) Este propone un nuevo algoritmo -- un objeto temporal para sub-consultas.
(5) Se permite al usuario final desplegar información visual de la base de datos.

CAPÍTULO 7

COMPARACIÓN EL LENGUAJE CQL CON SQL

Los sistemas comerciales de base de datos requieren un lenguaje de consultas más amigable para el usuario. Los lenguajes: **SQL**[RAYM91], **QBE** y **Quel** representan una variedad de estilos. **QBE** está basado en el cálculo relacional de dominios; **Quel** está basado en el cálculo relacional de tuplas y **SQL** usa una combinación de construcciones del álgebra relacional y del cálculo relacional[HENR93]. El **CQL** es un lenguaje de consulta visual basado en el modelo relacional.

El objetivo de este capítulo es estudiar algunos casos de consultas que son formúladas con la "maquina de consulta" y comparar CQL con SQL. Además, reportamos algunos resultados del lenguaje CQL con el fin de mostrar sus capacidades y contribuciones en estudios comparativos con casos.

El lenguaje SQL consta tres partes:

(1). El lenguaje de definición de datos que permite definir esquemas y subesquemas.
(2). El lenguaje de manejo de datos, usado para modificar y actualizar datos.
(3). El lenguaje de consulta que permite resolver problemas de preguntas a la base de datos.

En el sistema VISDA, el lenguaje de definición de datos que se usa es EVE, que se describió en el capítulo 2. El lenguaje de manejo de datos es una parte independiente que se explicó en el capítulo 4. Entonces solo necesita comparar el lenguaje CQL con la parte de consulta de SQL.

Una diferencia significativa entre SQL y CQL, es que en SQL las únicas relaciones entre entidades que se trabajan están definidas implícitamente en las tablas. Mientras que en CQL, además se tiene la posibilidad de manejar relaciones espaciales entre entidades.

En términos normales, una relación (simple) se refiere al hecho de tener accesibilidad a los atributos de n entidades de manera controlada. La creación de una relación nos lleva a la creación de una nueva tabla, como se muestra en la figura 7-1.

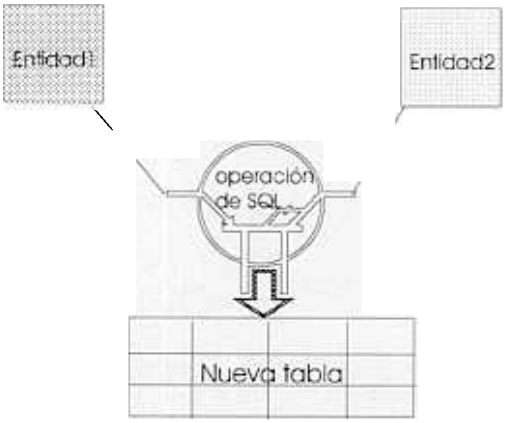


Figura 7-1 Se hace una operación sobre dos entidades a crear una nueva tabla en SQL.

Aquí, no es importante clasificar las tablas creadas. Cuando las relaciones entre entidades necesitan ser clasificadas, por ejemplo, relaciones de inclusión (INSIDE_OF), o de ubicación (UP_OF,BELOW_OF,...), es necesario proporcionar algún método para esta identificación, de manera que se tenga acceso a estos datos cuando el sistema recuperala información relacionada espacialmente, ver figura 7-2.

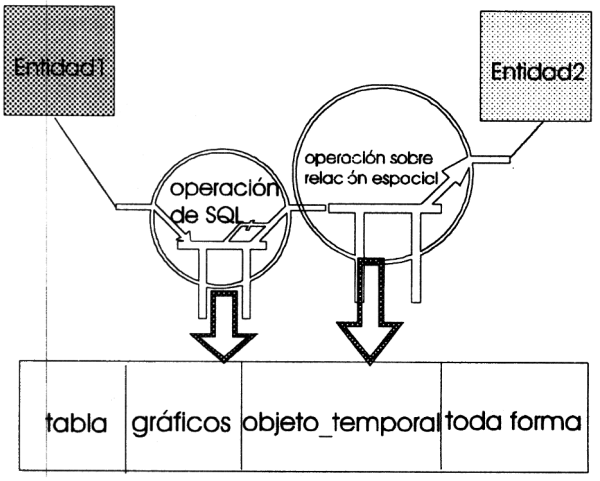


Figura 7-2 Recuperar información relacionada espacialmente y nominalmente en CQL

En este capítulo, tomando como base un ejemplo de exploración petrolera, realizamos una comparación entre construcciones de SQL (en forma de texto) y construcciones de CQL (en forma gráfica). Iniciamos con el formato de cada construcción.

7.1 Comparación las estructuras

La estructura básica de una expresión en SQL tiene la forma:

select $\mathbf{A_1,A_2,...,A_n}$
form $\mathbf{r_1,r_2,...,r_m}$
where **p**

En el lenguaje **CQL**, se usa un mecanismo básico llamado "**máquina de consulta**" que consta de tres cajas: **BOX1, BOX2 y BOX3**, como muestra en la figura 7-3 siguiente.

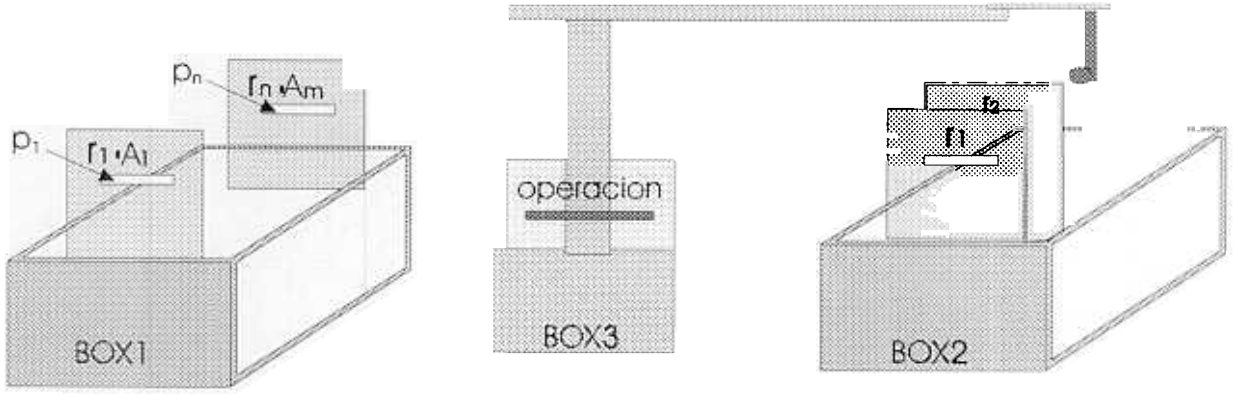


Figura 7-3 La programación en CQL

La **BOX1** se usa para construir una lista de tarjetas que representan los objetos que se desean en el resultado de una **VQS**. Las tarjetas pueden representar atributos de registro, registros o archivos (como se definió en el capítulo 5). El **BOX1** corresponde a la cláusula **select**, y las tarjetas dentro de él corresponde a la lista de atributos $\mathbf{A_1,A_2,...,A_n}$.

La **BOX2** se usa para construir una lista de las tarjetas que representa a los objetos que se van a examinar en la evaluación de la expresión. Las tarjetas pueden representar atributos de registro, registros o archivos . El **BOX2** corresponde a la cláusula **from**, y las tarjetas dentro de él corresponden a la lista de atributos $\mathbf{r_1,r_2,...,r_m}$.

La predicado P en la cláusula **where** es distribuido en cada tarjeta en forma de los parámetros para cada tarjeta.

La **BOX3** se usa para construir una lista de tarjetas de proceso que representan una acción, un procesamiento o una recuperación de una relación espacial entre las entidades que están definidos en **BOX1** y las entidades que están definidos en **BOX2**

7.2 COMPARACIÓN DE LAS EXPRESIONES DE LAS FUNCIONES

En el lenguaje **SQL,** las consultas se clasifican en:
(1) consulta básica
(2) funciones de agregación(sum,max,min,avg)
(3) subconsulta
(4) exists,in, not exists, not in, join
(5) operación de conjuntos: unión.
(6) group by, order by.
(7) creación de vista

En seguida se comparan las funciones entre **SQL** y **CQL.** Para una fácil comparación, supondremos una empresa de PEMEX con el diagrama entidad-relacion que se muestra, en el capítulo 2, figura 2-4.

Esquema **AREA** = (nom_area,nom_reg, descripción)
Esquema **PROSPECTO** = (nom_prosp,nom_area,clave_brig,descripcion)
Esquema **POZO** = (nom_pozo,x,y,nom_prosp,
clave_brig,fecha,prof_total)
Esquema **BRIGADA** = (clave_brig,compania,jefe_brig,ubicacion,tel_brig,ETE_energia)
Esquema **HOJAPROS** = (nom_hoja,nom_prosp,descripcion).

Las relaciones **AREA, PROSPECTO, POZO, BRIGADA, HOJAPROS**, se muestran respectivamente en las tablas 7-1,7-2,7-3 , 7-4 y 7-5

En esta sección y los subsiguientes, las consultas se basan en estas relaciones

nom_area	**nom_reg**	**descripcion**
acapulco	centro	simulacion
camargo	norte	simulación
el_chichon	sur	simulación

Tabla 7-1 la relación **AREA**

nom_prosp	**nom_area**	**clave_brig**	**descripción**
detalle_dr_coss	camargo	nes_9	verdad
ju1	camargo	nes_8	simulación
ju2	acapulco	nes_7	simulación

Tabla 7-2 La ralación **PROSPECTO**

nom_pozo	x	y	nom_prosp	clave_brig	fecha	prof_total
cerro_nanchital	313000	1221000	detalle_dr_coss	nes_9	19850301	4567
dr_coss_101	483700	2865250	detalle_dr_coss	nes_9	19851101	2611
el_plan	337000	1225000	detalle_dr_coss	nes_8	19850201	4786
[illegible]_101	478550	2869250	detalle_dr_coss	nes_8	19851001	3611
tijerina_1	486550	2865000	detalle_dr_coss	nes_8	19860301	3232
carmen_1	480000	2800000	carmen	nes_8	19841101	4000
carmen_2	481000	2810000	carmen	nes_8	19870201	4000

Tabla 7-3 La relación POZO

clave_brig	compania	jefe_brig	ubicación	tel_brig	ETE_energia
nes_8	PEMEX	José Rivas	cd_carmen	52413	minas
nes_9	PEMEX	Juan Pérez	villahermosa	65342	minas
nes_7	PEMEX	ricardo López	villahermosa	87654	minas

Tabla 7-4 La relación BRIGADA

nom_hoja	nom_prosp	descripción
h55_28	detalle_dr_coss	de DETENAL
h55_29	detalle_dr_coss	de DETENAL
h55_30	detalle_dr_coss	de DETENAL
h56_28	detalle_dr_coss	de DETENAL
h56_29	detalle_dr_coss	de DETENAL
h56_30	detalle_dr_coss	de DETENAL
h56_32	ju1	de simulación
h56_33	ju2	de simulación

Tabla 7-5 La relación HOJAPROS

En la tabla 7-5, los nombres de hojas son exactamente los nombres de los archivos gráficos. Para cada hoja de mapa le corresponde dos archivos con extensiones *.DEF y *.GRA, como explicó en el capítulo 3. Para entender facilmente las consultas, presentamos una parte de los datos de archivo h55_28 en las tablas 7-6 y 7-7.

id_nombre = RIOS
num_recs = 20
siguiente = 1
nombre = SAN JUAN
of_nodos = 6
nombre = CONCHOS
of_nodos = 15
.
.
.
id_nombre = CARRETERAS
num_recs_30
siguiente = null
.
.
.
nombre = CANTU
of_nodos = 350

Tabla 7-6 El archivo **h55_28.DEF**

Entidades	No. de puntos	x coordenadas	y coordenadas	parámetro
rio SAN JUAN	1	479610.8	2878672	1
	2	479580.9	2878867	1
	3	479579.7	2879029	1
	4	479598.3	2879150	1
	5	479601.4	2879152	1
	6	479603.7	2879160	0
	.	.	.	
	.	.	.	
carretera CANTU	1	481060.3	2865002	1
	2	480977.6	2865042	1
	3	4808936	2865083	1
	4	480798.3	2865148	1
	.	.	.	
	.	.	.	
	.	.	.	
	350	488967.4	2869002	0

Tabla 7-7 El archivo **h55_28.GRA**

7.2.1 comparación entre operaciones fundamentales

Vamos a demostrar cómo se pueden escribir en SQL y CQL las consultas de los ejemplos. Consideradas en la base de datos que se definió anteriormente, hacemos una consulta Q1:

Q1 con lenguaje natural: ¿Quién es jefe de brigada ns_9?

Q1 con lenguaje SQL:

```
select  jefe_brig
from    brigada
where   clave_brig = 'nes_9'
```

Q1 con lenguaje CQL:
En el lenguaje **CQL**, se usa un **VQS** para la consulta. El **VQS** incluye dos tarjetas. En **BOX1**, la tarjeta con parámetros corresponde la lista, **brigada.jefe_brig**, de la cláusula **select**. En **BOX2**, la tarjeta corresponde a la lista **brigada** de la cláusula **from** y el parámetro corresponde a la lista, **clave_brig = 'nes_9'**, de la cláusula **where**

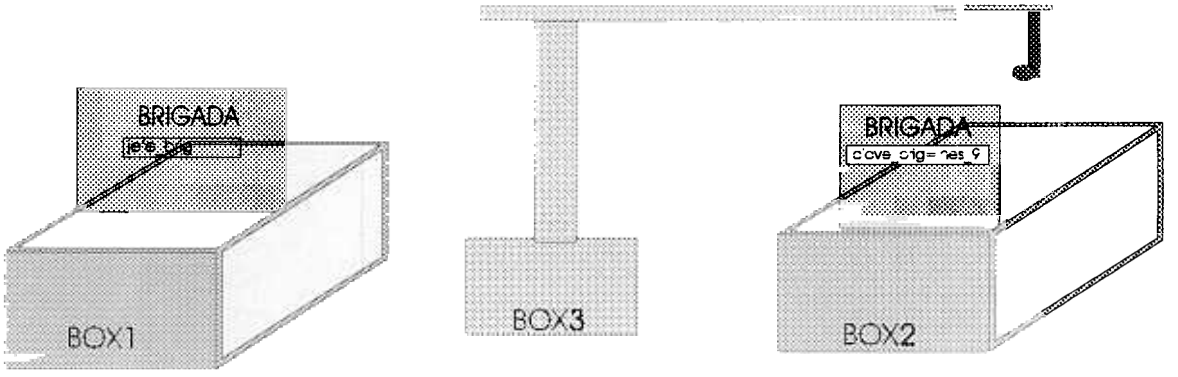


Figura 7-4 Q1: Quién es jefe de brigada ns_9?

el resultado del consulta en CQL será un lista en la área de texto como

clave_brig: **nes_9**
jefe_brig: **Juan Pérez**

7.2.2 comparación de las funciones de agregación

El CQL incluye funciones dentro de los parámetros de la tarjeta para calcular:
* promedio: avg
* mínimo: min
* máximo: max
* total: sum
*contar: count
Las funciones son equivalentes a las de SQL. Otro ejemplo de consulta es Q2

Q2 con lenguaje natural: Encontrar el pozo que se creó al último.

Q2 con lenguaje SQL:

select *max*(fecha)
from pozo

Q2 con lenguaje CQL:

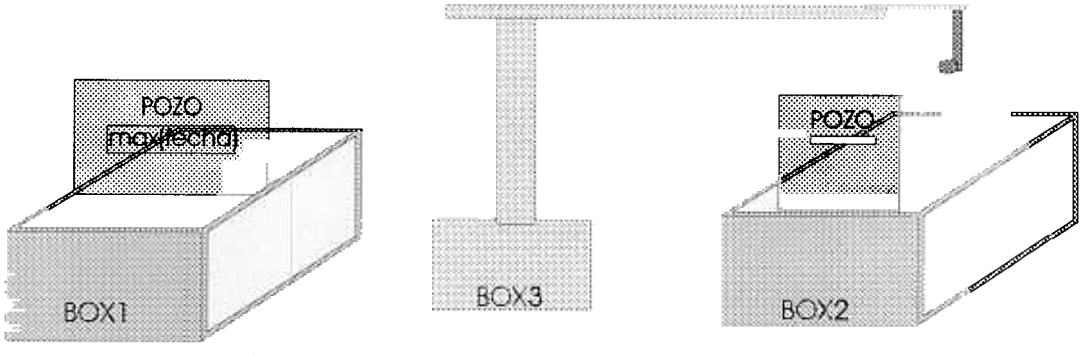


Figura 7-5 Q2: Encontrar el pozo que se creyó último fecha

.En **CQL**, la tarjeta en **BOX1** representa el campo **pozo.fecha** y la función **max** representa para calcular máximo sobre ese campo. La tarjeta en BOX2 representa que recupera la información desde la entidad **pozo**.

7.2.3 comparación de subconsultas con el objeto_temporal

En el CQL se emplea un mecanismo llamado "**objeto_temporal**" para solucionar el problema de una subconsulta. El objeto_temporal físicamente es un archivo temporal para guardar la información intermedia de un serie de subconsultas. El objeto_temporal puede ser un conjunto de campos, registros, o archivos dependiendo del resultado de la subconsulta. Al objeto_temporal se le puede asignar un nombre temporal dado por usuario. Usamos el ejemplo Q3 para mostrar cómo funciona el objeto_temporal y su equivalencia a una subconsulta.

Q3 con lenguaje natural: Encontrar los nombres de todos los pozos que tienen fecha atrás de algún pozo situado en el prospecto "detalle_dr_coss" y que se creó en 1985.

Q3 con lenguaje SQL:

```
select nom_pozo
from pozo
where fecha > some
      (select fecha
       from   pozo
       where  nom_prosp = 'detalle_dr_coss'
       AND fecha > '841231'
       AND fecha < '860101')
```

La subconsulta:

```
select fecha
from   pozo
where  nom_prosp = 'detalle_dr_coss'
AND fecha > '841231'
AND fecha < '860101'
```

generará el conjunto de **fecha** que cumplió las condiciones de la cláusula de **where.**

Q3 con lenguaje CQL:

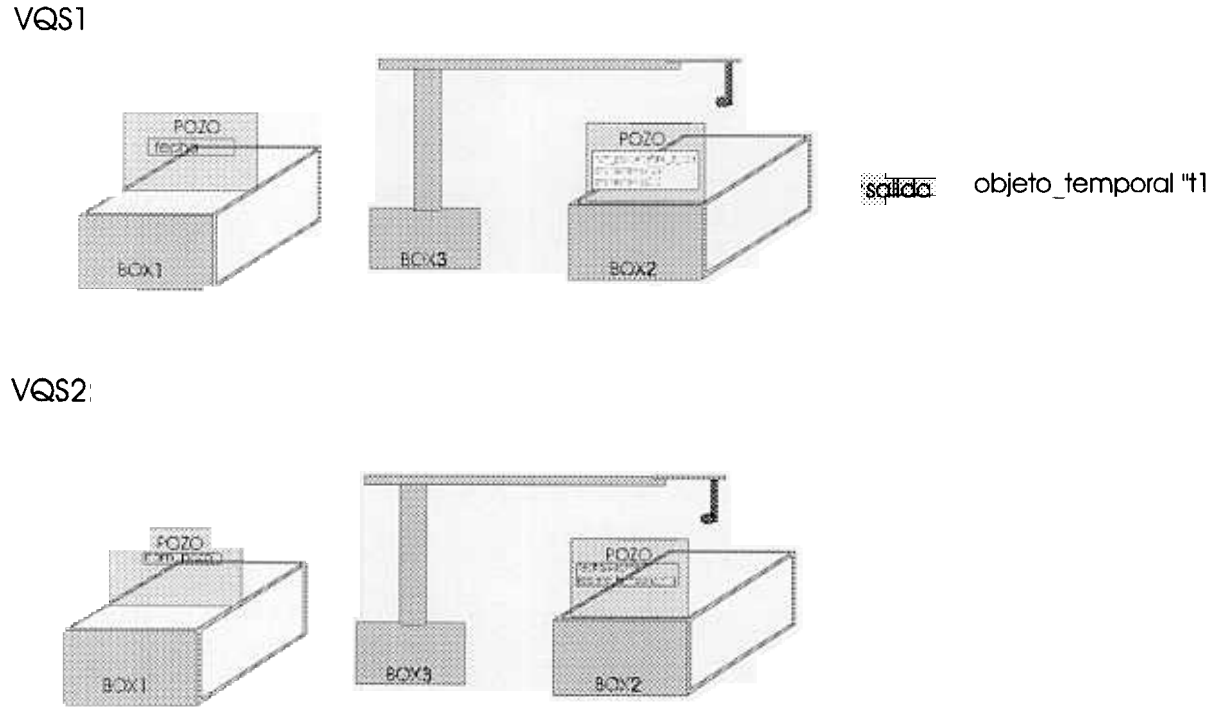


Figura 7-6 Q3: Encontrar los nombres de todos pozos que tiene la fecha atrás de alguno pozo que situada en prospecto "detalle_dr_coss" y creó en 1985.

En ésta consulta, se usan dos VQSs en lenguaje CQL. El VQS1 corresponde a la subconsulta de SQL. Se recuperan los dominios de **fecha**, y la salida de esa subconsulta se almacena en objeto_temporal y se le denomina "t1". El VQS2, como segundo paso de esa consulta, representa el procesamiento de recuperar los registros que su valor de campo, **fecha,** mayor que el valor de objeto_temporal "t1".

7.2.4 comparación de las funciones de la presentación

El SQL ofrece al usuario cierto control sobre el orden en el que se van a presentar las túplas en una relación. La cláusula **order by** hace que las túplas en el resultado dé una consulta salgan en un orden determinado. La cláusula **group by** se usa para formar grupos.

En CQL se tienen las funciones equivalente. El ejemplo Q4 muestra las representaciones de distintos lenguajes.

Q4 con lenguaje natural: Obtener una lista de nombres de pozo y clave de brigada agrupada por clave de brigada.

Q4 con lenguaje SQL:

```
select nom_pozo,clave_brig
from pozo
group by clave_brig
```

Q4 con lenguaje CQL:

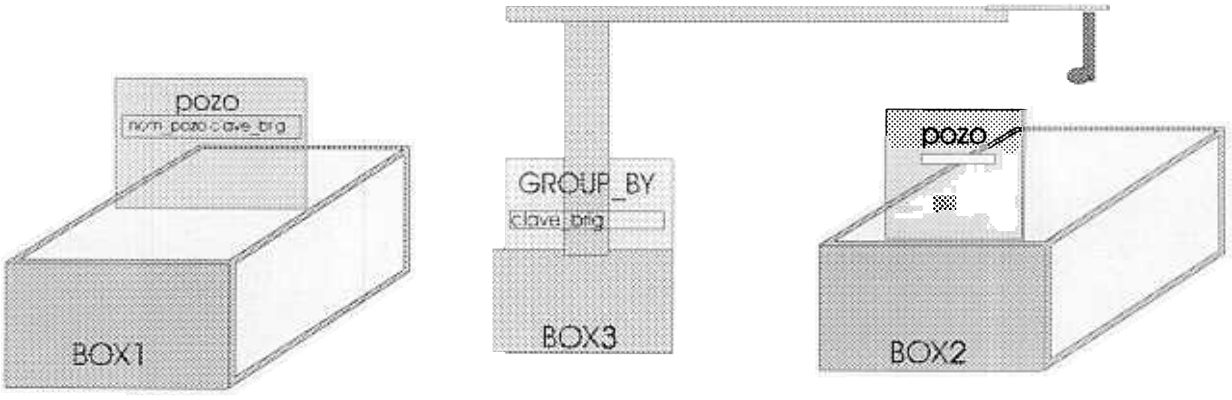


Figura 7-7 Q4: Obtener una lista de nombres de pozo y clave de brigada agrupada por clave de brigada.

En **CQL** se usa la tarjeta **GROUP_BY** que se funciona equivalente a la de **SQL**. En esa consulta, el **BOX3** se metió la tarjeta **GROUP_BY** y con parámetro **clave_brig**, lo que quiere decir agrupar el resultado en base al dominio **clave_brig.**

7.2.5 comparación de las operaciones de conjuntos

En algunas versiones de SQL incluyen las operaciones union,intersect y minus (no exist en versión estándar), que operan sobre relaciones y corresponden a las operaciones del álgebra relacioal $\cup$,$\cap$ y —. Vamos a demostrar, en Q5, cómo se pueden escribir en CQL las consultas de los ejemplos considerados anteriormente.

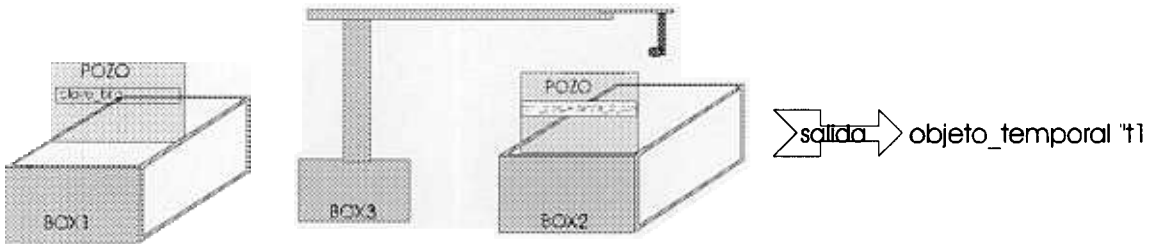
Q5 con lenguaje natural: Encontrar a todos las brigadas que se crearon en un pozo del proyecto "detalle_dr_coss", o que se ubican en Ciudad del Carmen, o los dos.

Q5 con lenguaje SQL:

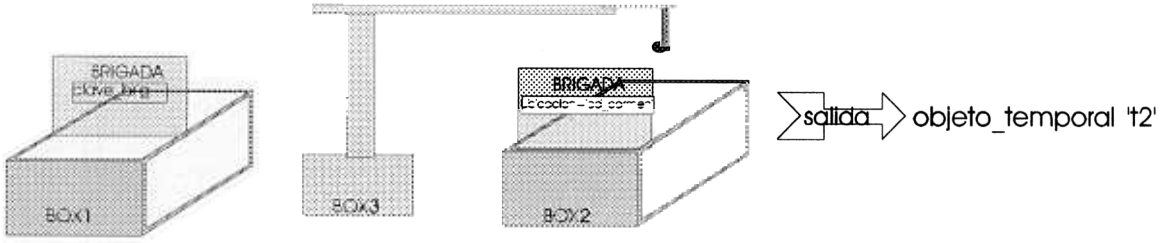
```
(select clave_brig
 from pozo
 where nom_prosp = "detalle_dr_coss")
union
(select clave_brig
 from brigada
 where ubicacion = "cd_carmen")
```

Q5 con lenguaje CQL:

VQS1:



VQS2



VQS3

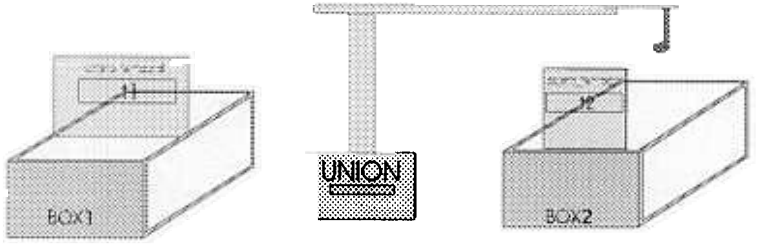


Figura 7-8 Q5: Encontrar a todas las brigadas que se crearon en un pozo en el proyecto "detalle_dr_coss", o que se ubican en Ciudad del Carmen, o los dos.

Esa consulta se dividió en tres pasos. El VQS1 corresponde a la subconsulta

select clave_brig
from pozo
where nom_prosp = "detalle_dr_coss")

y el resultado de la subconsulta se almacena en el objeto_temporal con nombre t1

El VQS2 corresponde la subconsulta

select clave_brig
from brigada
where ubicacion = "cd_carmen'

y el resultado de la subconsulta se almacena en el objeto_temporal con nombre t2.

El VQS3 corresponde a la operación UNION. En BOX1 y BOX2 se meten las tarjetas objeto_temporal con parámetros t1 y t2 respectivamente. En BOX3 se pega la tarjeta UNION. Este comando realiza la unión entre las entidades objeto_temporal t1 y t2.

7.2.6 comparación de la función de vista con objeto_temporal

En SQL, se usa el comando "create view" para crear una relación temporal. En CQL, el mecanismo objeto_temporal tiene la función equivalente. La forma de la orden en SQL create view es:

create view v as <expresión de consulta>

En CQL la forma es:

VQS $\xrightarrow{\text{salida}}$ **objeto_temporal "v".**

7.2.7 comparación de las operaciones espaciales

CQL, por su orientación, utiliza relaciones con clasificación, este atributo permite agregar una semántica muy poderosa a las relaciones "simples" existentes en SQL. Esta nos lleva a clasificar las " relaciones entre entidades" de la siguiente manera:

(1) **Operación de orientación**
En lenguaje CQL se oferente los comandos, UP_OF, BELOW_OF, RIGHT_OF, LEFT_OF, NORTH_OF, SOUTH_OF, EAST_OF y WEST_OF, para procesar las consultas de las operaciones de orientaciones. Por ejemplo, una consulta como Q6:

Q6 con **lenguaje natural**: Encontrar los puentes que se ubican arriba del río SAN JUAN.
Q6 con **lenguaje SQL:** no se puede expresar.
Q6 con **lenguaje CQL**:

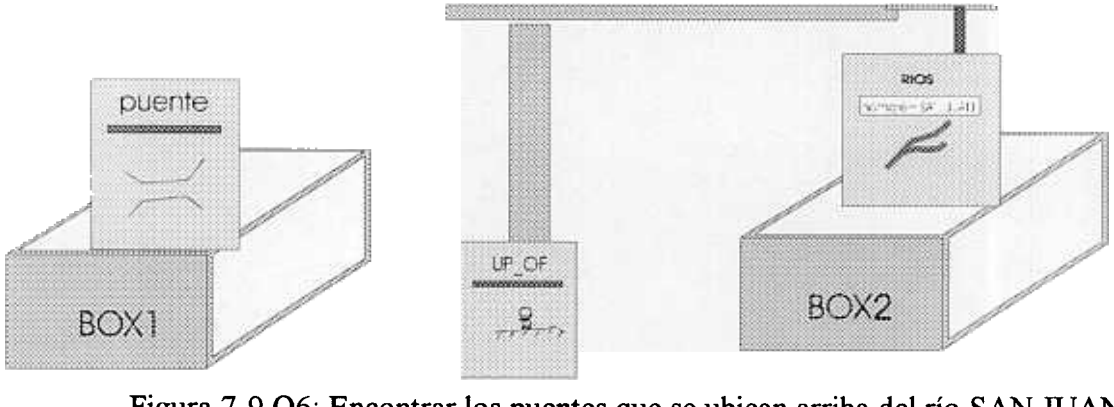


Figura 7-9 Q6: Encontrar los puentes que se ubican arriba del río SAN JUAN

(2) **operación metricas**
CQL ofrece los comandos para procesar las consultas de las operaciones metricas como NEAR_OF, FAR_OF.Una consulta de este tipo se muestra en Q7:

Q7 con **lenguaje natural**: Encontrar los pozos que se ubican cerca (2000m) del río SAN JUAN.
Q7 con **lenguaje SQL**: no se puede expresar.
Q7 con **lenguaje CQL**:

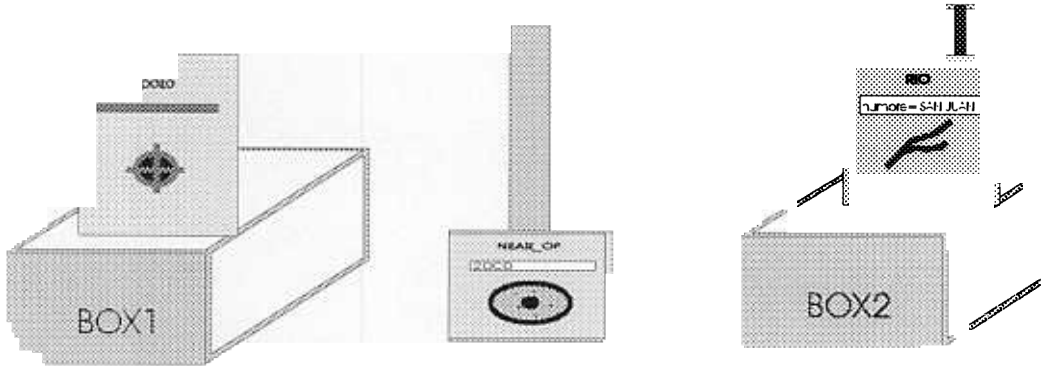


Figura 7-10 Q7: Encontrar los pozos que se ubican cerca (2000m) del río SAN JUAN

(3) **Operación topográfica**
En CQL se incluyen los comandos para procesar las consultas de las operaciones topográficas como INSIDE_OF, OUT_OF, ALONG_OF. Un ejemplo de consulta se muestra en Q8:

Q8 con **lenguaje natural:** Cuales pueblos y pozos están dentro del proyecto "detalle_dr_coss y cuales carreteras pasan el proyecto?
Q8 con **lenguaje SQL**; no se puede expresar.
Q8 con **lenguaje CQL**:

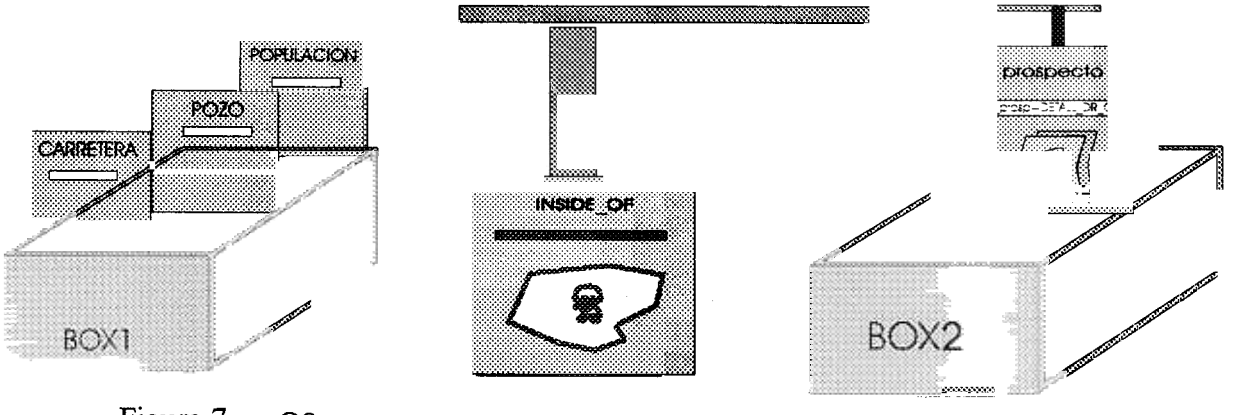


Figura 7- Q8: Cuales pueblos y pozos están dentro del proyecto "detalle_dr_coss y cuales carreteras pasan el proyecto?

(4) **Determinación de valores**

Debido a los avances de la ingeniería civil, el tamaño de las obras, y el volumen de trabajo, se hace necesario solucionar el mayor número de problemas en el menor tiempo posible. El CQL ofrece los comandos para, DISTANCE, LENGTH, AREA, VOLUME, para determinar los valores específicos de geometría y topografía. Un ejemplo de consulta se muestra en Q9:

Q9 con **lenguaje natural**: El pueblo "CANTU" de proyecto "detalle_dr_coss' qué tamaño tiene?

Q9 con **lenguaje SQL**: no se puede expresar.

Q9 con **lenguaje CQL**:

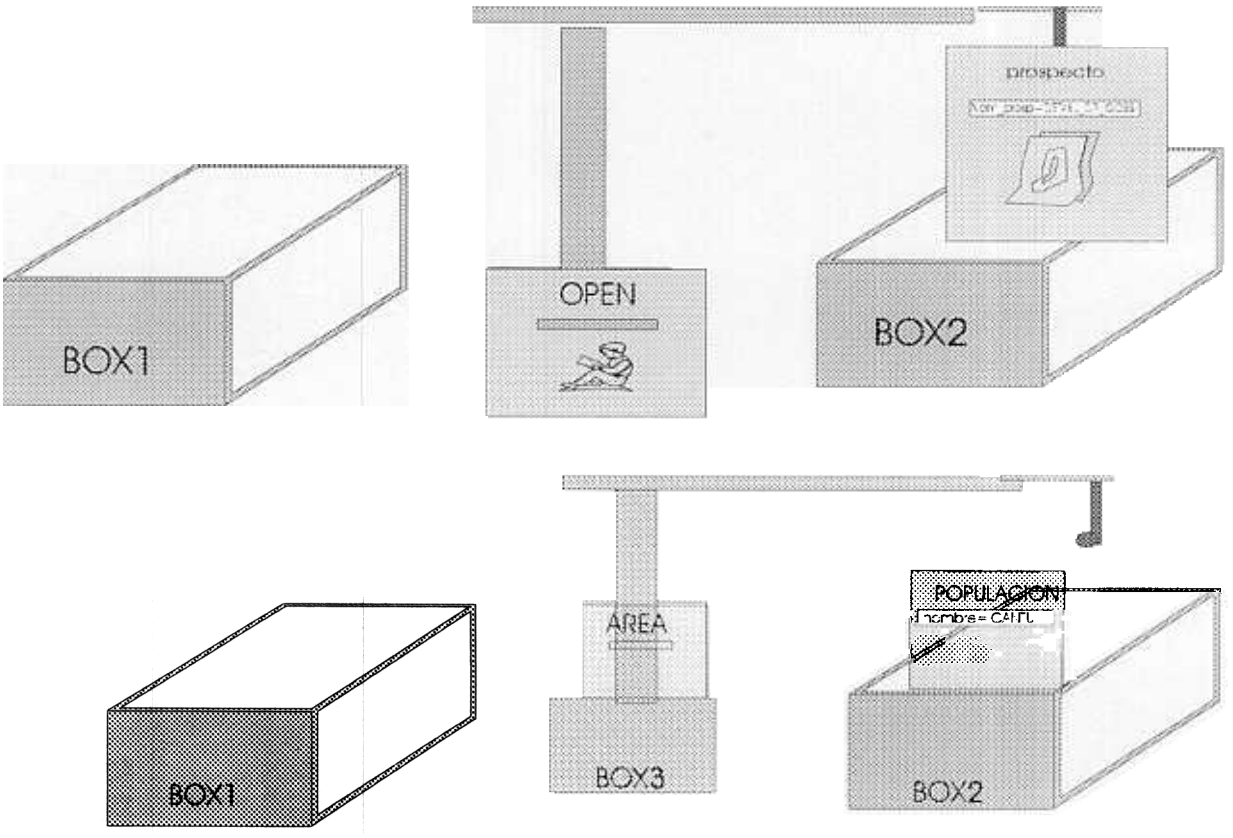


Figura 7-12 Q9: El pueblo "CANTU" de proyecto "detalle_dr_coss" qué tamaño tiene?

CONCLUSIONES

En este capítulo, se compararon las estructuras, las funciones y las formas de una consulta entre dos lenguajes: SQL y CQL. Se puede decir que CQL es una lenguaje de consulta de expansión del SQL. Esta expansión esta devidida en 3 aspectos:

- Salida de graficos
- Ambiente visual
- operaciones espaciales

Podemos a concluir la comparación entre dos lenguajes como se muestra en la tabla 7-8.

Tabla 7-8 conclusión de comparación las funciones entre lenguaje SQL y CQL

Clasificador	SQL	CQL
operación de tabla	hay	hay
operación de orientación	no hay	hay
operación topográfica	no hay	hay
operación metrica	no hay	hay
representación del resultado de la consulta	textual	textual y gráficos

CAPITULO 8

VISUALIZACIÓN DE LA INFORMACIÓN DE LA BASE DE DATOS

La base de datos muestra gráficamente en pantalla los resultados de la consulta, esto puede hacerlo en diversas formas según como lo deseé el usuario. En este capítulo se describen las formas de representación y los algoritmos para el despliegue de gráficos.

8.1 DIVERSAS REPRESENTACIONES GRÁFICAS

En la actualidad se han desarrollado algunas técnicas para transmitir información espacial. Un ejemplo de ello es la representación espacial en forma gráfica (mapas) usada en geografía. Pero esta técnica, también la usan otros especialistas, por ejemplo: un planificador urbano o un geólogo.

Anteriormente, se desarrollaron mapas con información de una manera muy general, independientemente de las necesidades de los usuarios ya que la información en la carta topográfica no se podía separar de acuerdo a sus necesidades, por lo tanto el usuario solo tenía una vista general de la información espacial. Cuando esta información era requerida para un proyecto específico, frecuentemente se encontraban con el problema de que era mucha la información que no era de su interés y que se incluía en el mapa.

La base de datos proporciona una buena solución a este inconveniente ya que los usuarios finales del lenguaje CQL pueden seleccionar cualquier subconjunto de información espacial que sea de su interés, en la forma que ellos quieran.

Un requerimiento necesario para el lenguaje CQL es la representación gráfica de los resultados de una consulta. En aplicaciones de bases de datos, los mismos datos se presentan en diferentes formas según las necesidades del usuario.

Con objeto de facilitar al usuario la consulta de los temas específicos, así como la localización rápida de aquellas cartas que por su escala -y que por consiguiente, el nivel de detalle que presentan- sean las que respondan a intereses específicos, en este apartado se hace una breve descripción de las características e información que reúne cada carta.

8.1.1 cartografía aplicada

El fin básico de la cartografía es dar información sobre los recursos con que cuenta el hombre para su mejor desarrollo. La información se refiere a servicios, infraestructura, topografía, recursos naturales, climas, equipamientos y otros aspectos.

La cartografía es relevante a cualquier nivel; desde el sectorial o regional hasta el de carácter nacional.

CARTA GEOLÓGICA.
Proporciona información referente a la naturaleza, las características de las rocas y el origen de los suelos.

Los afloramientos rocosos se clasifican en tres grupos principales: rocas ígneas, sedimentarias y metamórficas. Además, se incluyen datos sobre pozos acuíferos y sus grados de explotación ; manantiales y su temperatura; así como norias y minas. De estas últimas se indican los minerales que se extraen; los bancos de materiales y la utilización de éstos en la construcción. Se elabora a escala 1:50 000, 1:250 000 y 1:1 000 000.

CARTA HIDROLÓGICA.
a)De aguas superficiales.
b)De aguas subterráneas.
Ofrece información referente a la existencia de aguas superficiales y subterráneas, y el uso que se hace de ellas. Proporciona datos sobre la ubicación de estaciones climatológicas e hidrométricas, así como de manantiales, ríos, arroyos, lagunas, presas, bordos, aljibes, pozos y canales.

También indica el resultado del análisis químico de las aguas, las condiciones de permeabilidad de cada unidad, las áreas donde existen acuíferos con agua dulce, salobre o salada, además, el flujo de agua subterránea. Se elabora a escala 1:250 000 y 1:1 000 000

CARTA TOPOGRÁFICA.
Hace una representación del relieve, la hidrografía, la vegetación (áreas de cultivo y/o bosques) y las obras construidas por el hombre, con alta precisión métrica realizada mediante técnicas fotográmetricas y cartográficas, a partir, principalmente, de fotografías aéreas y de información geodésica.

Se elabora a diversas escalas, a fin de cubrir diferentes necesidades, razón por la cual la cantidad y detalle de información varía de una escala a otra. Se elabora a escalas 1:50 000, 1:250 000 y 1:1 000 000.

MAPA URBANO.
Se realiza para ciudades con más de 40 mil habitantes. Señala manzanas, nombre y trazo de las calles, espacios construidos; ubicación de escuelas, hospitales, templos, bancos, monumentos, teatros, cines, zonas industriales y demás puntos de interés general.

Constituye una base útil para la planificación del crecimiento racional de las ciudades, ya que facilitan el diseño de un plano regulador que considere las posibilidades de desarrollo demográfico. Se elabora a escalas 1:5 000, 1:7 500, 1:10 000, 1:12 500, 1:15 000 y 1:25 000. El mapa urbano realizado por el INEGI y la Comisión de Conturbación del Centro del País se elabora a escalas 1:10 000 y 1:20 000.

El uso de la computadora ha sido una herramienta eficiente para guardar, consultar y manipular gran cantidad de información espacial.

La base de datos espacial aplicada a la exploración petrolera, efectúa operaciones para el área geofisica y para el área de interpretación.

Para la operación geofisica el sistema VISDA permite graficar entre otros, los siguientes elementos:

plano de líneas sismológicas por prospecto
plano de pozos
plano de avance de observación
plano de avance topográfico

Para la interpretación permite graficar:

perfiles topográficos de líneas sismológicas
plano de geología superficial y líneas sismológicas
plano de configuración del prospecto

En la siguiente sección explicamos cuales son los elementos importantes en una carta y la manera de dibujar las cartas, o mapas.

8.1.2 elementos del mapa

La base de datos muestra gráficamente en pantalla los resultados arrojados después de una consulta, esto puede hacerse en diversos formatos según el área de aplicación. Una forma general es el mapa.

Un mapa consiste de varias clases de símbolos. Cada clase puede formar un mapa especial. Por ejemplo, mapa de pozos o de carreteras. Desde el punto de vista de la base de datos, los archivos corresponden a un mapa que almacena una serie de registros de entidades.

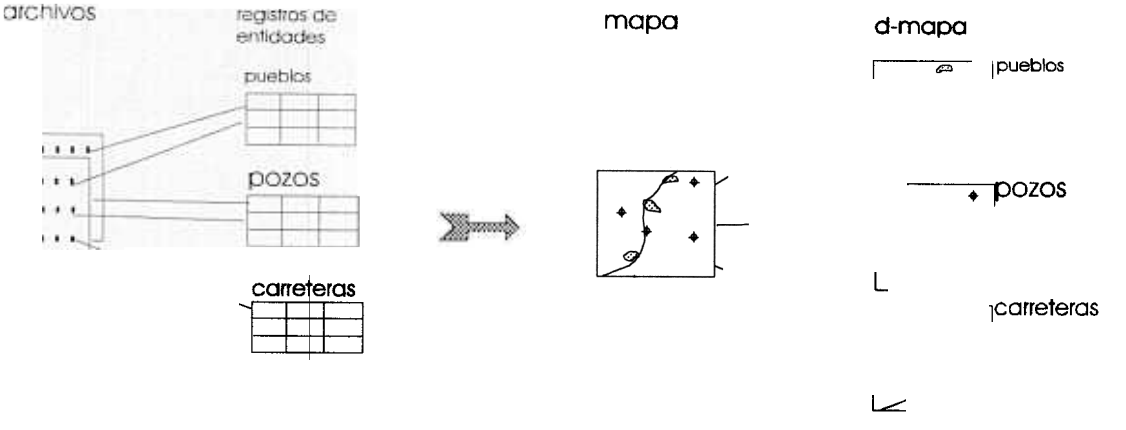


Figura 8- Los archivos contienen los registros de entidades

que corresponden a símbolos en los mapas.

Lo que se despliega en una mapa es un conjunto de los símbolos. Desde el punto de vista matemático, es un mapeo. La operación de mapeo es de los registros a los símbolos. Y tiene una cardinalidad de muchos a uno.

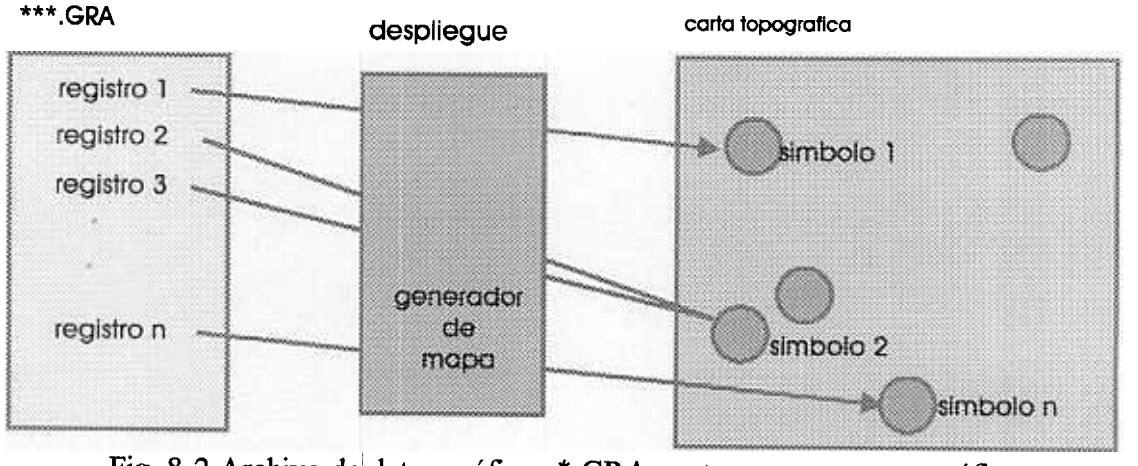


Fig. 8-2 Archivo de datos gráficos *.GRA contra una carta topográfica.

Escalares.

Desde la clasificación precedente de mapas se ha notado que las escalas se usan en un rango que va desde 0.1 pulgada igual a 1 pie como el más grande, y de 1 pulgada igual a varios cientos de millas como el más pequeño. La escala de mapa se muestra tanto gráfica como numéricamente en alguna parte cerca del título o como parte del mismo.

Las medidas de encuesta se realizan en pies y fracciones de pies, las cuales se expresan en pies y decimales de un pie, en vez de pies y pulgadas; de aquí en adelante las escalas de los ingenieros sobre la caja de reglas están en el sistema decimal - y tienen 10,20,30, divisiones de pulgada, por ejemplo. Estas unidades pueden representar también 1,2, o 3 pies a pulgada, 10, 20, 30 pies a pulgada, o 100, 200 o 300 pies a pulgada. Todas estas escalas ocurren frecuentemente en el trabajo de un ingeniero.

La selección de una escala que se haga para un mapa estará influenciada por muchos factores, los principales se cuentan entre el tamaño y el carácter del área ha mostrarse, la forma en que el mapa está presentado, y el propósito para el cual será usado. El costo de preparación y el tamaño de servicio debe ser considerado.

En el sistema usamos las subrutinas siguientes para hacer mapeo de coordenadas del mundo real a las coordenadas del mapa.

```
void escalarx(float *corx, float w_min_x)
    {
     *corx = v_min_x + fx * (*corx - w_min_x);
```

```
void escalary(float *cory, float w_min_y)
      {
       *cory = v_max_y - (v_min_y + fy *(*cory - w_min_y));
       }

factores(float w_min_x, float w_min_y, float w_max_x, float w_max_y)
     {
      fx = (v_max_x - v_min_x)/(w_max_x - w_min_x);
      fy = (v_max_y - v_min_y)/(w_max_y - w_min_y);
     }
```

* v_min_x,v_min_y, v_max_x, v_max_y son los coordenadas máximas y mínimas de mapa.
* w_min_x, w_min_y, w_max_x, w_max_y son los coordenadas máximas y mínimas del área que se está tratando.
* corx,cory son apuntadores de los registros de coordenadas.

8.2 SÍMBOLOS

Cuando la escala de cualquier mapa es pequeña, relativamente hablando, la representación de objetos debe ser altamente convencional. Sobre todo para una escala más grande -de diseño de mapas: aún los objetos más grandes deben ser mostrados por símbolos más bien que por vistas de planos de ellos. El propósito de un mapa es, después de todos, no mostrar la apariencia exacta a la vista, más bien, mostrar el tamaño comparativo de los objetos y su posición relativa de uno a otro. A partir de aquí los símbolos convencionales se han ideado guardando alguna semejanza posible, a la personalidad del objeto. El propósito de tener esta semejanza es por conveniencia de la interpretación.

Los mapas consisten de símbolos. Entonces, se puede decir que el despliegue de un mapa es el despliegue de los símbolos.

Un sistema de los símbolos bien estandarizado, usado prácticamente por todo el departamento diseñador de mapas del gobierno, se publica en un folleto pequeño llamado "los Símbolos Militares".

En la figura 8-3 se muestran los símbolos estándar utilizados en un mapa.

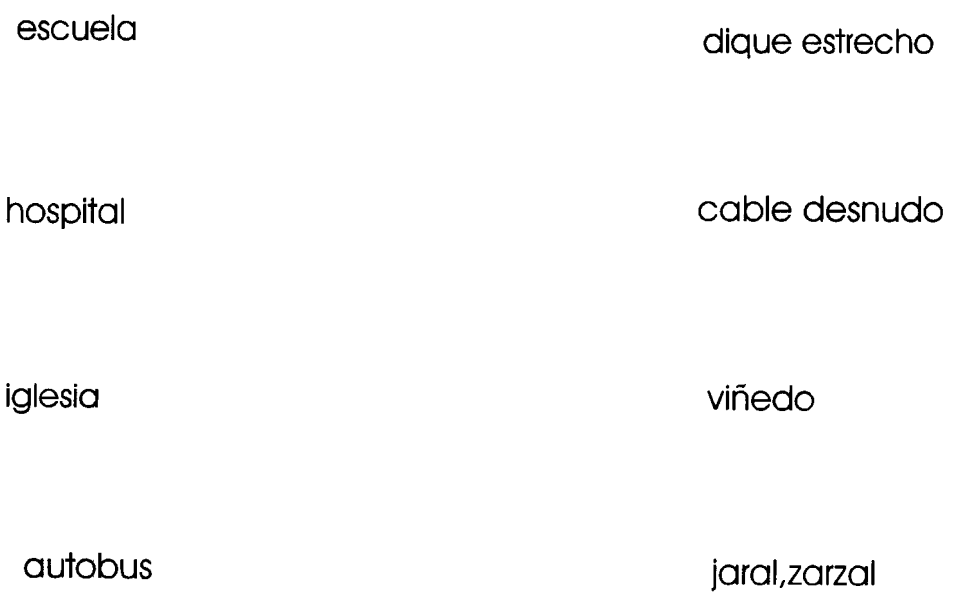


Figura 8-3 ejemplos de símbolos estándar.

En el sistema se necesita un conjunto de subrutinas para generar estos símbolos. Este conjunto es llamado Biblioteca de Símbolos.

Lógicamente, la subrutina hace un mapeo del registro del archivo al signo. Las subrutinas de la biblioteca son generales, es decir, cada subrutina puede dibujar una serie de símbolos. Por ejemplo, la subrutina para dibujar el símbolo de POZO puede aceptar los parámetros: coordenadas (x,y) del punto central del símbolo, escala del mapa y tipo de pozo. Según los parámetros será la posición de los símbolos en el mapa. En seguida se muestra un ejemplo simple de las subrutinas. Esta subrutina puede desplegar un serie de símbolos de pozos.

```
void p_pozo(float px,float py,char *nom_pozo,float xm,float ym, int tipo, float rp)

   escalarx(&px,xm)
   escalary(&py,ym);
   switch(tipo){

      case 1      '* dibujar un simbolo de tipo

                  XDrawArc(XtDisplay(draw),XtWindow(draw),gc6,(short)(px-rp),(short)(py-rp),
                        2*rp,2*rp,0,23040);    /*dibujar un cilculo*/
                  XDrawLine(XtDisplay(draw),XtWindow(draw),gc1,(int)px-2*rp,(int)py,
                        (int)px+2*rp,(int)py);    /*dibujar un segmento*/
                  XDrawLine(XtDisplay(draw),XtWindow(draw),gc1,(int)px,(int)py-2*rp,
                        (int)px,(int)py+2*rp);    /*dibujar un segmento*/
                  XDrawString(XtDisplay(draw),XtWindow(draw),gc6,(int)px,(int)py,
                        nom_pozo,strlen(nom_pozo)); /*escriber el nombre del pozo*/
```

```
                break;
        case 2:
```

8.2.1 Tamaño y protuberancia de símbolos

El tamaño de símbolos debería variar únicamente de manera ligera en relación con la escala del mapa, desde, casi cualquier escala, la mayoría de los símbolos son exageraciones, por más pequeños que estos se hagan. Los símbolos mostrados de la Figura 8-3 son del tamaño apropiado para los mapas comúnmente usuales en ingeniería.

Puesto que la variación en el tamaño de símbolos está bastante limitada, la protuberancia puede asegurarse mediante una variación en el ancho de las líneas usadas. El propósito del mapa podría determinar que símbolos serán elaborados mas prominentemente. Sobre un mapa de Propiedad de petróleo, por ejemplo, pozos fluyendo, pozos secos, ferrocarriles, caminos, y las líneas de propiedad son los aspectos más importantes.

8.2.2 Colores de símbolos

Sobre un mapa terminado los símbolos deberían mostrarse en diferentes colores. El color para cada símbolo se indica en el título de la figura. Estos colores pueden ser fácilmente referenciados en cuatro grupos simples: aspectos artificiales, o trabajos de hombre, se hacen en color negro; el agua se caracteriza en azul; contornos, arena, lava, entonces en marrón, y vegetación en verde. En el momento de imprimir los mapas, cada color requiere de una impresión separada: por lo tanto una reducción en el número de colores empleados disminuye el costo. Puesto que la vegetación, a excepción de bosques muy grandes, no es permanente, el verde comúnmente es omitido.

8.2.3 Esparcimiento de símbolos

Una de las más mayores dificultades al dibujar símbolos está en aprender como espaciarlos, de manera que no tengan las ubicaciones especificadas al trazar el plano. La tendencia general está en hacer la hoja demasiado gruesa. El dibujante debe constantemente estar sobre su resguardo contra esta práctica por dos razones. Primero, los mayoría de los símbolos se dibujan tomando el largo de este y el costo mayor para producir el mapa. Segundo, éste es más difícil para producirlo con uniformidad de textura cuando los símbolos son apiñados. Las áreas livianas y pesadas sobre el mapa son desagradablemente perceptibles cuando los símbolos se ponen demasiado juntos. Cuando existan áreas grandes que requieran ser cubiertas con símbolos que involucren el uso de líneas paralelas, como en el caso de pantanos,debe emplearse una sección de vapor.

8.2.4 Posición de símbolos

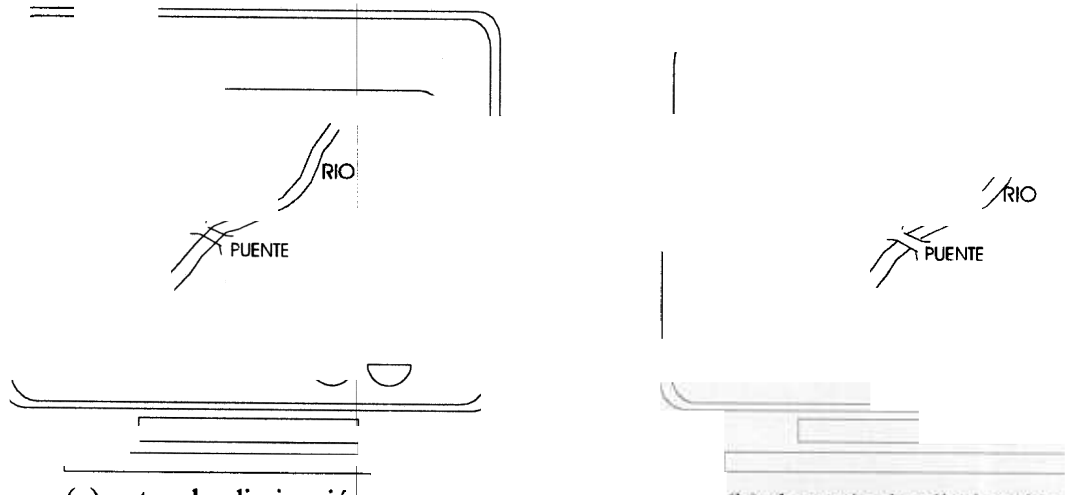
Otro punto muy importante es la posición de los símbolos sobre la hoja. Todos los símbolos que tienen una base definitiva, por ejemplo, pasto, marisma, árboles de palma y maíz, deberían dibujarse con la base paralela al fondo de la hoja, para que los símbolos aparezcan en una posición natural derecha. Ellos nunca deberían dar lugar en sus bases paralela a caminos o líneas de propiedad que corren diagonalmente a través de la hoja. Símbolos para la vegetación que ocurre en filas, sin embargo, puede tener filas que corren en cualquier dirección.

Los mapas de propiedad de diversas industrias pueden requerir también que el ingeniero defina símbolos para mostrar ciertos aspectos que no son incluidos entre los símbolos estándares. Debería ser propósito del ingeniero siempre hacer tales símbolos inconfundiblemente con respecto al significado y una fácil interpretación.

8.3 ALGORITMO PARA ELIMINACIÓN DE SÍMBOLOS OCULTOS DE 2 1/2 DIMENSIONES

8.3.1 problema

Cuando se despliegan los símbolos en un mapa, el mapeo es de muchos a uno como se describe en la sección 1. Por ejemplo, tenemos que en la posición (x,y) de un mapa, hay dos entidades, río y puente, que deben desplegarse en el mismo lugar (x,y) según el mapeo. Pero, si se despliegan dos símbolos en el mismo lugar, en el carta, se generará un caos como se muestra en figura 8-4 (a).



(a) antes de eliminación de símbolos ocultos

(b) después de eliminación de símbolos ocultos

Figura 8-4

Por lo anterior, necesitamos un algoritmo para eliminación de símbolos ocultos.

8.3.2 modelado del algoritmo

Puesto que la ventana de despliegue se divide en un red de M×N. Se usa un arreglo de identificación ID[0..m, 0..n] correspondiente a el. Al iniciar, el área se encuentra llena de ceros como se ve en la figura 8-5 (a).

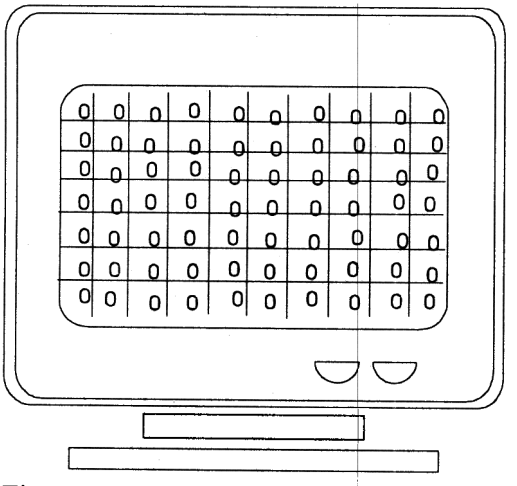
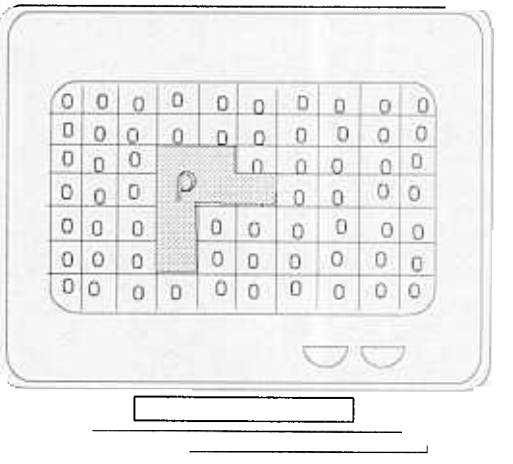


Figura 8-5 (a) iniciación de ventana



(b) área ρ ya está cubierto por un símbolo

Los valores de m,n dependen de la precisión del algoritmo. Si ya se despliegó un símbolo que cubre cierta área ρ, los cuadros cambian su identificación a 1. Después, si se requiere que se desplieque otro símbolo en el área ρ, esto se niega. En otra palabra, el símbolo es un símbolo oculto. Pero, posiblemente, no todo la parte del símbolo este dentro del área ρ. La parte que está dentro de área ρ es la parte invisible y otra es visible. La parte visible se debe desplegar. Entonces el problema consiste en decidir las fronteras entre las parte visible e invisible del símbolo.

Ya sabemos que el despliegue de un símbolo es precisamente el despliegue de las líneas que están definidas por dos puntos: punto inicial y punto terminal. Entonces, se puede decir que el despliegue de un símbolo es el despliegue de una serie de pares de puntos $\{P_i P_j\}$ es decir, un conjunto de puntos $\{P_1, .. P_j, ..., P_n\}$. Por ejemplo, el símbolo de puente, como se ve en la figura 8-6, puede definirse en 8 líneas.

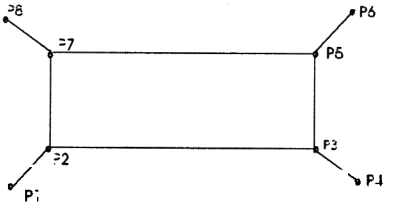


Figura 8-6 un símbolo de puente consiste de 8 líneas

P_8P_7, P_7P_5 , P_5P_6 , P_5P_3 , P_3P_4 , P_2P_3 , P_1P_2 o como un conjunto de puntos { P_1 , P_2 , P_3, P_4 , P_5 , P_6 , P_7 , P_8 }. Si la parte del sub-conjunto { P_3, P_4 , P_5 , P_6 } es una parte invisible, entonces será como se muestra en la figura 8-7.

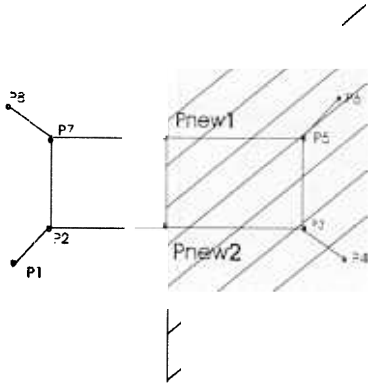


Figura 8-7 La parte sombreado es invisible

Por lo tanto es necesario determinar la frontera entre las dos partes. En este ejemplo, es necesario interpolar los puntos nuevos P_{new1} P_{new2} sobre líneas P_7P_5 y P_2P_3 puesto que los pares ordenados de salida son { $P_i(x_i y_i)$}, i = 1, 2, ..., n, donde el punto actual es i con coordenadas (x_i, y_i), entonces el punto anterior será i-1, el cual ya se procesó. Después se tiene que ver si la línea P_{i-1} P_i es una línea oculta.

En la figura 8-8 se muestran algunos de las diferentes relaciones entre el segmento P_{i-1} P_i y las regiones ocupadas. En esta figura la parte sombreada es la parte ocupada por otro símbolo.

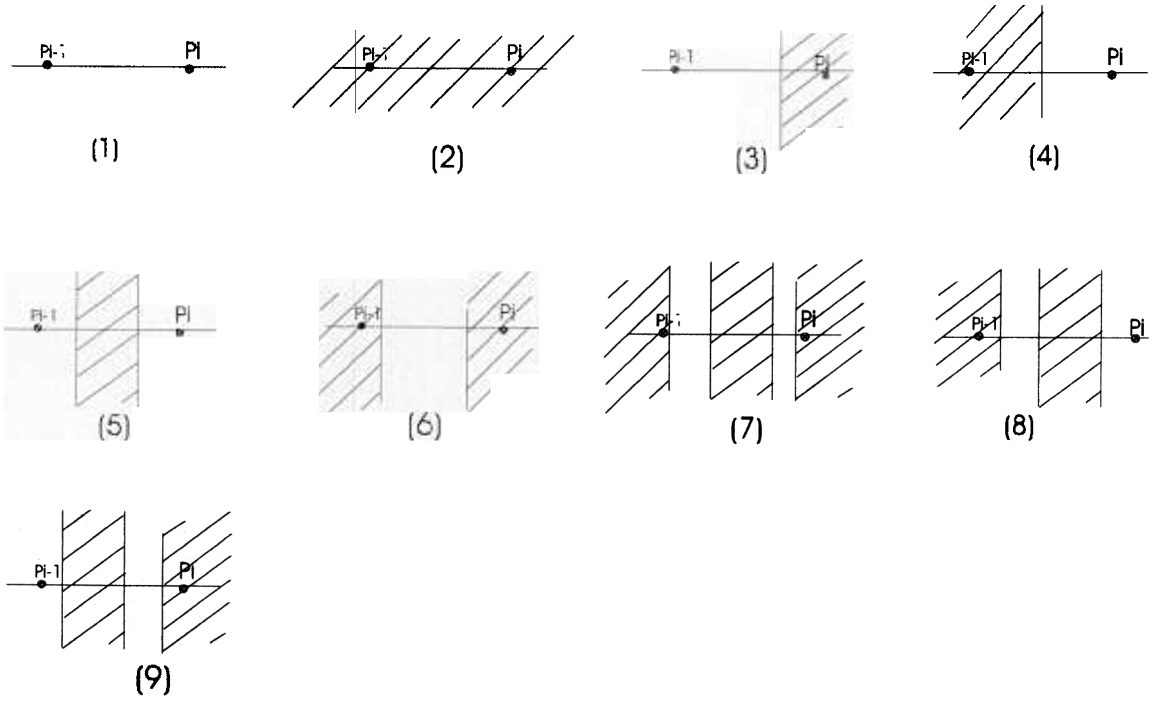


Figura 8-8 Casos de relaciones entre el segmento P_{i-1} P_i y las regiones ocupadas.

En el algoritmo se controla que la distancia entre el punto P_{i-1} P_i y el punto P_i sea una constante ε. Entonces no es necesario considerar todos los casos de la figura 8-8 , con los primeros cuatro casos es suficiente.

Algoritmo que elimina un segmento oculto

Paso Definición de ε=3

Paso 2: Si $| P_{i-1} - P_i | > \varepsilon$, toman el punto medio proporcional Pi'(x_i', y_i'), donde

$$x_i' = \frac{x_{i-1} + x_i}{2}$$

$$y_{i'} = \frac{y_{i-1} + y_i}{2}$$

Luego se regresa al paso 2 y se procesan los segmentos P_{i-1} P_i' y P_i'P_i

Paso 3: Si el punto P_{i-1} y punto P_i son puntos invisibles. El segmento es invisible. Si el punto P_{i-1} y punto P_i son puntos visibles, el segmento es visible. Saltar al paso 6.

paso 5: Si punto P_{i-1} y punto P_i no tienen la misma propiedad, en otras palabras, uno es visible y el otro no, el segmento P_{i-1} P_i se trata como un segmento invisible.

Paso 6: Procesamiento punto siguiente P_{i+1}
Paso 7: Fin del algoritmo.

Este algoritmo se basa en una función que decide si un punto es o no visible. Teóricamente es algo muy fácil, ya sabemos que la ventana de salida se divide en forma de red de m x n y se usa un arreglo ID[1..m, 1..n] para guardar los estados de cada cuadro, por ejemplo, el cuadro (i,j), se encuentra ocupado por un símbolo, entonces, su estado ID(i,j)=1.

Si existe otro símbolo que quiere desplegarse en un cuadro (ver figura 8-9), primero examina a ID(i,j), si es 1, esta parte del símbolo solo puede desplegarse como la parte oculta, en caso de que no sea un punto oculto ID(i,j) será 0 y se usa el algoritmo siguiente.

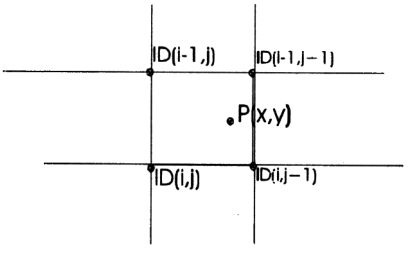


Figura 8-9 Un punto P(x,y) cayó dentro del cuadro (i,j).

Algoritmo de punto oculto:

Entrada: Coordenada de un punto P(x,y).

Paso 1 Mapeo de coordenadas del mundo real (x,y) a coordenada de ventana (i,j).

Paso 2: Si ID(i,j)=1, el punto P(x,y) es un punto oculto, en caso contrario el punto P(x,y) es un punto visible.

Paso 3: Fin del algoritmo.

8.4 ESTRUCTURA DE SALIDA DE LOS SÍMBOLOS

El algoritmo de eliminación se basa en el hecho de que los símbolos ya están ordenados. En esta sección describimos como ordenar los símbolos, primero explicamos el diagrama del procesamiento de estos. Ver figura 8-10.

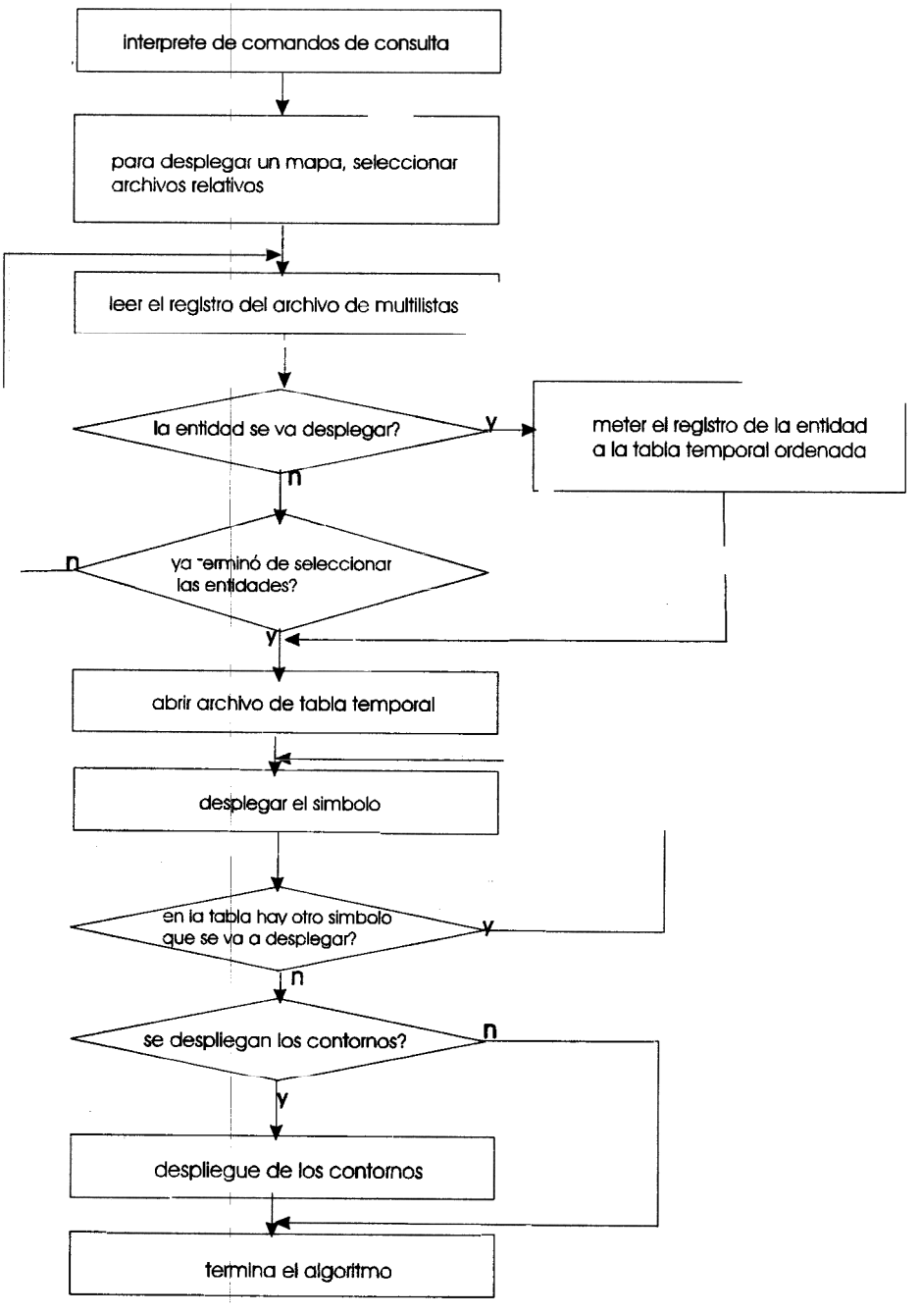


Figura 8-10 . Diagrama de despliegue de símbolos.

En el algoritmo, la selección de entidades depende del interprete de comandos de consulta, el interprete decide cuales archivos se van a abrir y selecciona los registros del archivo, si es

un registro el que se va desplegar en la pantalla, lo guarda en una tabla temporal ordenada como se muestra en la figura 8-11 .

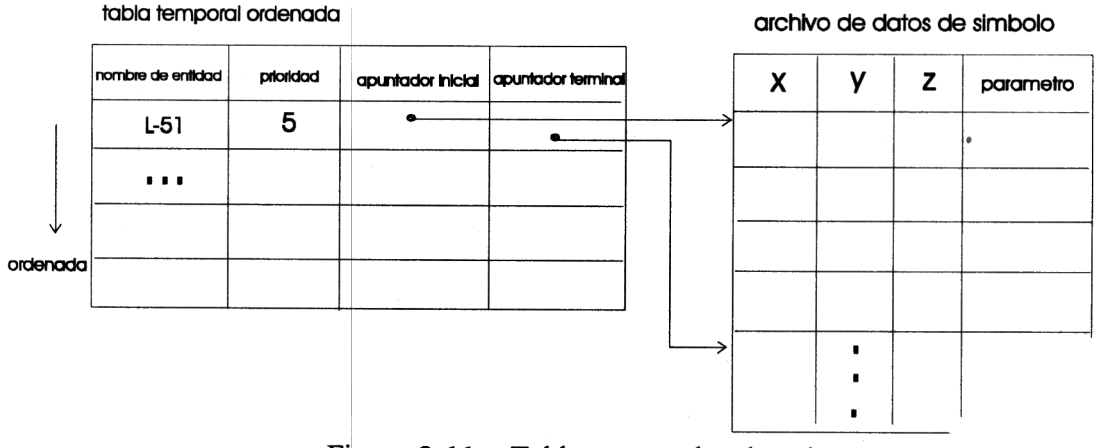


Figura 8-11 . Tabla temporal ordenada.

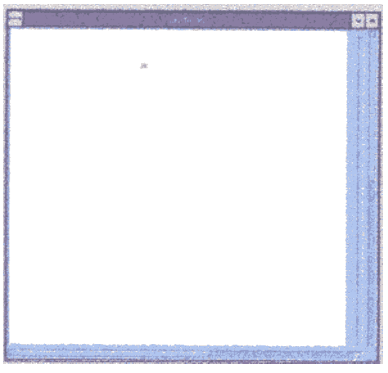
Cuando se desea meter un símbolo nuevo a la tabla, primero se busca su lugar según la regla de ordenación de símbolos analizada en la sección 2; generalmente la prioridad del símbolo punteado es más alta que la del símbolo lineal y la del símbolo lineal es más alta que la del símbolo de superficie. Por ejemplo, si un puente es punteado,y el río es una línea; entonces la prioridad del puente es más alta que la del río.

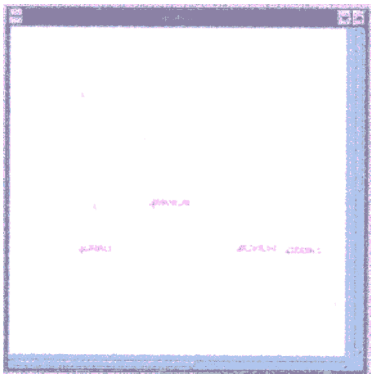
Si dos símbolos son de la misma clase, sus prioridades son iguales, y se usa el algoritmo FIFO. Por ejemplo, en carretera y vías férreas se usa este algoritmo. Entonces se despliegan los símbolos en tres segmentos:

El primer segmento de salida contiene las entidades punteadas, el segundo las lineales y el tercero las de superficie.

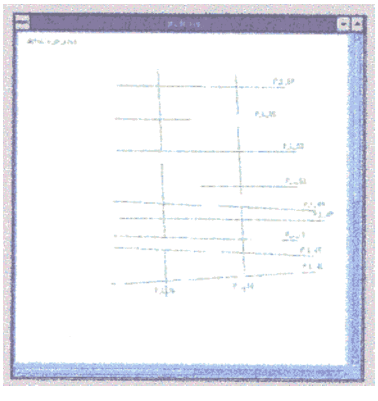
Para la representación de las características de superficie en una área de tierra los ingenieros siempre usan la forma de mapas de contornos. En el sistema, los contornos se tratan como un símbolo especial. En la tabla de salida los símbolos tienen menor prioridad, en otras palabras, primero se despliegan los símbolos punteados, luego los símbolos lineales, los de superficie y al final se despliegan los contornos. Los algoritmos de despliegue de los contornos se vea en [JU 92].

En las figuras de 8-12,8-13 y 8-14 se muestran algunos ejemplos de las presentaciones de resultatos de consulta a la base de datos. Los elementos de mapa se pueden combinar según los requerimientos de usuario.



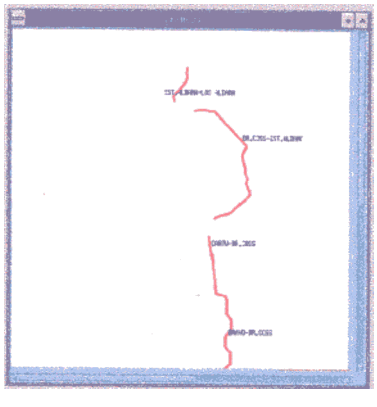


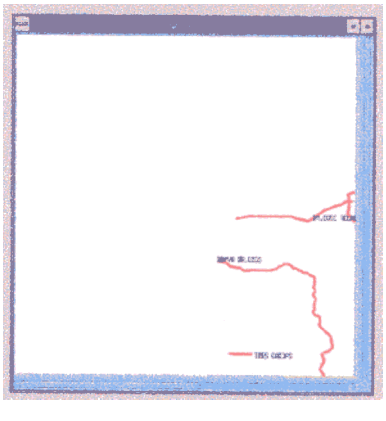
(b)

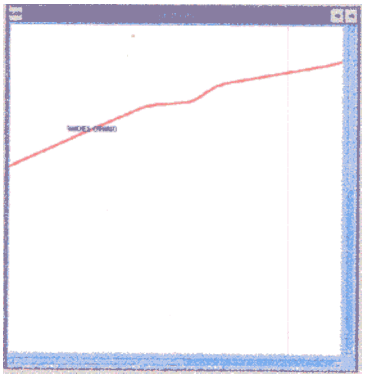


d)

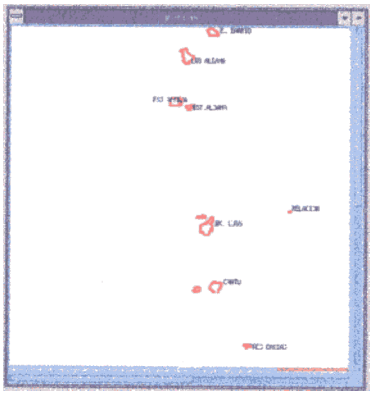
gu de ıpaı de brigada (b) mapa de los (c) mapa de los
ıpa de rio ıN







(c)

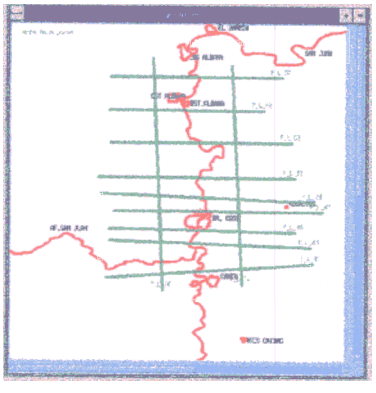


(d)

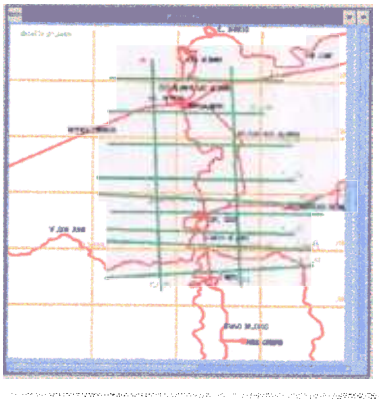
Figura 8-13 (a) mapa de carretera (b) mapa de caminos (c) mapa de ferrocarril (d) mapa de los pueblos

(a)

(b)



(c)



(d)

Figura 8-14 (a) mapa de pueblos y carreteras (b) mapa de pueblos y rios (c) mapa de pueblos, rios y lineas sismología (d) mapa de proyecto DETALLE_DR_COSS

Este capítulo detalla como se visualiza la información de la base de datos espacial. Como la base de datos muestra gráficamente en pantalla los resultados arrojados después de una consulta, el sistema los puede presentar en diversas formas según la aplicación que le de el usuari. Por lo tanto, el usuario puede manipular los resultados de consulta y convertirlos en varias formas según sus necesidades. En esta parte se describen las formas de representaciones gráficas y los algoritmos para el despliegue de gráficos.

CONCLUSIONES

El desarrolló del sistema, **Visualización de una Base de Datos Espacial** , se llevó a cabo en combinación de dos vertientes principales: La implementación de un sistema de base de datos espacial aplicada a la exploración petrolera, y una nueva propuesta como lenguajes y ambientes visuales para la base de datos implementada. Los características del sistema son:

* Para el manejo de información espacial y nominal, se desarrolló un motor de la base de datos que incluye un modelo lógico relacional de datos y un modelo físico de datos basado en árboles B y multilistas.

* Se emplea un ambiente visual para la interfaz de usuario que usa primordialmente las entidades espaciales como objeto de visualización. Los usuarios finales, sin saber programación, pueden manejar fácilmente el sistema.

* Se da al usuario una herramienta que le permite "desplegar" la estructura de un atributo, un archivo, y del sistema.

* El usuario puede manejar los resultados de consulta y convertirlos en varias formas según sus necesitades. Las formas de representación pueden ser gráfica , texto, objeto temporal. La forma de grafico tambien puede ser mono-entidad o multi-entidades. Por ejemplo, plano de pozos, plano de líneas sismológicos por proyecto y plano de configuración del prospecto.

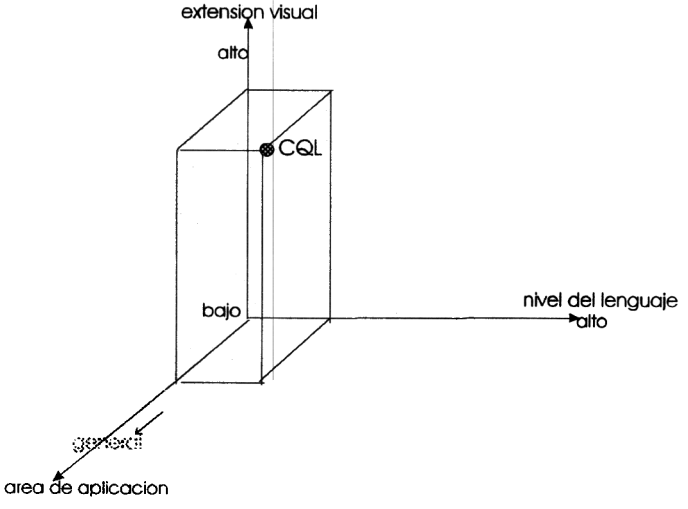
* Se definió un lenguaje de consulta visual CQL. El lenguaje se caracteriza en que las operaciones se especifican de forma visual y manejan información espacial. La especificación de operaciones se lleva a cabo mediante elementos visuales (tarjetas) que manejan información geográfica relativa a la cartografía de las regiones vinculada con información de pozos, rios, entre otras. Las características son:

emplear un ambiente gráfico.

- representación de las entidades y de las acciones sobre la base de datos usando tarjetas.
- ampliación de la construcción gráfica del lenguaje (objeto temporal), a fin de facilitar la expresión de una clase llamada sub-consulta, como almacenamiento temporal de la información.

La consulta realiza en forman de la relación tabla y la relación espacial.

Según el esqueleto mencionado en la figura 5-1, las características del lenguaje son: alta extensión visual, área mediana de aplicación y un nivel mediano de lenguaje, como se muestra en la figura siguiente.



BIBLIOGRAFIA

[ACZI 75] Aczil and Daroczy, On measures of information and their characterizations, New York, Academic Press, 1975.

[ADOR 93] Adoracion de miguel/Mario Piattin, Concepción y diseño de base de datos del modelo E/R al modelo relacional, Addison-wesley Iberoamericana, 1993.

[ALDA 96] Pedro Alday, El sistema del EVE, tesis de maestria. 1996 (por publicar).

[ALEJ 94] Alejandro Hinojosa-Corona, etc. Ubicación del riesgo geológico en la ciudad de Tijuana, Memoria de congreso IBEROAMERICANO de sistemas de información de base geografica, ciudad de méxico, marzo, 1994. pp150-158.

[AHO 86] A.V.Aho, R. Sethi and J.D. Ullman(1986) compilers: priciples, techniques and tools, Addison Wesley.

[BALL 82] Ballard D. H., Brown C.M., Computer vision, Prentice Hall Inc., 1982.

[BATI 92] Batini Ceri Navathe, Conceptual Database Design, The Benjamin/cummings publishing company, 1992.

[BAYE 72] Bayer and E.M.McCreight, Organization and maintenance of large ordered indices. Acta Informatica,1:3 pp173-189

[BORD 91] Bordogna G., I.Gagliardi and D.merelli, Iconic queries on pictorial data. Proceedings 1991 IEEE Workshop on visual languages. pp38-42.

[BOUR 84] Boursier P., Image data bases: a status report, Computer architecture for pattern analysis and image data base management, Miami Beach, Florida, Nov. 1985.

[BROL 89] Brolio J., Draper B.A., Beveridge J.R., Hanson A.R., ISR: A Database for symbolic processing in computer vision", Computer, Vol.22, No. 12, Dec. 1989.

[CALC 94] Calcinelli and D., M. Mainguenaud, Cigales, a visual query language for a geographical information system:the user interface, journal of visual languages and computing. No.5 1994, pp113-132.

[CATA91] T, Catarci, A.Massary and G. Santucci, Iconic and Diagrammatic Interfaces: An integrated Approach, Procedings 1991 IEEE Workshop on visual language, pp199-204

[CINQ 89] Cinque L., F.Ferloni, S.Levialdi and A. Sargeni, X-VIQU: An expert system for visual representation of database queries. Proceedings 1989 IEEE Workshop on visual language. pp183-188.

[CHAN 80] Chang N.S., K.S. Fu, Query by pictorial example, IEEE Trans. Sofw. Engng., SE-6, 1980.

[CHAN 81] Chang S.K., Kunii T.L., Pictorial data-base systems, IEEE Computer, Nov. 1981.

[CHAN 86] Chang S.K., Image database systems, in Handbook of pattern recognition and image processing, Academic Press, Inc., 1986.

[CHAN 87] Chang S.K., et. al., Iconic indecing by 2-D strings, IEEE Trans. Pattern Recognition and Machine Intelligence 9,3,1987.

[CHAN 87a] Chang S.K, Visual languages: A tutorial survey, IEEE Software, 1987.

[CHAN 88] Chang S.K. etc. An intelligent image database system, IEEE Transactions on software engineering, Vol.14, No. 5, 1988,pp681-688.

[CHAN89] Shi-Kuo Chang, Priciples of pictorial information systems design. Prentice Hall,1989.

[CHAN90] Shi-Kuo Chang, A visual language compilar for information retrieval by visual reasoning, IEEE Transactions on software engineering, Vol.16,No.10,1990,pp1136-1149.

[**CHAP91**] Chapa V.Sergio, Programación Automática a partir de descriptores de flujo de información. Tesis de doctoral. 1991, CINVESTAV del IPN.
[**CHAP 94**] Chapa V. Sergio, Fernando Fiurentino Pérez Diseno Conceptual y Lógico de una Base de Datos Geográfica Aplicada a la Exploración Petrolera.
[**CHAP 94a**] Chapa V.Sergio, Ju Shiguang y Pedro Aldey, Desarrollo de un sistema de información de base geográfica. Soluciones Avanzadas, Año 2, No. 15, Noviembre 1994, pp34-40.
[**CHAP 93**] Chapa Vergara Sergio, Perspectivas en la Automatización de las Metodologías de Integración de Software.
[**CHEN 76**] Chen P.P. The entity-relationship model:toward a unified view of data, ACM Transactions on Database systems, Vol.1 No.1 pp9-36.
[**CHOC 81**] Chock M., A. Cardenas and A.Klinger, Manipulating data structure in pictorial information systems. IEEE Comput. Vol.14, No.11, Nov. 1981.
[**CODD** 73] Codd E.F. further normalization of the database relational model in database systems. Prentice-Hall New York, 1972, pp33-64.
[**CODD** 74] Codd E.F. Recent investigation in relational database systems. Proc. IFIP 74, North-holland, 1974, pp1017-1021.
[**DANT 90**] Dante Alcántara García, Topografía, 1990, McGrawHill.
[**ERLA** 90] Erland Jungert, Towards a visual query language for and object oriented geographical information system, Proceedings 1990 IEEE Workshop on visual languages. pp132-137.
[**GABR** 91] Gabriele Rohr, Graphical user languages for querying information: Where to look for criteria? Proceedings 1991 IEEE Workshop on visual languages. pp21-28.
[**GERA94**] Gerald L.Lolise, Kevin Biolsi, etc. "A Classification of Visual Representations, December, 1994 Vol 37. No. 12, Communications of ACM pp36-
[**GLIN89**] E.P.Glinert, out of flatland:towards 3-D visual programming, IEEE procedings 2nd. Fall Joint computer conference,1987,pp292-299.
[**GLIN90a**] E.P.Glinert, Visual programming Enviroments:system. 1990.
[**GLIN90b**] E.P.Glinert, Visual programming Enviroments:application.1990.
[**GOOD** 89] Goodman A.M., Haralick R. M., Shapiro L.G., "Knowledge-based computer vision", Computer, December 1989.
[**GREE** 89] Greene Diane, An implementation and perfomance analysis of spatial data access methods, Proc, 5th. Int. Conf. on Data Engineer. Feb. 6-10, Los Angeles Calif., 1989.
[**GUTT 84**] Guttman A., R-trees: a dynamic index structure for spatial searching, Sigmod Record. Vol. 14, No. 2, Boston MA, June 18-21, 1984.
[**HANA 89**] Hanan Samet, Applications of spatial data structures:computer graphics, image processing and GIS, Addison-Wesley, 1989.
[**HART89**] H.R. Hartson and D. Hix, 'Human-computer interface development: concepts and systems, "ACM Comput. Surveys Vol.21, Mar. 1989.
[**HEAL 81**] healy D.J., and O.R. Mitchell, Digital videoband with compression using block truncation coding, IEEE Trans. Comunications, COM-29, No. 12, 1981.
[HENR93] Henry F.Korth, Abraham Silberschatz, Fundamentas de base de datos. MeGrawHill, 1993.
[**HIRA90**] M. Hirakawa, M. Tanaka, and T. Ichikawa, "An Iconic Programming system, HI-Visual", IEEE Trans. Software Eng., Vol 16, No. 10, Oct. 1990, pp. 1178-1184.
[**INEG** 92] INEGI, Catalogo de productos INEGI,1992.
[**JACK** 88] Jack A. Orenstein and Frank A. Manola, PROBE spatial data modeling and query processing in an image database application, IEEE Transactions on software engineering, Vol.14, No. 5, 1988 pp611-628.
[**JAGA** 89] Jagadish H. V., O'Gorman L., An object model for image recognition, Computer, December 1989.
[**JOSE 94**] José F.G.Mendes, Un modelo de sistema de información espacial para el control del desarrollo urbano. Memoria de congreso IBEROAMERICANO de sistemas de información de base geografica, ciudad de méxico, marzo, 1994 pp165-173.
[**JU 94**] Shi-Guang Ju, Segio V. Chapa and Pedro Alday, Visualization of the database, procedings of the 4th international conference, Pacific graphics'94/CADDM'94, pp531-536, August 26-29, 1994, Beijing, China.

[JU94a] Ju Shiguang, y Chapa V. Sergio, Base de datos geográficas con aplicación a la exploración petrolera. Segundo congreso internacional sobre aplicaciones de cartográfia digital, percepción remota y sistemas de información geográfica, 21-25 de noviembre de 1994. Aguascalientes, México.

[JU 95] Ju Shiguang and Sergio V.Chapa, The development of the pictorial database for petroleum exploration. Proceedings 1995 IEEE International conference on systems,man and cybernetics. Oct.22-25,1995, Vanconver, canada. Vol. 3, pp2806-2811.

[JU 92] Ju Shiguang etc. Implementación de GLSTMPS sistema. Revista de computación. No.6, 1992, pp465-470.

[JU96] Ju Shiguang and Chapa V. sergio, A visual query language for the database of petroleum exploration. Journal of visual languages and computing.(por publicarse)

[KAST 89] Kasturi R., J. Alemany, "Information extraction from images of papaer-based maps", IEEE Trans. on Software Engineering, Vol. 14 No. 5, May 1988.

[KUNT 80] Kunt M., Electronic file for X-ray pictures, in pictorial information systems, S.K. Chang, K.S. Fu eds., Springer Verlag 1980.

[MARC 90] Marc Eisenstadt,john Domingue etc. Visual knowledge engineering, IEEE Transactions on software engineering, Vol.16, No. 10, 1990 pp1164-1177.

[MAX90] Max J. Egenhofer, Manipulating the graphical representation of query results in geographics information systems, proceedings of the 1990 IEEEWorkshop on visual languages, pp 119-124

[MCCO87] McCormick, B.H., Defanti, T.A., and Brown, M.D. Visualization in Scientific Computing - a Synopsis. IEEE Comput. Appl. Graph.. 7, 4(1987), pp61-70

[MEND96] Mendez Segundo Laura, DALI: herramientas para la visualización de datos, Tesis de maestria CINVESTAV IPN, 1996, por publicación.

[MICH 88] Michele Angelaccio, Tiziana Catarci and Giuseppe Santucci, QBD*: A graphical query language with recursion. IEEE Transactions on software engineering, Vol. 16, No.10, 1988 pp1150-1163.

[MIGU 94] Miguel Leal Rosales, SIG para el estudio del desarrollo socieconomico de méxico. Memoria de congreso IBEROAMERICANO de sistemas de información de base geografica, ciudad de méxico, marzo, 1994. pp93-102.

[MOHA 88] Mohan L. and R.L.Kashyap, An object-Oriented knowledge representation for spatial information, IEEE Transactions on software engineering, Vol.14, No. 5, 1988 pp 675-681

[MORI85] M. Moriconi and D.F. Hare, "Visualizing Program Design throught PegaSys, "Computer, Vol. 18, No. 8, Aug. 1985, pp. 72-85

[NACH96] Ignacio Vega Paez, Enfoque relacional de la base de datos espacial, Tesis de maestria, CINVESTAV IPN, por publicación.

[NICK 88] Nick poussopoulos and Christos Faloutsos, An efficient pictorial database systema for PSQL. IEEE Transactions on software engineering, Vol.14, No. 5, 1988 pp639-650.

[NIEM 81] Niemann H., Pattern analysis, Spronger-Verlag, 1981.

[NIEV 84] Nievergelt J., et. al., The Grid file: An adaptable, symmetric multikey file structure, ACM TODS, 9(1), pp 38-71, March 1984.

[NING 80] Ning-San Chang and King-sun Fu, Query-by-pictorial-example, IEEE Transactions on software engineering, Vol.Se-6, No.6 1980, pp519-524.

[PEUC 75] Peucker,T.K., y Chistiman, N. Cartographic data structures, The american cartographer 2, April 1975, pp55-69.

[PIZA 89] Pizano A., Klinger A., Cardenas A., Specification of spatial integrity contraints in pictorial databases, Computer, December 1989.

[LARR 91] Larry Leifer, Machiel Van der Loos, Visual language programming: for robot command control in unstructured environments, Proceedings 1991 IEEE Workshop on visual languages, pp31-36.

[LEE 89] Lee S., Sha M., Yang W., "Simmilarity retrieval of iconic image database", Pattern Recognition, Vol 22, No, 6, 1989.

[LEE 89a] Lee S., Shan m., "acces methods of image database", International Journal of Pattern Recognition and AI, Vol 4, Nr. 1, March 1990.

[LIN 79] Lin B.S., S.K. Chang, Picture algebra for interface with picture database system, Proc. COMPSACD'79, 1979.

[LUIS93] Luis Roberto, Tecnicas para modelados de fractales, tesis de maestría, CINVESTAV del IPN, 1993

[RANG 88] Rangachar Kasturi and Juan Alemany, Information extraction from images of paper-based. IEEE Transactions on software engineering, Vol14. No.5, 1988. pp671-675.
[RAUL 90] Raul Hernandez Stefanoni, Sergio Chapa,Normalización de base de datos en C. Tesis amaria,1990, CINVESTAV del IPN.
[ROBE 90] Robert V.Rubin, James Walker and Eric J.Golin, Early experience with the visual programmer's workbench, IEEE Transactions on software engineering, Vol.16, No. 10, 1990, pp1107-1121.
[ROBL92] L.Altamirano Robles, S.Chapa Vergara,S.Pfleger, sistemas de base de datos pictograficos, Tesis amaria, 1992,CINVESTAV del IPN.
[ROBI 81] Robinson J.T., The K-d-B-tree: Asearch structure for large multidimensional dynamic indexes, in Proc. ACM-SIGMOD, 1981, pp. 10-18.
[ROUS 85] Roussopoulos N., Leifker D., Direct spatial search on pictorial data bases using packed R-Tress, Proc., ACM SIGMOD, May 1985.
[ROUS 88] Roussopoulos n., Faloutsos C., Sellis T., An efficient pictorial database system for PSQL, IEEE Trans. on Software Engineering, Vol. 14, No. 5, May 1988.
[SAME 90] H.Samet, The designing and analysis of spatial data structures, Addison-wesley,1990.
[SATO 91] Satomi Kaneko, Hiroyuki Ikemoto and Yoichi Kusui, Approach to designing Easy-to-understand icons, Proceedings 1991 IEEE Workshop on visual language, pp246-253.
[SELL 87] Sellis T., Roussopoulos N, Faloutsos C., Then R^+-tree: A dynamic index for multi-dimensional objects, in Proc. 13th. Int. Conf very large Data Bases, Sept. 1987.
[SELI 89] Selinger,P.G. et.al Access path selection in a relection in a relational database management system, Reprint in ärtificial intelligence&databases, Morgen kaufman Inc. 1989,pp511-522.
[SHU 92] N.C.Shu, Visual programming, Van Nostrand Reinhold, New York,1992.
[SIER95] Sierra Romero Noe, HErramienta VIsual para la COnstruccion de Programas, tesis de maestria. 1995, CINVESTAV del IPN.
[SILV 86] Silver H., S.K. Chang, Picture information measures, Policy ans Information, Vol. 10, No. 1, June 1986.
[SHU92] Nan C. Shu, Visual programming, Van Nostrand Reinhold, 1992.
[STAE 91] Staes F., L.Tarantino and A.Tiems, A graphical query language for object oriented databases, Proceedings 1991 IEEE Workshop on visual languages. pp205-210.
[TAMU 84] Tamura H., Yokoya N., Image database systems: a survey, Pattern Recognition Vol. 17, No. 1, 1984.
[TANA 88] M.Tanaka and T.Ichikawa, A visual user interface for map information retrieval based on semantic significance. IEEE Transactions on software engineering, Vol. 14, No. 5. 1988 pp666-670.
[TEOR 90] Database Modeling and Design: The Entity-Relationship Approach. Morgan Kaufmann, 1990.
[TETS 92] Tetsuto Yoshihawa and Astem Ri, A visual knowledge representation language for layout problem, Proceedings of the 1992 international conference on computer language. pp181-189.
[THOM 88] Thomas Joseph and Alfonso F. Cardenas, PICQUERY: A high level query language for pictorial database manegement,IEEE Transactions on software engineering, Vol. 14, No.5, 1988. pp630-638.
[TODD 75] Todd,S.Prtv., An efficient implementation for large relational databases, Framingham, Mass.,sept. 1975.
[TSIC 83] Tsichritzis D., et. al., Multimedia office filing system, Proceedings of VLDB, 1983.
[WU 82] Wu J. k., and R. E. Burge, Adaptive bit allocation for image compresion, Comp, Grphics and image processing, 19, 1982.
[WU 87] Wu J. k. Zhang W. M. Adaptive transform image compresion by using texture analysis, Communications china, 1987.
[RAYM91] Raymond A. Lorie, Jean-Jacques Daudenarde, SQL & its applications. PRENTICE HALL, 1991.