B18-16190

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL
INSTITUTO POLITÉCNICO NACIONAL

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

DALI
HERRAMIENTA PARA LA REPRESENTACIÓN GRÁFICA DE INFORMACIÓN

TESIS QUE PARA OBTENER EL GRADO DE:

MAESTRA EN CIENCIAS
EN LA ESPECIALIDAD DE INGENIERÍA ELÉCTRICA

PRESENTA:

LIC. LAURA MÉNDEZ SEGUNDO

DIRECTOR DE TESIS:

DR. SERGIO VICTOR CHAPA VERGARA

México, D.F. Diciembre de 1998

A mis padres Rufina y Carlos,

A mi esposo Antonio,

A mis hermanos Fabián y Beatriz.

En memoria de mis abuelos Lolita, Eleazar y Leovigilda.

AGRADECIMIENTOS

A Dios,
por haberme permitido llegar a esta meta y lograr todo lo que hasta ahora he logrado.

A mis padres Rufina y Carlos,
por el apoyo incondicional que he recibido a lo largo de toda mi vida
de quienes siempre me sentiré orgullosa
a quienes amo y respeto y
quienes estoy segura que comparten mi alegría.

A mi esposo Antonio,
por el amor, paciencia, apoyo y comprensión que me ha brindado durante esta etapa de mi formación profesional y durante todo el tiempo que he estado a su lado.

A mi suegra Cruz María,
por el cariño y el apoyo que me brindó para poder dedicarle el tiempo necesario a la realización de este trabajo.

A mis hermanos Fabián y Beatriz,
por su cariño y su apoyo.

Al Dr. Sergio Victor Chapa Vergara,
por el apoyo, las observaciones y comentarios recibidos durante la realización de esta tesis.

A mi jurado el Dr. Pedro Mejía Alvarez y el Dr. Manuel González Hernández
por la revisión y comentarios recibidos sobre este trabajo.

A todos mis familiares y amigos
de quienes siempre recibí palabras de aliento.

A todos mis amigos y compañeros del CINVESTAV
con quienes compartí esta etapa de mi formación profesional y de quienes recibí múltiples consejos y palabras alentadoras.

A todo el personal de la Sección de Computación
por su apoyo y amistad.

Al Consejo Nacional de Ciencia y Tecnología CONACYT y al Centro de Investigación y Estudios Avanzados del IPN, CINVESTAV
por el apoyo recibido para la realización de mi maestría.

Muchas Gracias.

Tabla de Contenido

Introducción

El objetivo de este proyecto de tesis es desarrollar una herramienta que permita visualizar la información contenida en una base de datos ordenando y clasificando los datos para interpretarlos y comunicarlos al usuario de una manera gráfica fácil de comprender. De esta forma el usuario será capaz de analizar el comportamiento de la información poder realizar una buena toma de decisiones.

Este trabajo tiene como antecedente un proyecto más amplio: "Programación Automática a partir de Descriptores de Flujo de Información" propuesto por el Dr. Sergio Víctor Chapa Vergara. El proyecto se desenvuelve principalmente en dos líneas de investigación: Lenguajes y Ambientes Visuales y, Bases de Datos.

Este proyecto que tiene el objetivo de tener un sistema en donde prevalezca un entorno visual en problemas en bases de datos; desde el diseño conceptual de las bases de datos hasta su explotación, como puede ser minería de datos está conformado por diversas partes que son los siguientes proyectos han sido y son nuevas plataformas de desarrollo: LIDA "Lenguaje Iconográfico de aplicaciones" [4], EVE "Entidad Vínculo Extendido" [14], HEVICOP [3] y DALI "Herramienta para la representación gráfica de información" , como lenguajes y ambientes visuales. Y como gestores de bases de datos las tesis: "Representación de Sistemas de Consulta en "Tableaux" y su implementación en una base de datos relacional []; Visualización de una base de datos espacial [13]; y el mismo Sistema LIDA [4].

El proyecto de "Programación Automática a partir de Descriptores de Flujo de Información", inicia con la propuesta de un nuevo lenguaje visual para resolver gestiones en bases de datos. LIDA es un sistema de programación visual cuyo paradigma de gráficas de íconos y líneas de flujo, tienen un enfoque de lenguaje de flujo de datos[4]. El sistema se sustenta con una base de datos con modelo relacional. Mediante LIDA es posible resolver consultas típicas de una base de datos y resolver problemas de aplicaciones mediante la incorporación de iconos de procesos y funciones que tienen un nivel de abstracción alto con la definición visual y simbólica [4]. Uno de los aspectos importantes de LIDA es la representación de un gran contenido visual a través de los diagramas denominados flujogramas; estos tienen una representación bidimensional que permiten una ejecución en paralelo. La ejecución puede ser interrumpida en algunos puntos del diagrama con el fin de mostrar resultados parciales los cuales son tablas de la base de datos o resúmenes.

El poder tomar decisiones considera que se puedan ver de una manera visual los datos con el fin de cambiar procesos, fórmulas o parámetros. Es en estos puntos donde se requiere tener el desplegado de gráficas que permita al usuario ir explorando la información y tomar decisiones. El resultado fue la propuesta de un sistema automático visual llamado DALI.

Dado que los sistemas están sustentados en el modelo relacional de datos, el proyecto consideró un sistema que pudiera automáticamente generar esquemas de bases de datos. EVEX es el sistema para el diseño conceptual de bases de datos con la metodología entidad-vínculo extendido [14]. El sistema implementado en Xwindows permite diseñar conceptualmente bases de datos dejando un modelo lógico relacional y uno físico basado en un descriptor de archivos. La herramienta EVEX auxilia en el modelo conceptual con los módulos de edición de atributos y de diagramas del sistema EVE. El modelo lógico se obtiene mediante la transformación del diagrama de entidad a esquemas de relación y la verificación como esquemas en tercera forma normal.

Algunas veces, aplicaciones específicas mediante un procedimiento algorítmico requieren ser implantadas. HEVICOP es una herramienta visual para la construcción de programas, que utiliza iconos que son piezas de software y a partir de esta definición, es capaz de generar líneas de código en lenguaje de programación "C" [3].

HEVICOP surge de una propuesta de abstraer LIDA omitiendo las líneas de flujo de datos y mediante el ensamblaje de un conjunto de iconos elementales, tener procesos más complejos. El resultado fue el sistema HEVICOP que basado en el paradigma de BLOX se tiene un sistema basado en esqueletos de control y es dirigido por sintaxis [3].

La naturaleza visual de todo el proyecto, comprometió a la necesidad de tener un ambiente gráfico para mostrar la información, dando como resultado el proyecto del sistema DALI. Cuyo objetivo se describe a continuación:

El contenido de este trabajo se describe a continuación:

En capítulo 1 se describe el concepto de visualización, la clasificación de medios ambientes visuales y lenguajes visuales. Se realiza la distinción entre visualización científica y de datos. También se describe la importancia del uso de tablas y el de gráficas para la representación de información. Se describen los diferentes tipos de gráficas que se utilizan para la visualización de información.

En el capítulo 2 se describen específicamente las técnicas utilizadas para la visualización de información con múltiples variables. La primera parte del capítulo describe 3 métodos de visualización: gráficas de función, polígonos y árboles. La segunda parte del capítulo se enfoca principalmente a las Caras de Chernoff que es la técnica desarrollada en el presente trabajo. En la última parte del capítulo, se presentan las conclusiones del mismo.

En el capítulo 3 se describe la metodología utilizada por el sistema para realizar el análisis de la información que se va a visualizar. Describe los procedimientos y las técnicas para determinar la gráfica más apropiada para la visualización de información dependiendo de la naturaleza, el tipo y la cantidad de datos.

En el capítulo 4 se describe la arquitectura del sistema DALI y la función de cada uno de sus componentes.

En el capítulo 5 se describe el ambiente gráfico Xwindows sobre el cual fue desarrollado DALI. Se definen los conceptos generales de Xwindows: displays y pantallas, modelo cliente-servidor, eventos, capas que lo constituyen. Se describe la definición del ambiente gráfico utilizando las bibliotecas de Motif y Xlib. Se definen también las principales funciones gráficas para este ambiente.

En el capítulo 6 se describe el entorno de programación del sistema DALI, sus pantallas y su operación. Así también, se presentan dos casos para ejemplificar el funcionamiento del sistema.

Al final del presente trabajo se presentan las conclusiones generales.

Capítulo 1

VISUALIZACION DE DATOS

1.1 INTRODUCCION

La historia de la visualización se ha ido desarrollando de acuerdo a la tecnología disponible, conforme ha avanzado la civilización, la gente ha necesitado manejar cantidades de datos numéricos para ser almacenados, sumados e interpretados. Todos los fenómenos económicos, demográficos y políticos requerían ser representados de alguna manera. En 1786 John Playfair [1] publicó las primeras gráficas de series de tiempo económicas revolucionando la visualización de datos; los conceptos abstractos como tiempo y número podrían ser visualizados en términos de atributos más concretos tales como posición, longitud y tamaño.

La Visualización es el proceso de transformación de objetos, conceptos y números a una imagen que el ojo humano pueda percibir, surgiendo de la necesidad de observar un conjunto amplio de información de una manera fácil y sencilla [1].

La gran cantidad de datos generados por supercomputadoras y otras fuentes de datos hace muy difícil a los usuarios examinar cuantitativamente la información, es por ello que se requieren herramientas que faciliten el trabajo de observación, exploración y análisis de los datos. En la actualidad, con el surgimiento de las gráficas raster, los investigadores pueden convertir campos enteros de variables (densidad, presión, entre otros) a imágenes a color. Las secuencias del DNA, los modelos moleculares, el rastreo de imágenes médicas, mapas cerebrales, simulaciones de vuelo, simulaciones de flujos de fluido y muchas cosas más, necesitan ser expresadas visualmente.

La visualización de datos se usa rutinariamente para resumir grandes cantidades de datos. Existen muchas formas de diagramas y gráficos que pueden usarse para visualizar datos dependiendo de la cantidad y complejidad de los mismos. Existen sistemas como los de la NASA que son muy sofisticados y que permiten visualizar toda la información recopilada por sus satélites, hasta sistemas sencillos que pretenden representar el comportamiento de la información a través de un tiempo determinado.

La visualización transforma grupos de datos a imágenes que la gente pueda entender rápidamente. El objetivo de la visualización es extraer conocimiento a partir de datos crudos. Los trabajos de investigación sobre la visualización tienen como objetivo investigar aquellos mecanismos en humanos y en computadoras que permiten percibir, interpretar, usar y comunicar información visual[1].

La visualización de datos es todo aquello que tiene que ver con el entendimiento de relaciones entre números, no entendiéndolos como números individuales sino como patrones y relaciones que existen en grupos de números. Estos patrones, tendencias y relaciones tienen que estar asociadas a algún modelo del dominio de interés y se deben utilizar para cerrar el abismo existente entre el entendimiento del usuario acerca de la situación que él interpreta y la situación real.

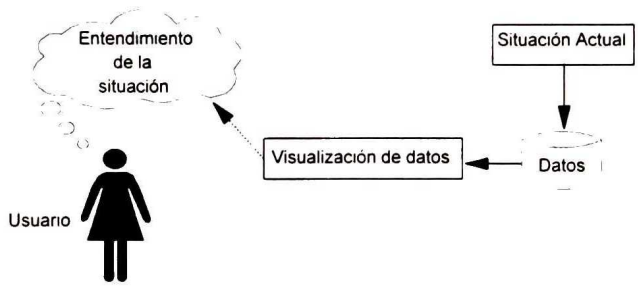


Figura 1.1.- Función de la visualización en el entendimiento humano

La representación visual de información aprovecha la vasta y muchas veces no utilizada capacidad del ojo humano para detectar información de ilustraciones y dibujos.

1.2. CLASIFICACIÓN DE AMBIENTES Y LENGUAJES VISUALES

Actualmente, se pueden clasificar los trabajos referentes a visualización como se muestra en la figura 1.2

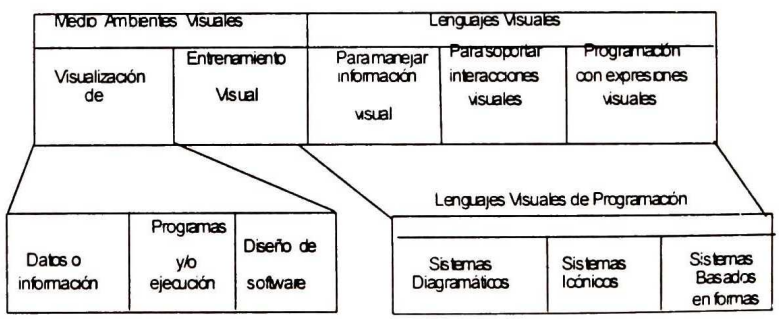


Figura 1.2.- Clasificación de Ambientes y Lenguajes Visuales

A continuación se describe brevemente cada una de las categorías que se indican en la figura 1.2.

1.2.1 Medios ambientes visuales

Visualización de Datos o de Información

Dentro de esta área, la información o datos, se almacenan internamente en bases de datos tradicionales, pero es expresada en forma gráfica y presentada al usuario dentro de un contexto espacial. Los usuarios pueden analizar la superficie gráfica que en algún punto se pueda expander para observar mejor aquellos detalles de interés. Este enfoque permite diferentes tipos de consultas sin la necesidad de usar un teclado, solo utilizando un dispositivo apuntador (mouse). En esencia, estos sistemas son utilizados principalmente como un medio de manipulación directa para la recuperación de información, usando una vista gráfica de una base de datos para visualizar la información recuperada. Un ejemplo se muestra en la figura 1.3. La información nominal de una BD puede contener coordenadas, colores, dirección de trazos, etc. El sistema traduce estos datos a una representación gráfica permitiendo el análisis y estudio por parte del usuario.

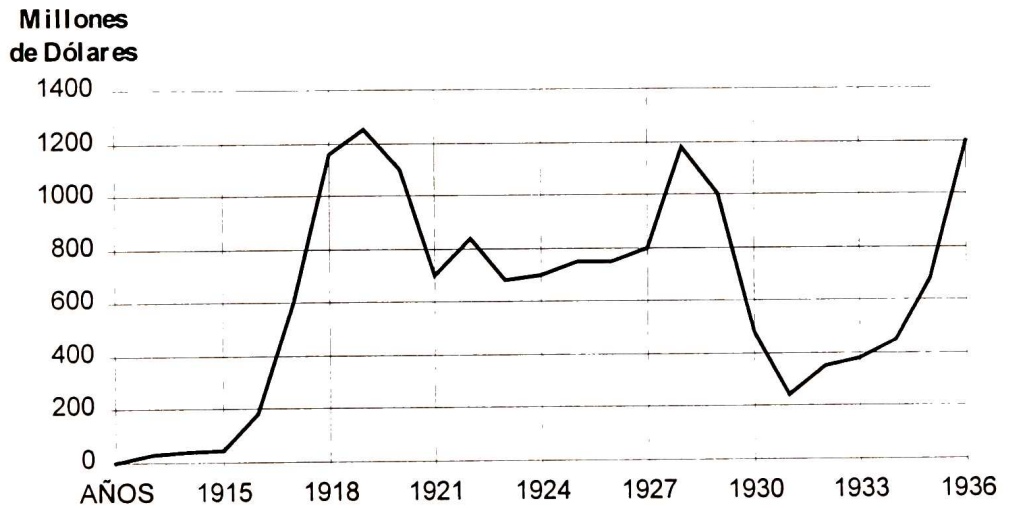


Fig. 1.3 Impuestos Federales sobre la renta individual en Estados Unidos

Visualización de programas y/o su ejecución

Esta categoría consiste en soportes gráficos para los programas que están escritos en lenguajes tradicionales de texto. La visualización de los programas comprende el estado y resultados en tiempo de ejecución. El objetivo es usar un dispositivo gráfico de alta resolución para facilitar el desarrollo y pruebas de programas. Las actividades de esta área tienen un amplio espectro, que va desde una "bonita impresión" del código fuente, hasta un análisis de la ejecución de un programa en múltiples vistas o en forma animada. Un ejemplo es el sistema HEVICOP desarrollado en la sección de computación del CINVESTAV dentro de la línea de Lenguajes Visuales y Visualización[3].

Visualización del Diseño de Software

La visualización del diseño de software consiste en proporcionar los ambientes gráficos para obtener facilidades en diversos aspectos del desarrollo del 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. Como ejemplo se menciona el sistema EVE[14].

Entrenamiento Visual

Este tipo de sistemas basados en programación gráfica tienen la intensión de solucionar problemas mediante una programación más rápida. Dentro del medio ambiente de estos sistemas, el usuario no necesita construir mentalmente los efectos de sus instrucciones y programas. Todas las acciones se desarrollan sobre la pantalla. La sintaxis del lenguaje en sentido tradicional está ausente desde el punto de vista del usuario. Muchos de los recientes sistemas de "Programación por Ejemplos" y "Demostracional" entran dentro de este grupo. La siguiente figura muestra parte de una sesión de trabajo con Kaestle, un sistema para el aprendizaje en el manejo de listas[3].

1.2.2 Lenguajes Visuales

Lenguajes para el Manejo de Información Visual

Los lenguajes dentro de esta categoría están diseñados principalmente para el procesamiento de información visual o imágenes. Estos sistemas están motivados por la necesidad de tener lenguajes fáciles de usar para la manipulación y consulta de datos pictóricos. Generalmente el lenguaje permite referenciar directamente imágenes, sin embargo, aunque los elementos que se manejan son gráficos, el lenguaje en si es textual.

Lenguajes que Soportan Interacción Visual

Los avances en el hardware, han hecho posible el uso de íconos u objetos gráficos como un medio de comunicación con las computadoras. Sin embargo, sin el software, los íconos e imágenes no tienen vida dentro del mundo de la programación. Por eso, es natural que surgieran lenguajes diseñados para definir, crear y manipular símbolos gráficos. Los lenguajes dentro de esta categoría, en general, soportan representación e interacción visual, pero los lenguajes mismos son textuales, no visuales. Un ejemplo se clásico de este tipo de sistemas es Windows de Microsoft como se observa en la figura 1.4.

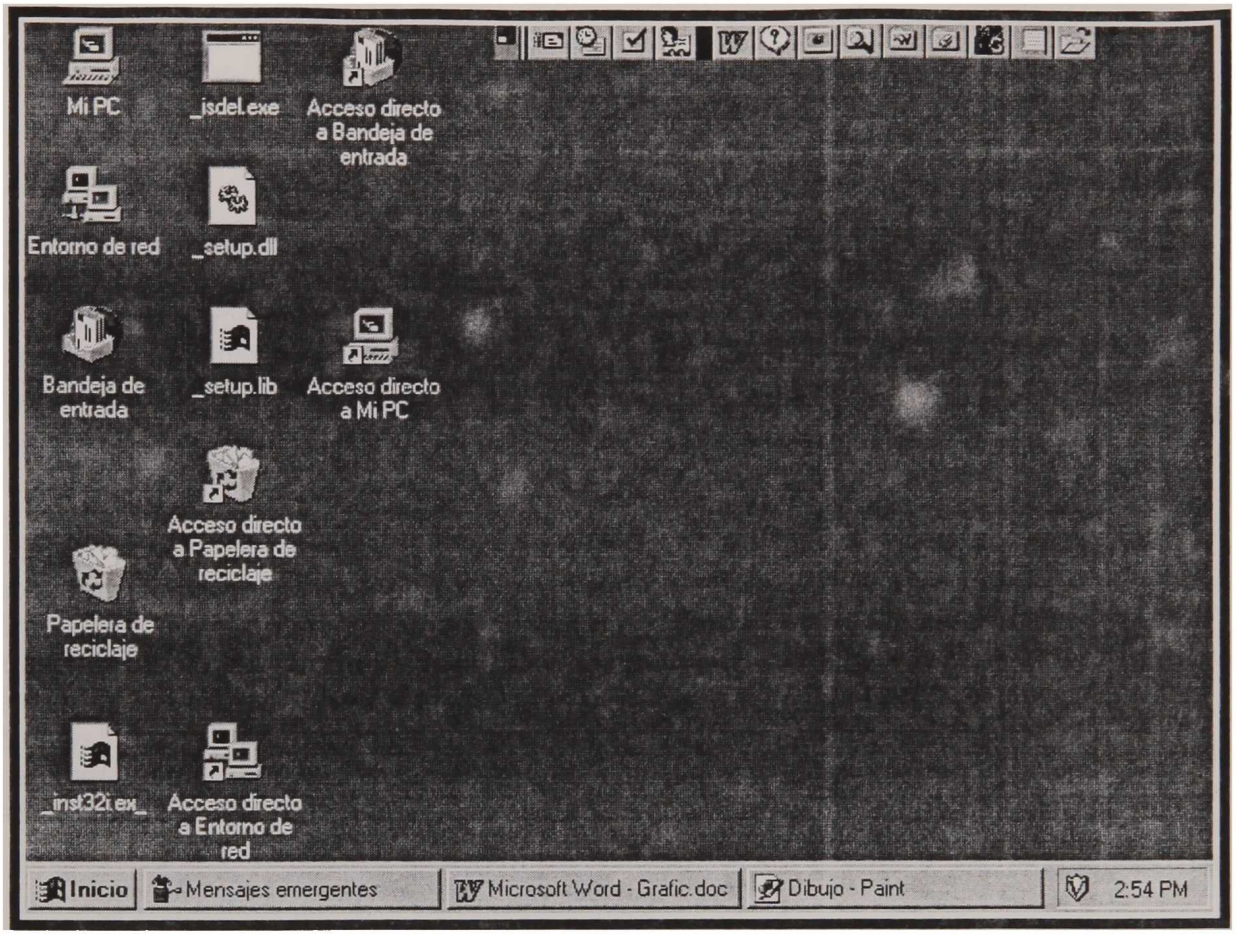


Figura 1.4.- Pantalla de Windows 95

Lenguajes Visuales de Programación con Expresiones Visuales

En esta categoría de los lenguajes visuales, la idea central es permitir a los usuarios programar con expresiones visuales. Estos lenguajes son un nuevo paradigma para expresar sistemas computacionales. Nos ofrecen la posibilidad de manipular directamente objetos computacionales (datos, programas, proposiciones, etc.). Un programa visual escrito en algún lenguaje visual dado, consiste de un arreglo especial de íconos. Cuando los íconos son metáforas adecuadas para objetos computacionales, el significado de una proposición visual tal como es interpretada por el hombre tiende a ser similar o posiblemente a coincidir con el significado que es asimilado por un sistema. El grado de similitud depende de qué tanto las construcciones mentales correspondan a las construcciones del lenguaje visual. El sistema LIDA [4] es un buen ejemplo de este tipo de sistemas. LIDA permite definir gráficamente aplicaciones de Bases de Datos.

Sistemas Diagramáticos

Dentro de los principios de diseño, la mayoría de los lenguajes visuales de Programación pueden dividirse en tres categorías. En un extremo, los esquemas de flujo y los diagramas que son usados en papel son incorporados en construcciones de programación como una extensión de los lenguajes de programación convencionales, o son empaquetados en unidades interpretables por una máquina para ser usados junto con ellos. Dentro de esta categoría se encuentra el sistema HEVICOP [3].

Sistemas Icónicos

En el otro extremo, los símbolos icónicos o gráficos son deliberadamente diseñados para jugar un papel central dentro de la programación. En años recientes, este tipo de sistemas han mantenido gran auge. El papel que juegan los íconos está restringido a la representación del ámbito de trabajo y a los comandos definidos dentro del mismo.

1.3 VISUALIZACIÓN CIENTÍFICA Y DE DATOS

Otra clasificación de visualización la realiza Parsaye [1]

Parsaye sugiere dos tipos de visualización

- Visualización Científica y
- Visualización de Datos

La *visualización científica* usa animación, simulación y gráficas complejas generadas por computadora para crear modelos visuales de estructuras y procesos que no pueden ser vistos de otra manera . Las visualizaciones científicas son imágenes generadas por computadora que facilitan una gran variedad de tareas como suele ser el entendimiento de mecanismos y las predicciones de un modelo científico completo.

La *visualización de datos* es generalmente mucho más simple y global, debido a que nos podemos olvidar de muchos detalles que nos quitan atención en los conceptos. Es un componente esencial en aplicaciones de Bases de Datos Inteligentes que pueden desplegar y presentar información en una forma que fomenta la interpretación, selección y asociación apropiada. Explota la habilidad de la gente para extraer una gran cantidad de información en un período de tiempo corto.

La forma gráfica alternativa a la forma textual o escrita, tuvo sus objeciones históricas naturales por el trabajo de diseño y dibujo que se requería, presentándose mas como un arte con sus dificultades propias del problema. De esta manera, la visualización de datos se presenta más como un arte en donde se requieren una gran cantidad de técnicas de dibujo para poder enfrentar ciertas dificultades propias del problema que estamos tratando.

Reconociendo el potencial de la visualización de datos se puede obtener mayor información de una manera gráfica que de una forma textual o escrita. La visualización disminuye la carga de razonamiento numérico a razonamiento visual, permitiendo que el usuario capte la información de una forma más sencilla y sin complicaciones.

La visualización de datos es una manera de reducir la complejidad percibida de información mientras retiene los elementos esenciales de esa información y promueve un mejor entendimiento de los datos.

Este trabajo se fundamenta en la importancia que tiene la visualización de datos para la toma de decisiones y el estudio del comportamiento de los datos.

En la actualidad la programación visual, la visualización de datos y las herramientas de hypermedia están teniendo gran auge debido a los avances tecnológicos en cuanto a equipo y dispositivos. Existen sistemas que cuentan ya con herramientas para la visualización de información que permiten al usuario apreciar de manera gráfica la información contenida en la base de datos.

Se debe tomar en cuenta que la visualización incluye tanto el entendimiento de imágenes como la síntesis de imágenes.

1.4 TABLAS Y GRÁFICAS

La información puede ser presentada a través de tablas o gráficas por lo que es importante conocer la función de cada uno de estos conceptos para identificar su utilidad.

Para que la información pueda ser estudiada o analizada, es necesario ordenarla, de manera que sea fácil de comprender. Una manera de resumir y organizar la información es a través de **tablas**. Las tablas de datos se usan para propósitos de trabajo pero no son necesarias para comunicar resultados finales. Las tablas proporcionan un registro detallado de la información para un posible análisis posterior.

1.4.1 Tablas

Tablas para prueba

Son tablas que se utilizan para la comprobación de información ya que mucha gente no confía en el analista y requiere de tablas de datos detallados para observar los resultados que el analista ha obtenido. Estas tablas son por lo general sencillas pues la cantidad de información no es extensa. La siguiente tabla es un ejemplo de una *tabla de prueba.*

	Area				
1969	No	Ea	We	So	Promedio
Q I	98	75	50	48	68
Q II	92	75	57	42	67
Q III	101	(100)	(80)	50	(83)
Q IV	90	74	51	39	64
Promedio	95	75	53	45	67

Tabla 1.1 Resultados de 1969 Cuarto por Cuarto

Esta tabla representa estadísticas sobre una investigación arqueológica en donde se representa la cantidad de huesos encontrados en las distintas zonas de investigación: Norte, Sur, Este, Oeste clasificados en trimestres (QI corresponde al primer trimestre de 1969) y es una forma en la que el analista puede proporcionar una prueba tabular de sus resultados [12].

a) El puede demostrar que sus datos tienen una cierta dispersión mencionando que cada valor tiene una desviación en promedio de cerca de 3 unidades.
b) Puede demostrar que actualmente está buscando los datos en algún detalle mencionando que la desviación es aparentemente irregular y que el tamaño no varía mucho de área en área o de cuarto en cuarto.
c) Puede demostrar que presenta los datos tal cual estos son y que no trata de esconder ninguna información con los valores que muestra entre paréntesis.

En resumen, puede representar y explicar el resultado de su investigación utilizando como herramienta una tabla simple de prueba.

Para casos en los que el análisis es simple (como promedios y porcentajes), es suficiente el uso de una tabla de prueba. Para análisis más complejos que utilizan una mayor cantidad de información se requieren de otras técnicas de representación.

Tablas para Comparar

En la siguiente tabla se tiene la representación de los datos correspondientes a dos años, los valores obtenidos (Obt.) y Esperados(Esp.). En esta tabla se puede observar que la variación es poca y que los valores esperados se vuelven importantes.

	Area								Promedio	
	No		Ea		We		So			
	Obt.	Esp.	Obt.	Esp.	Obt.	Esp.	Obt.	Esp.	Obt.	Esp.
1969 Q I	98	95	75	75	50	53	48	44	68	67
Q II	92	95	75	75	57	53	42	44	67	67
Q III	101	95	(100)	75	(80)	53	50	44	(83)	67
Q IV	90	95	74	75	51	53	39	44	64	67
1970 Q I	96	95	74	75	53	53	46	44	67	67
Q II	94	95	77	75	49	53	50	44	68	67
Q III	91	95	72	75	53	59	42	44	66	67
Q IV	98	95	76	75	53	53	37	44	66	67
Promedio	95	95	75	75	53	53	44	44	67	67

Tabla 1.2 Valores obtenidos y esperados

Una comparación de este tipo requiere una tabla bien organizada. Necesita hacerse solamente con una base ilustrativa pequeña. Hasta este punto la información es de fácil comprensión pero, ¿qué sucedería si se quiere representar el valor obtenido y esperado de un plazo de cinco años? La tabla se vería con demasiada información y sería un poco más difícil poder observar las tendencias y variaciones. Las gráficas en este caso pueden ser más efectivas aquí.

Tablas para el Registro

Otro de los usos más comunes para las tablas es como herramienta de registro. En lugar de presentar datos como ayuda para una comunicación inmediata, muchas tablas conservan datos para posibles usos posteriores. Ejemplo de esto son los registros de censos y otras estadísticas oficiales donde los datos se publican y todavía no se analizan todavía. Los datos se proporcionan para tenerlos registrados por lo que cualquiera puede referirse a ellos en alguna etapa futura.

El almacenamiento de datos no explotados se está volviendo más común en nuestros días con el desarrollo de grandes almacenes de bases de datos y sistemas de información. La idea es proporcionar "todos los hechos" al presionar un botón. Pero esto puede ser exagerado ya que grandes sumas se han invertido en elaborar métodos de carga de datos aún cuando nadie sabía qué hacer con los datos antes de que se perdieran.

Ejemplos menos extremos de almacenamiento de datos son los casos en que los datos ya han sido analizados y sintetizados pero los detalles completos se proporcionan en caso de que alguien quiera reanalizarlos, generalmente para propósitos de investigación. En este caso, los datos necesitan ser proporcionados en un apéndice o se pueden mantener copias guardadas de datos que pueden estar disponibles para una consulta ocasional [12].

1.4.2 Gráficas

Las gráficas tienen dos usos: para propósitos de trabajo y para la presentación de resultados finales. En el primer caso, el analista puede graficar las lecturas de las primeras etapas del análisis para obtener una impresión visual rápida de cuando una relación es lineal.

El uso de gráficas para la presentación de resultados finales permite simplificar y enfatizar la información más relevante.

En general, las gráficas pueden demostrar un resultado cualitativo como "Existe mucha gente cuya edad es mayor de 15 años" o "Hizo mucho calor en el verano", pero no pueden comunicar resultados cuantitativos.

Las gráficas son hasta cierto punto inútiles para propósitos de análisis detallados puesto que es imposible manipular cualquier lectura. En contraste, con una tabla es fácil formar promedios, tomar diferencias, etc. Más importante aún es que las gráficas

rara vez presentan un resumen memorable de resultados excepto en un nivel cualitativo como "Hace mucho calor en el verano".

Existen muchas formas de diagramas y gráficas que se utilizan para visualizar datos y también existen varias clasificaciones de acuerdo a su función y características. El sistema DALI se enfoca principalmente a la visualización de datos utilizando gráficas estadísticas y caras de Chernoff para datos multivariados[10]. A continuación se describen estas técnicas y algunas otras que posteriormente fueron implementadas en el sistema.

El sentido de la vista es el medio por el cual podemos captar toda aquella información visual que nos rodea para que ésta sea procesada por nuestro cerebro. Los despliegues gráficos nos permiten explotar ese sentido para obtener un análisis profundo de la estructura de datos. A través de las gráficas, se puede representar una gran cantidad de información cuantitativa, el sistema visual humano es capaz de captar rápidamente esta misma cantidad de información, extraer características sobresalientes y también es capaz de percibir el detalle de lo que se está representando. Aún para pequeños conjuntos de datos existen muchos patrones y relaciones que son considerablemente más fáciles de discernir en un desplegado gráfico que por otro método analítico.

De ahí el famoso refrán que dice "Una imagen vale más que mil palabras".

Los métodos gráficos proporcionan herramientas de diagnóstico poderosas para confirmar suposiciones, o cuando no se encuentran suposiciones para sugerir acciones correctivas. Los métodos gráficos se utilizan principalmente en el análisis estadístico de datos.

Las gráficas son un medio para la representación de información que cuenta ya con varios años de existencia, a continuación se describen algunas características y ventajas de estas herramientas que permiten demostrar que las gráficas tienen una mayor ventaja sobre otras técnicas verbales, textuales o numéricas que se utilizan en la representación de información.

- Primero que nada, las gráficas comunican información de manera rápida y directa. No solamente ahorran tiempo sino que permiten, al usuario de la gráfica apreciar a simple vista las implicaciones y características principales. Esto significa que mucha información puede ser percibida con el hecho de hacer una simple observación sobre el contenido de una gráfica en lugar de leer grandes líneas de datos lo cual resulta muchas veces tedioso.

- Las gráficas son poderosas para proporcionar énfasis a un mensaje completo, coherente y decisivo.

- Son reveladoras : puesto que pueden leer claramente los datos, frecuentemente sacan a flote hechos y relaciones escondidas, considerándose de gran ayuda para la investigación.

- Debido a su apariencia las gráficas atraen la atención y mantienen el interés del lector haciendo a los datos más interesantes y llamativos [5].

Principales funciones de las gráficas

Cabe destacar que las gráficas tienen dos funciones principales: *Presentación* y *Análisis* . A través de una gráfica podemos presentar grandes cantidades de datos que podrían ocupar cientos de registros en una base de datos y que a simple vista no serían posibles de comprender e interpretar, en cambio, una gráfica nos permite representar de forma más clara toda esa información. Una vez desplegada la gráfica, es posible hacer un análisis detallado de la misma con mayor facilidad que si se tuviera que revisar registro por registro de una base de datos.

El propósito de una gráfica es ilustrar, describir, interpretar y transmitir información. Su función primaria es la presentación. Su función analítica se indica claramente cuando hay una aplicación específica de técnicas gráficas hacia un problema estadístico bien definido, donde el cálculo, la medida, la exploración y la derivación de relaciones son objetivos básicos[5].

La calidad o grado de aceptación de una gráfica debe basarse en varios aspectos como son: a) su propósito o función; b) la naturaleza de los datos que pretende representar; c) las características de los usuarios; d) las restricciones físicas involucradas en el proceso de construcción de una gráfica tales como tipo, tamaño, calidad de reproducción y el uso de uno o más colores; e) tipo de equipo y material disponible.

Una gráfica también debe contar con las siguientes características:

Seguridad.- Una gráfica no debe ser engañosa ni debe ser susceptible a interpretaciones equivocadas a consecuencia del poco cuidado en su construcción. Tampoco deben ocasionar ilusiones ópticas por la colocación de las líneas o puntos de la misma.

Simplicidad.- El diseño básico de una gráfica estadística debe ser simple, no debe llevar símbolos irrelevantes, superfluos, o triviales. Una buena gráfica no debe ser compleja ni difícil de interpretar, debe contener los elementos necesarios para que sea fácil de comprender y analizar.

Claridad.- La claridad es una de las principales características con que debe contar una gráfica. Si una gráfica es ambigüa o confusa, el proceso de comunicación se puede venir abajo[5].

Todos los puntos anteriores se han tomado en cuenta para el desarrollo de este trabajo para que se cumpla el objetivo definido de servir como una herramienta de visualización de información.

1.4.3 Clasificación de Gráficas

Existen diversas clasificaciones de gráficas de acuerdo a varios autores:

Schmid [5] sugiere que hay básicamente tres propósitos fundamentales para los gráficos: a) ilustración, b) análisis y c) cálculo. Estas tres categorías son similares a las sugeridas por Tukey [6] y se describen como se indica a continuación:

- Las gráficas intentan demostrar lo que ya se ha aprendido por alguna otra técnica.
- Las gráficas nos permiten ver lo que puede estar pasando sobre y debajo de lo que ya ha sido descrito. (Gráficas analíticas) y
- Las gráficas que tienen como función sustituir tablas de datos.

De acuerdo al número de dimensiones las gráficas también se pueden dividir en: *curvas o diagramas lineales*; *gráficas de barras*, que implican comparaciones de una dimensión; *diagramas de área* implican comparaciones de dos dimensiones; *diagramas de volúmen* que implican la visualización de una tercera dimensión y comparaciones de tres dimensiones. Cabe señalar que el sistema DALI sólo genera gráficas en dos dimensiones.

La clasificación propuesta por Parsaye [1] es la siguiente:

- Mapas Estadísticos
- Diagramas Basados en Coordenadas
    - Gráficas de línea o curvas simples
    - Scatter Plots
    - Gráficas semilogarítmicas
- Diagramas Basados en Proporciones
    - Diagrama de Pastel
    - Diagrama de Barras
- Diagramas para datos multivariados
    - Diagramas de Matriz
    - Curvas de Andrews
    - Polígonos
    - Árboles
    - Caras de Chernoff

Mapas Estadísticos

Son representaciones simbólicas de geografía física. Describen lugares geográficos de características particulares utilizando símbolos o letreros , se incluyen cartas oceanográficas y mapas topográficos. Son una forma natural para desplegar la información sobre el ambiente. Muestran la información cuantitativa sobre una base geográfica.

Gráficas basadas en coordenadas

Este sistema tiene sus raíces en Descartes[8], el inventor del sistema de coordenadas. Las líneas (ejes), se usan para definir un sistema de coordenadas donde cada eje presenta atributos diferentes. El valor de cada atributo se representa a lo largo de las líneas del eje. A continuación se detallan sus principales características:

1.- Ejes de Coordenadas: La estructura básica de la gráfica de línea de coordenada rectilínea se deriva de graficar figuras en relación a dos ejes formados por la intersección de dos líneas perpendiculares. La línea horizontal se denomina eje de las X o abscisa, la línea vertical se denomina eje de las Y o eje de las ordenadas. Los valores colocados a la derecha y sobre el origen son valores positivos, los valores colocados a la izquierda o abajo del origen se consideran negativos. Las áreas de coordenadas se denominan cuadrantes y son cuatro, éstas se observan en la figura 1.5.

2.- Divisiones Escalares: Son las porciones en que se dividen los ejes, pueden ser diferentes para los dos ejes.

A este grupo pertenecen las siguientes gráficas:

- Gráficas de línea o curvas simples
- Scatter Plots
- Gráficas semilogarítmicas

que se describen a continuación.

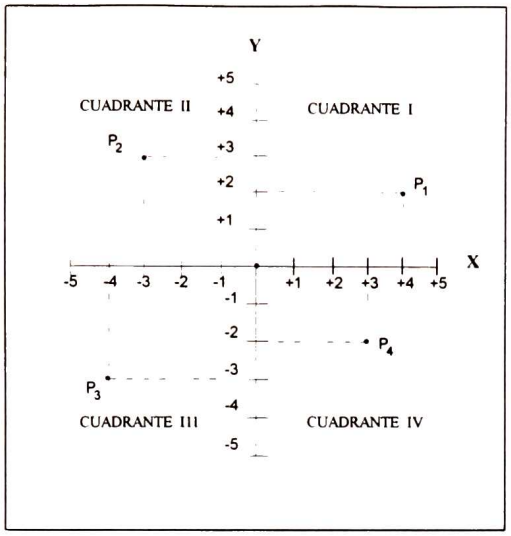


Figura 1.5.- Ejes de coordenadas

Gráficas de curvas simples

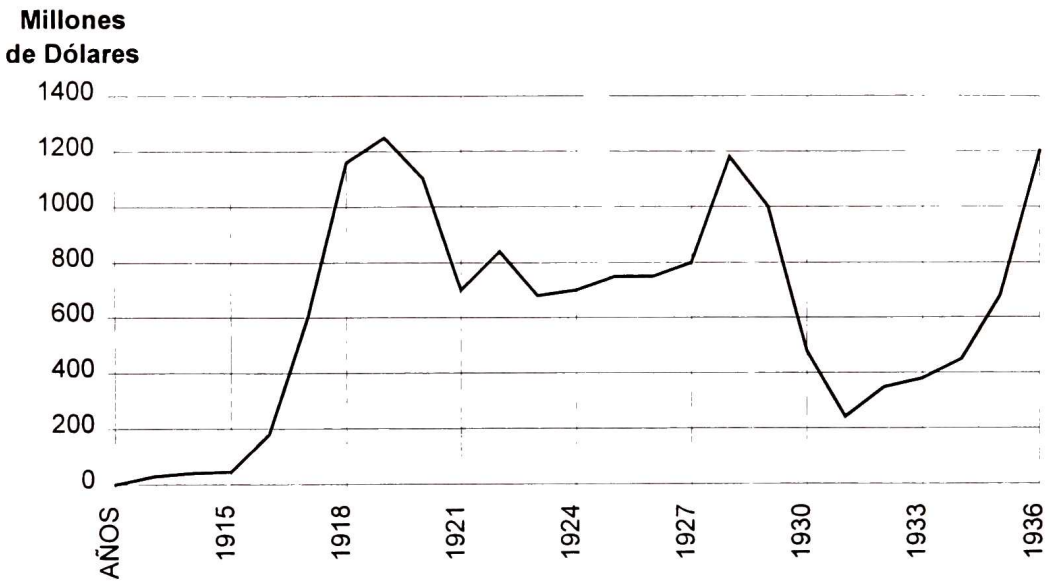
Las curvas se usan a menudo para representar series cronológicas, así como las distribuciones de frecuencia y fluctuaciones.

Curvas de series cronológicas.- El método para trazar series cronológicas depende de la clase de datos que se van a representar. Se debe distinguir entre los datos acumulativos que corresponden a un período y los datos que corresponden a un momento dado.

Los datos de período se refieren a un período de tiempo. Los datos de momento se refieren a un instante determinado del tiempo como son los valores de inventario, cotizaciones de precio o lecturas de temperatura.

Con respecto al tamaño de la gráfica debe cuidarse de no prolongar o contraer demasiado cualquier escala de la curva.

En la figura 1.6 se muestra un ejemplo de gráfica de curva simple.



Impuestos federales sobre la renta individual en Estados unidos.

Figura 1.6.- Gráfica de Curva Simple

Scatter Plot

Este tipo de gráfico puede ser considerado como la herramienta estadística más poderosa para analizar la relación entre dos variables, *x* y *y*. Suponiendo *xi* y *yi* para *y= 1 a n.*

Este tipo de gráfico puede ser considerado como la herramienta estadística más poderosa para analizar la relación entre dos variables *x* y *y*. Suponiendo *xi y* y*i* para *i*=1 a *n.*

Son una herramienta poderosa para ayudarnos a entender la relación entre dos medidas o variables observadas *x* y *y*. Cuando *x* es un factor y *y* una respuesta, el scater plot puede demostrarnos cómo la distribución empírica de *y* depende de *x*.

Gráfica Semilogarítmica o de razones

Muchas veces, al estudiar el desarrollo de una serie de datos estadísticos durante un período, se desea saber el crecimiento que ha tenido lugar pero se interesa uno más por saber algo acerca del crecimiento *relativo* o *tasa de variación.*

Las gráficas lineales más comúnmente conocidas, poseen escalas aritméticas y se usan principalmente para mostrar las variaciones absolutas en el factor que aparece en el eje de las Y. En las gráficas semilogarítmicas, el rayado de los ejes es diferente y hace posible observar la variación relativa que tiene lugar en una serie representada gráficamente.

Una serie que muestra una razón constante de aumento o disminución se conoce con el nombre de *progresión geométrica* y cualquier progresión geométrica da una línea curva cuando se traza sobre un rayado aritmético. Un aumento geométrico está representado por una curva que se mueve hacia arriba y es cóncava también hacia arriba; una disminución geométrica se representa con una curva que se mueve hacia abajo y es cóncava hacia arriba. Sin embargo existe una seria dificultad al interpretar tales curvas debido al hecho de que el ojo no puede percibir si una línea curva determinada representa o no una tasa constante de variación.

En el ejemplo de la gráfica 1.7 no es posible determinar qué país tuvo mayor aumento relativo. Es difícil ver las variaciones en el registro de Canadá porque la escala que debe usarse para los datos de Estados Unidos hace que la curva para Canadá baje hasta la línea base. El hecho de que una curva esté abajo de otra en una cuadrícula aritmética dice que, con solo mirarla que la curva inferior representa una serie de menor magnitud que la curva superior. Si se usan dos escalas verticales, se tienen en realidad dos gráficas distintas no comparables y no pueden hacerse comparaciones visuales satisfactorias con respecto a: 1) la magnitud de las dos series trazadas, 2) la variación que ha tenido lugar en una serie en comparación con la variación de la otra, y 3) las tasas de variación de las dos series.

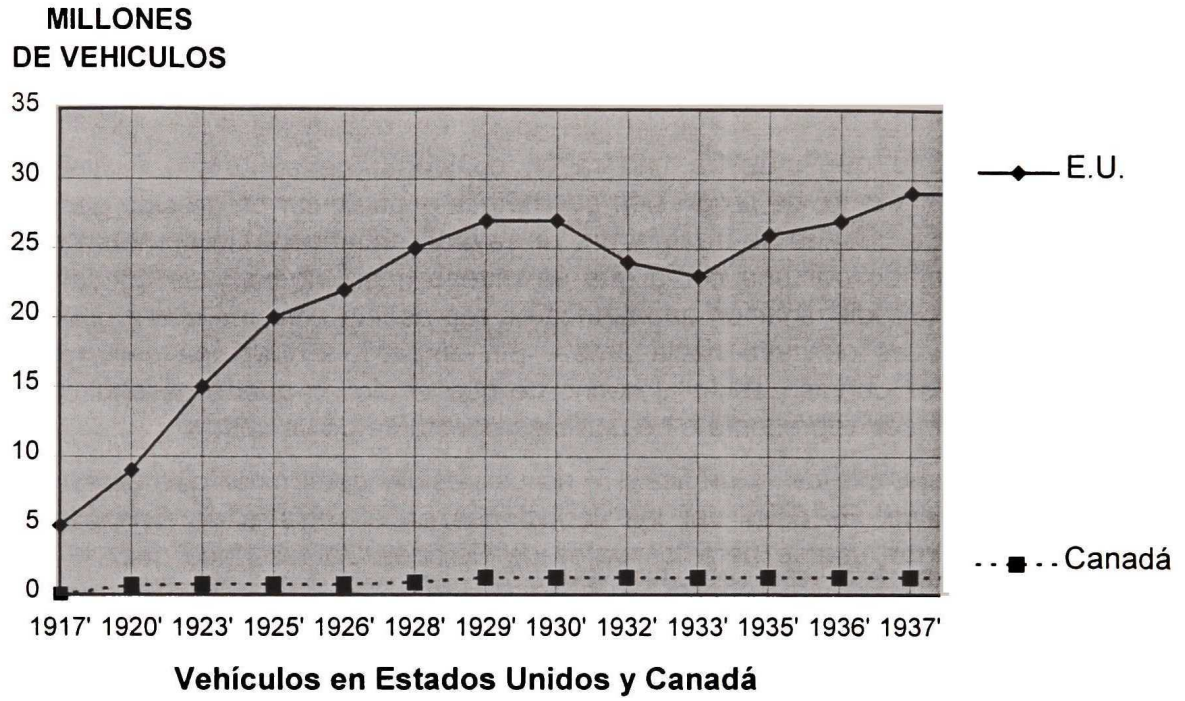


Figura 1.7.- Gráfica Lineal

Con lo anteriormente expuesto es evidente que se facilitarán las comparaciones gráficas con respecto a tipos de variación si se puede hacer uso de un rayado especial que haga aparecer un porciento constante de variación como una línea recta. El rayado que se requiere se denomina *semilogaritmico* porque una escala es logarítmica y la otra es aritmética. Este rayado también se denomina *rayado de razones.*

Construcción del rayado semilogarítmico .- La construcción de la escala logarítmica implica simplemente espaciar los valores de la escala vertical proporcionalmente a las diferencias entre sus logaritmos. Calculando la proporción con respecto a los valores logarítmicos de los primeros 10 números naturales, podemos observar en la siguiente gráfica que la distancia de 2 a 3 es en la escala de 0.352 pulgada y de 3 a 4 es de 0.250 de pulgada. Se debe hacer un mapeo para obtener los espaciamientos entre los "ticks".

Se debe recordar que en un rayado aritmético, las distancias iguales en la escala vertical representan cantidades iguales. Pero en una escala logarítmica, distancias iguales representan porcentajes iguales de variación o razones.

Escala logarítmica .- Observando la figura 1.8 se puede observar que la escala vertical está dividida en dos partes a las que generalmente se designa con el nombre de *ciclos* o *fases* .

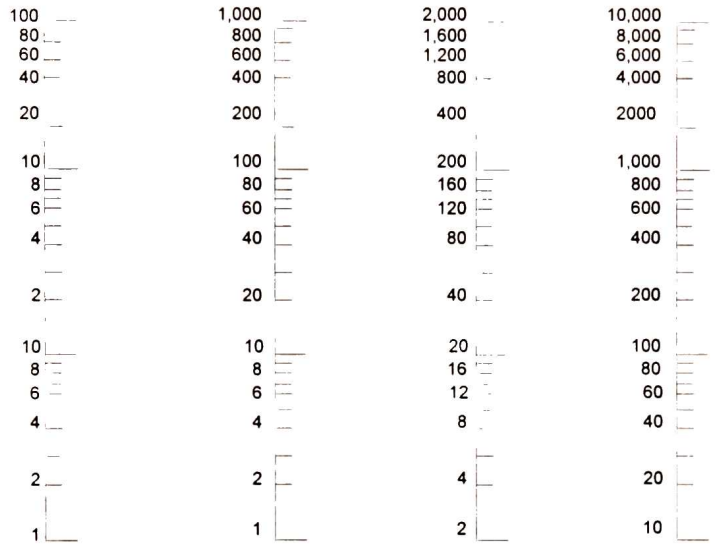


Figura 1.8.- Escalas Semilogarítmicas

Utilizando un rayado semilogarítmico podemos representar los datos de la gráfica anterior de la siguiente manera: (figura 1.9)

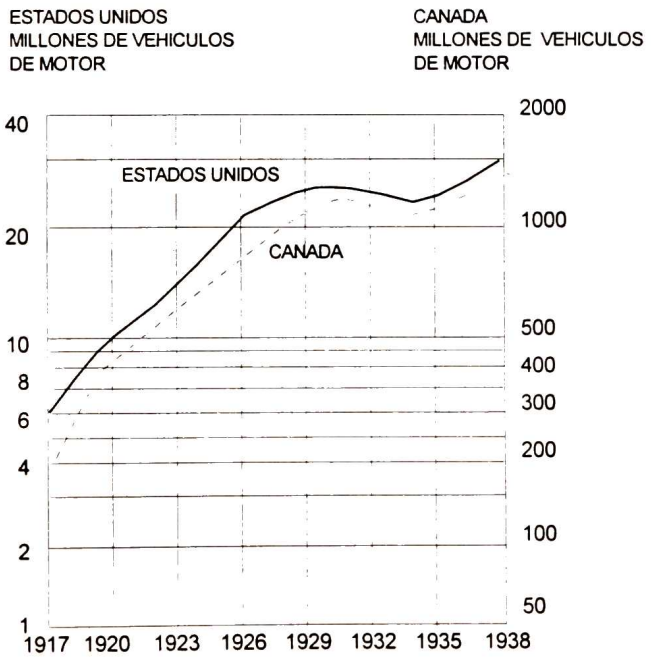


Figura 1.9.- Gráfica Semilogarítmica

En la figura 1.9 podemos observar la variación proporcional de la producción de vehículos en Estados Unidos y la variación con respecto a Canadá, ahora sí es posible hacer comparaciones con respecto a la proporción.

GRAFICAS BASADAS EN PROPORCIONES

Las partes de un total pueden mostrarse por medio de una barra o como un diagrama circular. La gráfica de barras implica una comparación unidimensional de las longitudes de las secciones de la barra; mientras que el diagrama circular implica la comparación bidimensional de los sectores del círculo o la comparación unidimensional de los arcos del círculo.

Gráficas de barra y columna

Las gráficas de barra y gráficas de columna generalmente se conocen con el mismo nombre y se usan de manera indistinta, pero cabe señalar que ambas gráficas tienen características diferentes que se citan a continuación:

Generalmente en el primer tipo de gráfica, las barras se colocan horizontalmente; en una gráfica de columna, se colocan verticalmente. La gráfica de barra se utiliza principalmente para comparaciones directas de magnitud para categorías etiquetadas descriptivamente mientras que la gráfica de columna se utiliza para comparaciones de una o más series de variables sobre el tiempo. Esto implica que la gráfica de barra sólo tenga una escala y la gráfica de columna posea dos escalas como la gráfica rectilínea de coordenadas.

El propósito más importante de ambas gráficas es mostrar comparaciones de cantidades utilizando un objeto gráfico simple y sencillo de comprender como son las barras o columnas unidimensionales.

A continuación se presenta una clasificación sobre algunos tipos de gráficas de barra:

Gráfica de Barra Simple .- Es una de las más útiles y más comúnmente usadas, se basa en valores lineares directos, la longitud de la barra está determinado por el valor o cantidad de cada categoría como puede observarse en la figura 1.10.

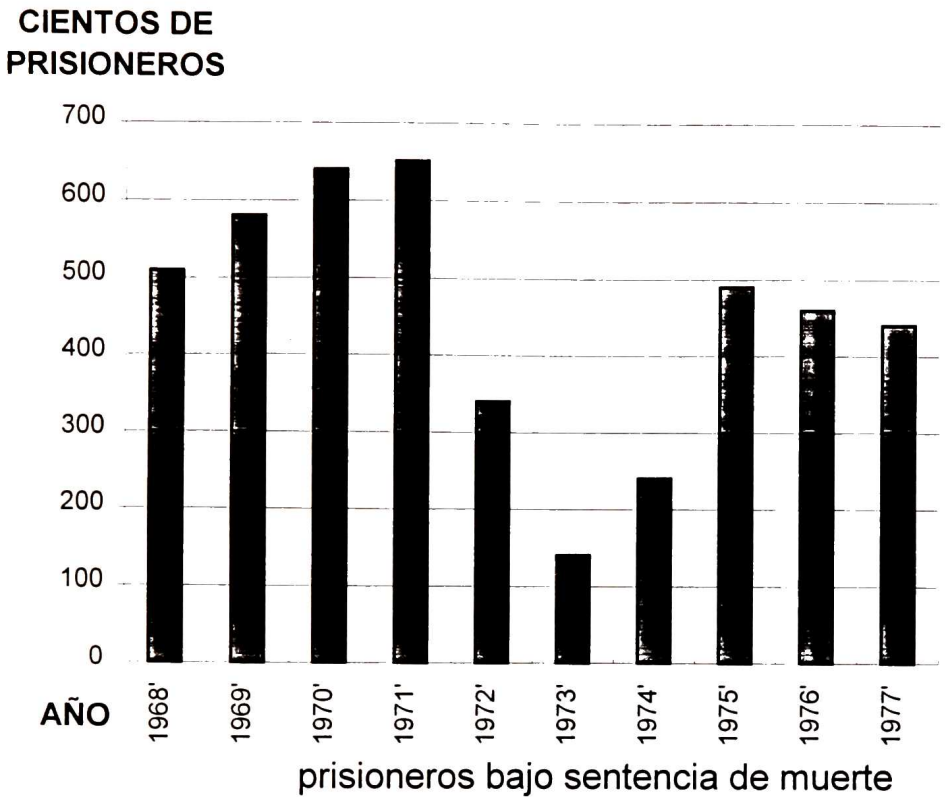


Figura 1.10.- Gráfica de Barras Simples

Gráfica de Barra Subdividida .- La gráfica de barra subdividida, se usa para mostrar una serie de valores con respecto a sus partes componentes. La escala puede ser expresada en términos de valores absolutos o relativos.

Las características básicas con las que debe contar este tipo de gráficas son principalmente las siguientes:

a) Las columnas no deben ser excesivamente grandes o pequeñas ni excesivamente anchas o angostas. El espacio entre ellas debe ser de por lo menos un cuarto del ancho de las mismas, dependiendo del espacio y del ancho de las columnas, algunas pueden llegar a tocarse.

b) La escala vertical de la típica gráfica de barras es semejante a la gráfica de coordenadas rectilíneas. Tiene siempre un cero en la línea base. La línea vertical no debe de romperse o cortarse.

Todas estas características han sido tomadas en cuenta para la generación de gráficas del sistema.

Diagrama de Pastel

Un diagrama de pastel no tiene ejes externos. Las cantidades se mapean sobre segmentos de un círculo. El tamaño (incluido el ángulo) de cada segmento corresponde al valor de la cantidad que le ha sido asignada como una proporción del valor total de todas las cantidades desplegadas en el pastel. Este tipo de representación, es útil en el despliegue de proporciones. Los segmentos en el diagrama de pastel pueden etiquetarse con los porcentajes correspondientes por lo que las estimaciones precisas del tamaño de cada segmento no tienen que hacerse visualmente.

Los diagramas de pastel son un medio gráfico conveniente para enfatizar las proporciones de una variable que está asignada a cada una de las categorías de otra variable nominal.

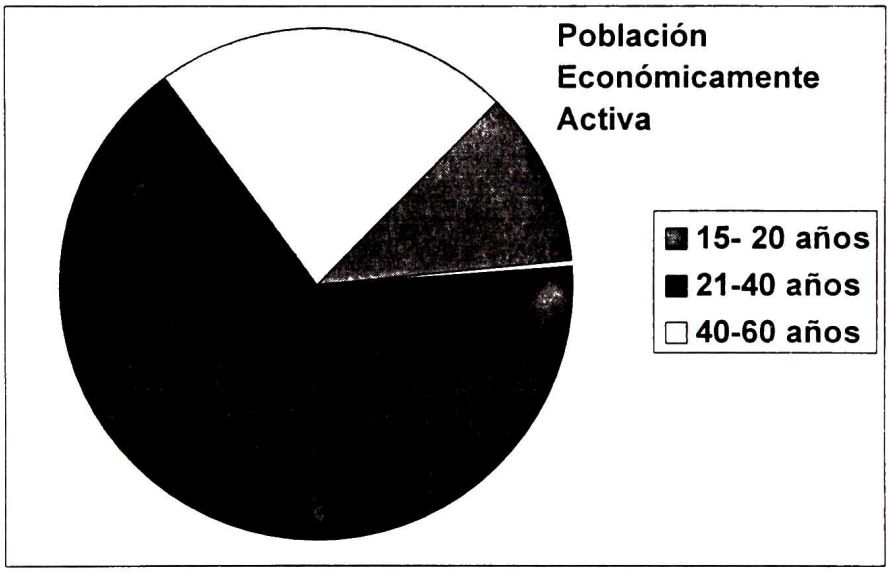


Figura 1.11.- Gráfica de pastel

1.5 CONCLUSIÓN DEL CAPÍTULO

A través de este capítulo se conoce el concepto de visualización de datos, así mismo se tiene una idea sobre los diferentes métodos que existen para la representación de datos y la diferencia entre la representación de datos a través de tablas y a través de gráficos.

De este capítulo se puede concluir que la visualización proporciona al humano una mayor facilidad para comprender e interpretar la información contenida en un conjunto de datos; le permite observar los cambios que en ésta se generan a través del tiempo de una manera mas clara e inmediata; le permite realizar comparaciones más fácilmente. Entre otras cosas puede decirse que la visualización de datos ayuda en el análisis de información para la toma de decisiones

Capítulo 2

AGRUPAMIENTOS Y TRATAMIENTOS CON VARIAS VARIABLES

Otro tipo de gráficas importantes son las que pueden representar objetos con múltiples variables. Su construcción requiere de algunas técnicas que puedan agrupar las diversas variables y las gráficas contengan la funcionalidad y las propiedades visuales apropiadas. Este capítulo consiste en tratar algunas de las técnicas existentes, que resultarán de importancia en la exploración de una base de datos.

El sistema DALI tiene por objetivo poder visualizar información de varias variables. Las bases de datos relacionales presentan resúmenes de una gran cantidad de atributos (variables), los cuales deben ser sintetizados para una rápida comprensión de la situación que se tiene en un momento dado en la base de datos. DALI se enfoca principalmente en las caras de Chernoff para la representación de varias variables.

Los métodos icónicos para el almacenamiento y despliegue de datos multivariados han estado en uso desde hace siglos, desde los Quipu peruanos que eran una forma de desplegar datos utilizando cuerdas anudadas de diferente color, tamaño, número y posición, y que se utilizaban para registrar una amplia variedad de datos estadísticos, hasta las figuras de Chernoff [10] y sus subsecuentes modificaciones [9]. A pesar de la aparente variedad, el estudio de la historia de despliegues multivariados revelan más similaridades que diferencias en los aspectos de características de despliegue utilizados. Herman Chernoff [10] realizó una gran contribución para esta metodología cuando enfatizó a través de su famoso diagrama de cara la utilidad de un ícono el cual es representativo en su totalidad.

Las caritas de Chernoff han sido la metología visual más representativa en varias variables, su forma concisa, de gran significado y amplias posibilidades, han originado un uso extenso y en particular la importancia en este trabajo. Muchas de las técnicas desarrolladas para la búsqueda de datos multivariados se derivan de la misma noción básica; un ícono con muchas partes se usa para representar un punto multivariado, el tamaño o forma de cada parte del ícono representa un componenente de los datos. A continuación se describen algunas técnicas para la representación de datos multivariados.

2.1 MÉTODOS DE TRATAMIENTO CON VARIABLES COMPUESTAS

2.1.1 Gráficas de función

En 1972 Andrews [9] desarrolló un esquema para despliegues multivariados. Su método involucra el cálculo de una función periódica de componentes k Fourier; para cada punto k-dimensional, el valor de cada componente Fourier de esa función, está determinada por el valor de su variable asociada. El poderío de esta técnica es que permite la inclusión de muchas variables pero limita el número de datos que efectivamente pueden ser mostrados; con más de 10 o 20 puntos la gráfica puede dejar de ser útil. Un ejemplo de esta curva se presenta en la figura 2.1.

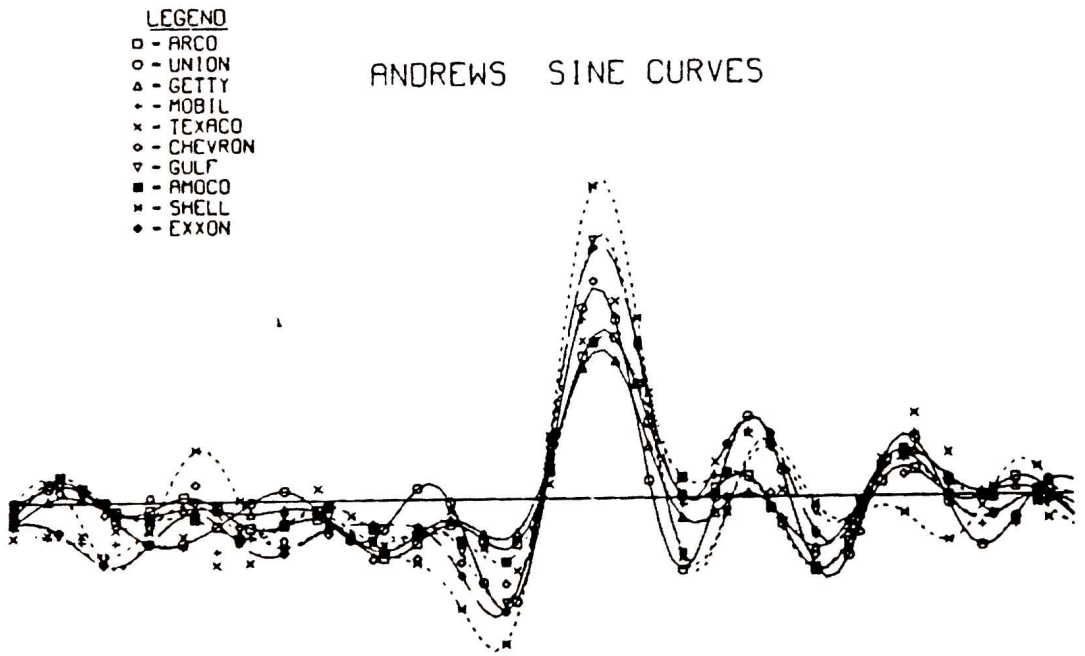


Figura 2.1.- Curva de Andrews

2.1.2 Polígonos

Un ícono más tradicional para la representación de datos multivariados es un polígono formado por la k-ésima segmentación de un círculo con un radio k asociado con una variable particular. La longitud de cada radio es proporcional al valor de la variable para el punto del dato que será representado y estos puntos se conectan para formar un polígono irregular. Esta técnica tiene más de cien años de utilización y proporciona un útil despliegue de datos en varias variables. Generalmente, cada variable se escala separadamente, por lo que el valor máximo de cada variable llega a tocar la circunferencia [9].

Las desventajas de este tipo de gráfica son las siguientes:

- Se recomienda no utilizar más de siete variables ya que la legibilidad de la gráfica se hace más difícil.
- La figura de los polígonos es arbitraria debido a la arbitrariedad del orden de las variables alrededor del círculo.
- El uso de leyendas para identificar las variables es crucial para el despliegue. Puesto que es difícil retener en la mente qué esquina se refiere a qué variable.

Las similitudes en la figura pueden causar apreciaciones equivocadas como en el siguiente ejemplo se indica:

Podemos observar que los estados de Vermont y Oregon en Estados Unidos son similares excepto por la rotación de los polígonos. Por supuesto esta similaridad es falsa debido al ordenamiento particular de las variables.

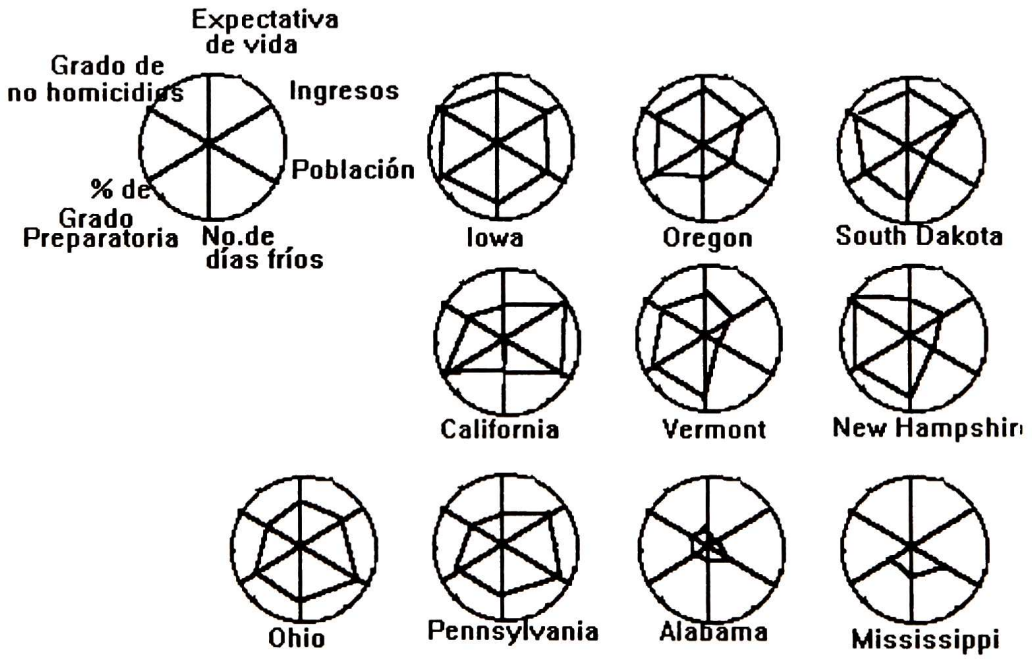


Figura 2.2.- Polígonos

2.1.3 Árboles

En los métodos de despliegue discutidos anteriormente el observador es más o menos capaz de agrupar visualmente las observaciones multivariadas a través de la similaridad de sus íconos representativos. La agrupación de las variables no se hace fácilmente. Esta imposibilidad para representar simultáneamente la agrupación de variables y de observaciones se ve como una debilidad del despliegue. Kleiner y Hartigan [9] resolvieron este problema de una manera simple e ingeniosa. Ellos notaron que un árbol es un ícono común para representar una estructura de agrupación jerárquico, y que la estructura del árbol está razonablemente bien determinada [9]. Ellos razonaron que se podría usar una variación de la estructura del árbol obtenida desde las distancias entre observaciones como un ícono básico. El tamaño de cada rama puede determinarse por el valor de las variables representadas por esta rama. Así a cada observación sería representada por un árbol con la misma estructura general pero con diferente tamaño de ramas.[9]

En la figura 2.3 observamos un árbol derivado del ejemplo anterior.

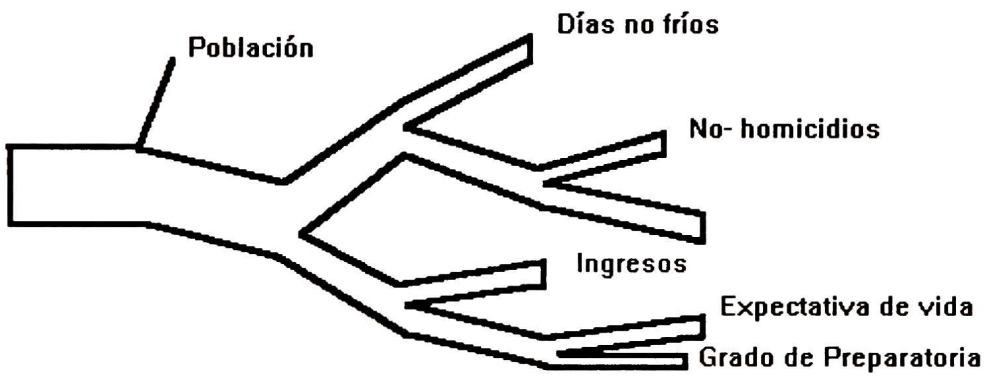


Figura 2.3.- Árbol

2.2 CARAS DE CHERNOFF

En 1971 Herman Chernoff [10], auspiciado por la Oficina de Investigación Naval y la Universidad de Stanford, introdujo la idea del uso de caras para la representación de datos multidimensionales (denominados también multivariados) desde entonces esta técnica ha sido utilizada en una gran variedad de aplicaciones, principalmente para representar información, descubrir agrupamientos y para mostrar los cambios ocurridos en los datos a través del tiempo.

Chernoff consideró que los datos podrían tener 18 variables o características como máximo y cada una de estas características puede representarse sobre una característica facial. En la figuras 2.4 y 2.5 se pueden apreciar cada una de éstas características, en la figuras 2.6 y 2.7 se aprecia una característica más añadida por Herbert T. Davis, Jr. [10] quien agregó a la cara original la nariz y orejas. En la siguiente tabla se describen cada una de estas características:

Variable		Característica facial	Valor Default	Rango	
X_1	controla h*	Ancho cara	0.60	0.20	0.70
X_2	controla q	Nivel de oreja	0.50	0.35	0.65
X_3	controla h	Longitud de	0.50	0.50	1.00
X_4	es	Excentricidad de la elipse	0.50	0.50	1.00
X_5	es	Excentricidad de la elipse	1.00	0.50	1.00
X_6	controla	Longitud de la nariz	0.25	0.15	0.40
X_7	controla P_m	Posición del centro de la boca	0.50	0.20	0.40
X8	controla	Curvatura de la boca	0.00	-4.00	4.00
X9	controla	La longitud de la boca	0.50	0.30	1.00
X10	controla Ye	Anchura del centro de los ojos	0.10	0.00	0.30
X11	controla Xe	Separación de los ojos	0.70	0.30	0.80
X12	controla θ	Inclinación de los ojos	0.50	0.20	0.60
X13	es	La excentricidad de los ojos	0.60	0.40	0.80
X14	controla L_e	Longitud media del ojo	0.50	0.20	1.00
X15	controla	La posición de las pupilas	0.50	0.20	0.80
X16	controla Yb	Anchura de cejas	0.80	0.60	1.00
X17	controla θ**	Angulo de la ceja	0.50	0.00	1.00
X18	controla	Longitud de la ceja	0.50	0.30	1.00
X19	controla r	Radio de la oreja	0.50	0.10	1.00
X20	controla	Ancho de la nariz	0.10	0.10	0.20

Tabla 2.1 Características faciales de las caras de Chernoff

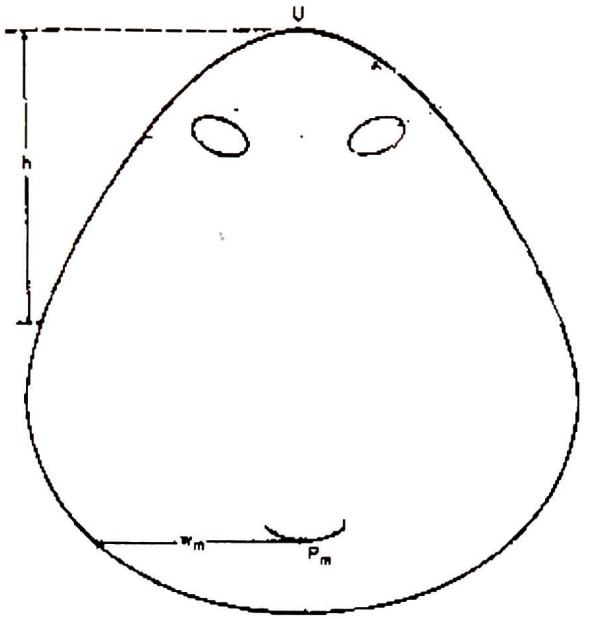


Figura 2.4.- Cara de Chernoff con 4 características

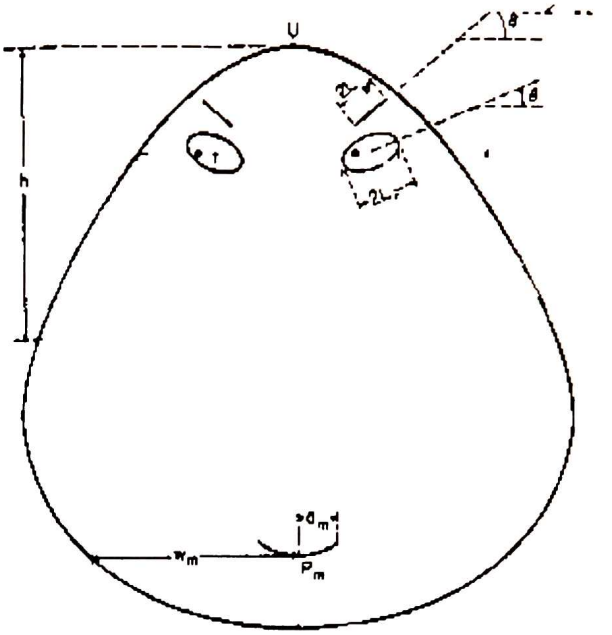


Figura 2.5.- Cara de Chernoff con 8 características

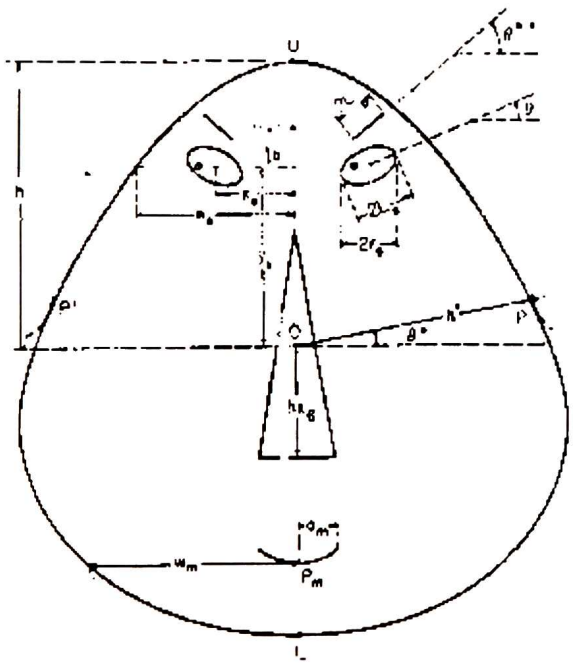


Figura 2.6.- Cara de Chernoff con 16 características

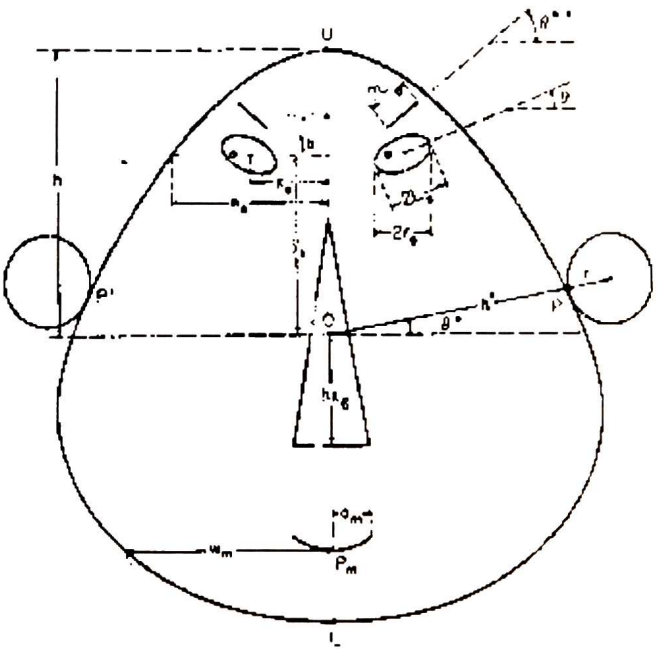


Figura 2.7.-Cara de Chernoff con 20 características

Muchas de las características faciales están controladas por los datos asociados a una característica o a los datos asociados con otras características. Por ejemplo la anchura real de la boca es una función no solamente de h^* sino también de θ^*; la longitud de la boca depende de a_m y también de w_m.

Los rangos de los rasgos faciales se han ajustado de manera que las caras parezcan más humanas con el fin de que los rasgos tengan un significado y su observación sea lo más natural posible. Puede ser que el uso de rasgos parecidos a los humanos contribuya a la interpretación de un conjunto de datos pero no de otro.

2.2.1 Construcción de las caras

Primero que nada es necesario asignar los valores de cada una de las variables a las características faciales. Consideramos variables a todos aquellos datos que representan un concepto global desde el punto de vista de base de datos, pueden considerarse como el conjunto de atributos de una entidad. Esta asignación puede hacerse de manera deliberada o al azar.

El sistema DALI realiza esta asignación relacionando las características en el orden en que se encuentran en el descriptor de archivos y el orden en que se encuentran en la tabla 2.1, como extensión al sistema puede existir la posibilidad de que el usuario proponga qué características de los datos se van a asignar a qué rasgos faciales.

Una vez que se ha realizado la correspondencia entre variables y rasgos de la cara, se determinan los valores más altos y más bajos de cada variable de datos (de cada característica) para realizar un mapeo con respecto a los valores de los datos y los valores del rango indicados en la tabla 2.1.

Ya que se han mapeado los valores de los datos con respecto a los rasgos faciales se ejecuta el algoritmo propuesto por Chernoff para la generación de las caras.

A continuación se presenta un ejemplo en donde se utilizan caras de Chernoff para representar la situación de algunas empresas petroleras de Estados Unidos de acuerdo a las variables más representativas de la empresa. En la tabla 2.2 se describen las variables que se van a representar y en la tabla 2.3 se listan los valores de esas variables.

Tabla 2.2 Descripción de Variables

Número	Característica	Descripción
1	Ganancia Neta	Total de dólares pagados (billones).
2	Excedente	Promedio bruto de dólares pagados
3	Acres netos	Total de acres netos contratados
4	Número de contratos ganados	Número de contratos ganados
5	Propiedades en promedio	Porcentaje promedio de propiedades en acres
6	Pct. Prod. De contratos ganados	Porcentaje de contratos que últimamente tuvieron producción ganados por la compañía
7	Promedio de años de producción	Promedio del número de años entre la venta y la primera producción (demora de producción)
8	Prod. De Gas Neta	Producción de Gas neta (en trillones de metros cúbicos
9	Prod. Líquida Neta	Producción líquida neta (en millones de barriles)
10	Derechos Netos	Derechos netos pagados al gobierno (en millones)
11	Derechos /Ganancias	Derechos netos / número de años de producción.
13	R**2:Roy/Pyr	Cuadrado del coeficiente de correlación de regresión lineal múltiple / producción anual, demora de producción y años de producción (solo para producción de contratos)
14	Roy/Pyr/Pr.Ac	Derechos/año de producción/producción de acre (en dólares)
15	Pct. Contratos Terminados	Porcentaje de contratos propios terminados.

Tabla 2.3 Datos correspondientes a las 15 variables

Nombre	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Arco	.56	1.1	.78	306	49	10	4.5	.38	66	62	.11	174	.84	2.8	35
Union	.53	1.2	.49	203	47	4	4.2	1.22	103	99	.19	527	.98	8.5	38
Getty	.54	1.0	.32	197	31	11	4.0	.67	51	57	.11	160	.38	2.8	26
Movil	1.21	2.8	.50	211	50	8	3.9	1.04	68	78	.06	339	.81	4.0	25
Texaco	1.16	2.7	.56	176	66	8	7.8	.31	56	50	.04	277	.91	2.5	34
Chevron	.84	1.2	1.16	378	70	13	5.8	.70	197	141	.17	355	.50	1.6	32
Gulf	1.01	2.2	.67	219	65	11	4.1	1.53	338	235	.23	481	.93	2.9	37
Amoco	.66	1.3	.66	258	53	8	7.3	.45	37	44	.07	213	.31	2.7	30
Shell	.97	1.7	1.59	336	95	13	3.6	1.90	430	378	.39	656	.38	2.7	54
Exxon	1.44	2.9	1.02	250	84	8	5.2	.99	276	199	.14	609	.58	4.3	36

Utilizando los valores de la tabla 2.3 y realizando el mapeo correspondiente, las gráficas generadas se presentan en la siguiente figura.

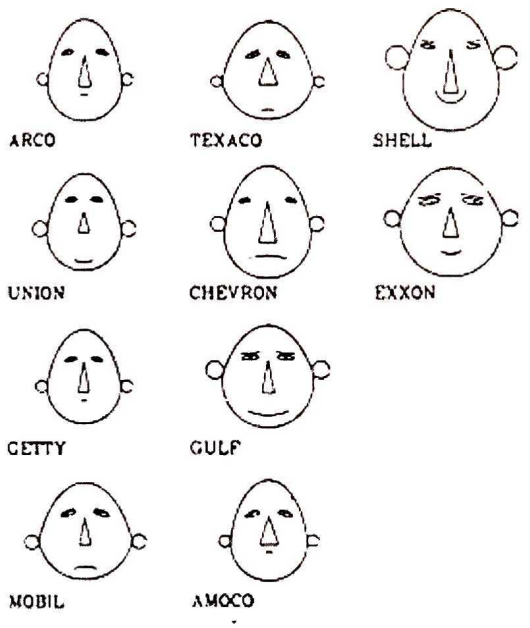


Figura 2.8.- Gráficas de Chernoff representativas de las compañías petroleras.

Como puede observarse, es mucho mas fácil visualizar el conjunto de variables de las compañías petroleras a través de las gráficas multivariadas de Chernoff que a través de tablas de datos.

2.2.2 Ventajas y desventajas del uso de caras de Chernoff

Como toda representación visual y en particular icónica, las caras de Chernoff tienen sus ventajas y desventajas. Ambas están relacionadas con su alto valor semántico asociado a su poder de síntesis.

- Las caras por su naturaleza misma son íconos que son más fáciles de reconocer y describir.
- Las características faciales se pueden asociar con el significado físico de las variables.
- Algún gesto facial o rasgo representativo puede ser útil para representar el valor de una variable. La expresión de la boca puede representar alguna variable con un alto valor significativo que sea determinante para un conjunto de características.

- La interpretación de las caras llega a ser subjetiva. La subjetividad radica en la forma en la que se van a distribuir las variables sobre cada una de las características faciales. Lo que para algunos resulte más representativo mapear sobre la boca, para otros puede resultar mejor mapear sobre los ojos o sobre el tamaño de la nariz.
- Las caras pueden agrupar patrones de información. Existe una asociación entre patrones de forma y patrones simbólicos.
- Es posible concentrar un conjunto de las variables de datos sin rehacer las gráficas. El observador se puede concentrar en las variables asociadas con los ojos y orejas y después concentrarse en las variables asociadas con las orejas y boca.
- 15 es el valor máximo de variables aceptables. Un número más grande de variables produce confusión en la interpretación.
- Las dependencias entre las características faciales llega a distorsionar lo suficiente la representación de datos que ocasiona impresiones erróneas.

2.2.3 Dependencia de las características faciales

En el uso de las Caras de Chernoff para la representación de datos existe un problema potencialmente serio relativo a la dependencia de las características faciales. La dependencia relacionada con la interrelación entre variables y rasgos faciales nos da dos clases.

Primera clase: Cuando algunas de las características faciales dependen solamente de los datos de entrada.

Segunda Clase: Cuando algunas características están relacionadas entre si y dependen unas de otras como por ejemplo, la estructura de la boca depende del largo y ancho de la cara, el nivel de la oreja de la excentricidad inferior de la cara y de la longitud de la nariz así como también de otros tres parámetros de la boca, estas dependencias ocurren con el fin de asegurar el posicionamiento adecuado de los rasgos de la cara.

2.3 CONCLUSIONES DEL CAPÍTULO

La primer conclusión que podemos dar a este capítulo consiste en la importancia misma que tiene la visualización por medio de gráficas al agrupamiento y tratamiento de varias variables. La naturaleza de las bases de datos relacionales presentada en tablas con un cierto número de atributos, presenta el problema de representación de la información. Las caras de Chernoff ofrecen una excelente herramienta visual para representar datos multidimensionales de tablas sumarizadas.

Las técnicas de reconocimientop de patrones son de gran importancia para resolver problemas de minería de datos. En este contexto, el sistema puede ser entendido a un tratamiento como lenguaje visual y no sólo como ambiente visual.

El presente trabajo está abierto para que le sean añadidos otros tipos de representación gráfica como son gráficas en 3D o extensiones de las caras de Chernoff como la versión adaptada, modificada y mejorada desarrollada por Flury y Riedwyl [9] o la versión desarrollada por Wakimoto [9] quien utilizó un diagrama de cuerpo completo para representar varias medidas fisiológicas como se muestra en la figura 2.9.

El color es un atributo que hasta la fecha no ha sido utilizado, éste puede llegar a tener un significado relevante en la visualización, añadiendo y/o sustituyendo algunas dimensiones. El espectro del color puede determinar un intervalo armónico de valores

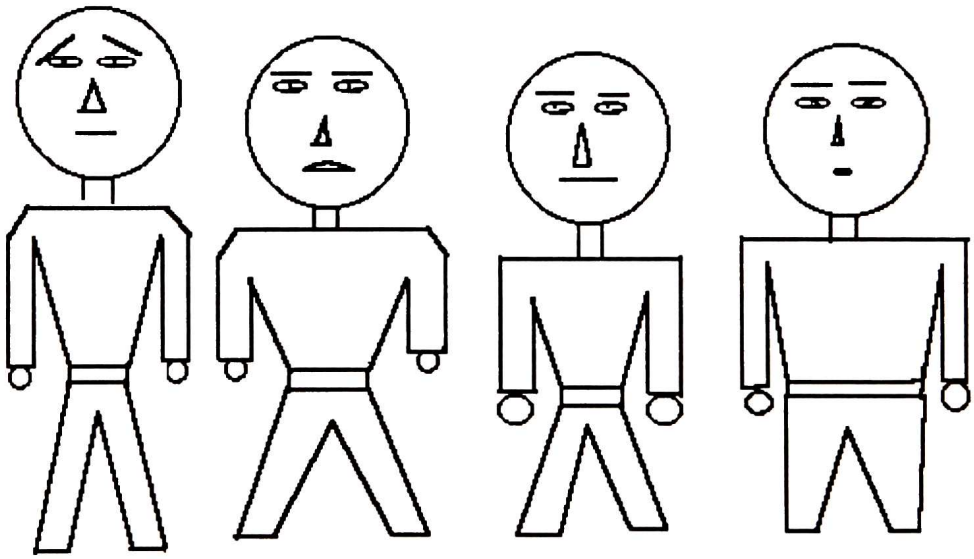


Figura 2.9.- Diagramas de Wakimoto

CAPITULO 3

MÉTODO DE CONSTRUCCIÓN GRÁFICA

3.1 OBJETIVO DE DALI COMO HERRAMIENTA DE GRAFICACIÓN

El sistema DALI sirve como herramienta en el ordenamiento y clasificación de datos y en su interpretación y comunicación, analizando la información contenida en una base de datos relacional y generando su representación gráfica.

En varios sistemas, la información que se extrae de la Base de Datos se presenta a través de tablas, los cuales, muchas veces no facilitan al usuario la observación de las variaciones de la información con el paso del tiempo.

El principal objetivo de DALI es proporcionar al usuario la representación visual de la información contenida en una base de datos relacional que permita al usuario observar fácilmente las variaciones o cambios en los datos y poder lograr una mejor toma de decisiones. Para ello, el sistema realiza un análisis previo de la información para clasificar los datos.

Este capítulo describe detalladamente la metodología del análisis de información que se menciona en la sección anterior y que se implementó en el sistema DALI.

En este capítulo se describe el método de análisis sobre el cual se basa DALI para la representación gráfica de la información. Este método es propuesto por Jaques Bertin [11] y consta de dos etapas fundamentales:

- Análisis de la Representación Gráfica
- Construcción Gráfica

Cada una de estas etapas está compuesta de elementos particulares que han sido implementados en el sistema DALI y que se describen a continuación.

3.2 ANÁLISIS DE LA REPRESENTACIÓN GRÁFICA

Una gráfica no es solamente un dibujo, es una representación de información que resulta muy importante en la toma de decisiones debido al conocimiento que conlleva ésta. Una gráfica no se dibuja una sola vez, sino que se construye y se reconstruye hasta que revela todas las relaciones constituidas por la interacción de los datos.

En cualquier lugar en donde existe una base de datos, se extrae información, ya sea para conocer totales, gastos mensuales, utilidades registradas por la empresa, etc. todo esto conlleva a tomar ciertas decisiones dependiendo de la situación que presenten los datos.

El método de construcción gráfica que aquí se maneja, parte del hecho de que a través de la representación gráfica de un problema, la toma de decisiones es mucho más fácil, ahora bien, sabemos que para tomar una decisión, se requiere el seguimiento de los siguientes puntos:

a) Definir cual es el problema que se quiere resolver

b) Ordenar y clasificar los datos

c) Procesar los datos

d) Interpretar, decidir o comunicar los datos simplificados

Para construir una gráfica útil, debemos conocer que es lo que se requiere previamente para su construcción. Jaques Bertin [11] propone las siguientes *Formas Sucesivas de la Aplicación Gráfica*, las cuales están relacionadas con las etapas en la toma de decisiones.

ETAPAS EN LA TOMA DE DECISIONES	**APLICACION GRAFICA**
Defininir el problema Construir la tabla de datos	Realizar el *Análisis de Matriz* del problema
Procesar los datos	Procesamiento de Información Gráfica
Interpretar, decidir o comunicar los datos simplificados	Comunicación Gráfica

A continuación se describen las etapas para la construcción gráfica. Estas etapas se implementaron en el Sistema DALI para que éste sirva como herramienta en la visualización de información y posteriormente en la toma de decisiones.

El problema que se quiere resolver con la generación de gráficas, es observar el comportamiento de los datos a través de una herramienta visual que facilite al usuario la interpretación de la información.

El *Análisis de Matriz*, es un proceso el cual nos permite ver el problema de manera global, para construirlo gráficamente y preveer las posibles repercusiones que pudieran ocurrir, nos permitirá clasificar los datos, verificar si son homogéneos o heterogéneos, es decir, si los datos tienen algún punto en común o no existe ninguna relación entre ellos. A través del Análisis de Matriz se define al **objeto** y sus **características**.

El *Procesamiento de Información Gráfica* consiste en la elección de una construcción gráfica dependiendo del tipo de objeto y del número de características relacionadas a él.

La *Comunicación Gráfica*, involucra decir o transcribir a otros lo que se ha descubierto, y si es posible contestar las preguntas que se hicieron al principio.

De acuerdo a los puntos anteriores, el Sistema DALI realiza el *Análisis de Matriz de Datos* extrayendo y clasificando los datos sobre los cuales se ha realizado una operación a través de LIDA.

El *Procesamiento de Información Gráfica* se realiza detectando los datos OBJETO y el número de CARACTERÍSTICAS relacionadas a ese objeto.

La *Comunicación Gráfica,* se lleva a cabo a través de gráficas representativas de los datos durante la ejecución de un flujograma de LIDA.

3.3 ANÁLISIS DE MATRIZ

Bertin [11] hace énfasis en que para resolver un problema y tomar una decisión sobre el mismo es deseable representar la información de una manera gráfica ya que nos permite ver el problema completo con una mayor claridad; en el caso del sistema DALI, el problema que deseamos observar se refiere a los cambios que van sufriendo los datos conforme se realizan operaciones sobre ellos tales como un "SELECT", una "UNION", un "JOIN" con el fin de que los resultados puedan ser observados más rápidamente y con una mejor comprensión.

El análisis de matriz de un problema es un proceso el cual permite ver a dicho problema de una manera global. Es un proceso de reflexión que se basa en la notación gráfica. Esto nos permite, cualquiera que sea la magnitud del problema, concebirlo como una tabla simple.

Este análisis visualiza un conjunto entero de información de tal forma que el investigador pueda definir una tabla de datos compatible con un método de procesamiento capaz de producir una respuesta a las preguntas pertinentes y que sea compatible con los medios disponibles.

En general, el Análisis de Matriz nos permite:

- Definir el punto de comparación y asegurar la homogeneidad del estudio.
- Registrar las preguntas en una forma apropiada a los datos asegurándose de que el método de procesamiento contestará a éstas preguntas.
- Disminuir la ambigüedad de la presentación estadística y desplegar la información considerándola en forma concisa y precisa para los usuarios potenciales (matemáticos, programadores, etc.)

El Análisis de Matriz se divide en dos fases:

- Construcción de la **Tabla de Distribución** o **Tabla de Datos**
- Definición del **Esquema de Homogeneidad**

Estos elementos se describen en las secciones siguientes.

3.3.1 Tabla de Distribución

La ***Tabla de Distribución*** hace un inventario preciso de todos los componentes tomados en consideración así como todas sus intersecciones para observar las relaciones entre los datos.

La Tabla de Distribución:

- Registra todos los componentes del estudio,
- Graba la longitud de cada componente y las intersecciones con otros componentes

Una Tabla de Distribución se construye realizando primero un *Inventario de Componentes* y registrando después las *Intersecciones* que existan entre ellos.

Permite clasificar los elementos que se desean analizar.

A continuación se da un ejemplo de cómo se puede ir construyendo la tabla de distribución:

Suponemos que tenemos los datos de Profesión y Edad de una muestra.

Existen 250 individuos y 10 profesiones diferentes .

a) Si conocemos la profesión de cada individuo, la relación individuo-profesión se representa como se muestra a continuación:

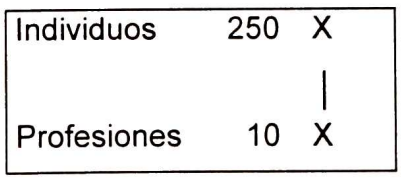


Figura 3.1.- Representación funcional 1 a 1 para Individuos y Profesiones

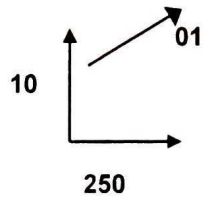


Figura 3.2.- Representación gráfica de la figura 3.1

A cada individuo le corresponde una de las 10 profesiones. La notación **01** significa valores si/no.

b) La edad se encuentra clasificada en cuatro grupos. Si no se conoce la edad exacta de cada individuo, pero se conoce el grupo al que pertenece, agregando este dato a la información anterior, se observa la siguiente relación.

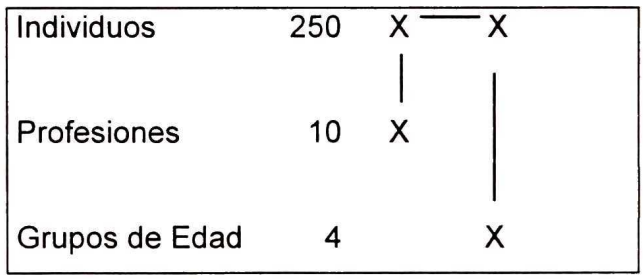


Figura 3.3.- Relación funcional 1 a 1 para Individuos y Grupos de Edad

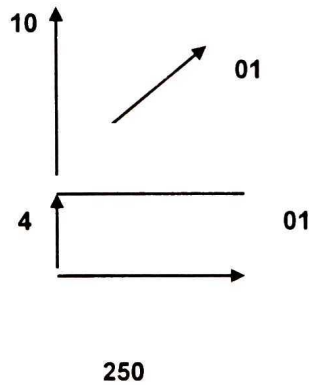


Figura 3.4.- Representación gráfica de la figura 3.3

Dos cruces en la misma fila, definen un punto común a las dos tablas correspondientes. Cada individuo tiene una profesión y una edad dentro de un grupo determinado.

c) Cuando la edad de cada individuo se conoce, se expresa como una cantidad en años. Para representar cantidades se utiliza como notación la letra **Q.**

Individuos	250	X
		|
Edad	1	Q

Figura 3.5.- Relación funcional 1 a 1 de Individuos con el valor de Edad

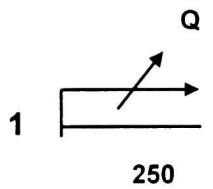


Figura 3.6 Gráfica de la figura 3.5

Si solo se registra una cruz en la columna, la tabla tiene solo una fila. A cada individuo le corresponde un solo valor numérico que representa su edad.

d) Si no se conoce la edad ni la profesión de cada individuo, pero sabemos el número de individuos por cada profesión y por cada grupo de edad, esta cantidad se representa como se muestra a continuación:

Individuos	--	Q
		|
Profesiones	10	X
		|
Grupos de Edades	4	X

Figura 3.7.- Relación funcional Individuo - Profesiones e Individuo - Grupos de Edades

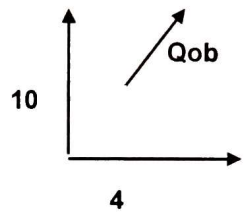


Figura 3.8.- Gráfica de la figura 3.7

La notación **Q**, define un componente cuantitativo expresado en *z* para una tabla definida por las cruces que aparecen en la columna. **Qob** significa **Objetos Estadísticos**.

e) Si nosotros sabemos la profesión de cada individuo pero solo conocemos el número de individuos por grupo de edad, el componente "Individuos" proporciona en el primer caso, una entrada en la tabla y en el segundo caso una cantidad expresada en *z* como se muestra a continuación:

Individuos	**250**	X — Q
	-----	| |
Profesiones	10	X — X
		|
Grupos de edad	4	X

Figura 3.9.- Relación funcional Individuos - Profesiones e Individuos Grupo de Edad

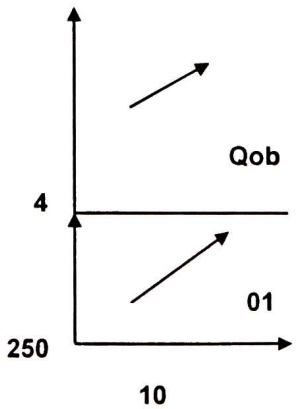


Figura 3.10.- Gráfica de la figura 3.9

La presencia de **X** y **Q** en la misma línea, define un componente que proporciona para **X** un elemento en el eje de la tabla y para **Q** una cantidad expresada en *z*.

Para definir todas las posibles situaciones estadísticas, se agregan dos signos, los cuales se explican a continuación.

Números ordinales.

Un componente ordenado, como los grupos de edades, puede transcribirse como el número ordinal de cada grupo. Esto solamente activa una línea de la tabla como se muestra en la figura 3.11. En este caso, **Q** se reemplaza por **O**.

Individuos	250	X
		|
Grupos de edades	1	**O**

Figura 3.11.- Relación Individuos-Grupos de Edades ordinales

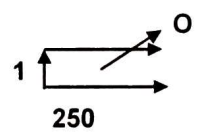


Figura 3.12.- Representación gráfica de la figura 3.11

La notación **O** indica la presencia de números ordinales.

La notación \ designa la presencia de celdas vacías o de datos comunes a varios elementos.

Datos en común.- Observemos los componentes en la siguiente figura. Conocemos la profesión de cada individuo. También conocemos el material con el que está fabricada la vivienda en la que habita. El material caracteriza tanto a la vivienda como a los individuos que viven en ella.

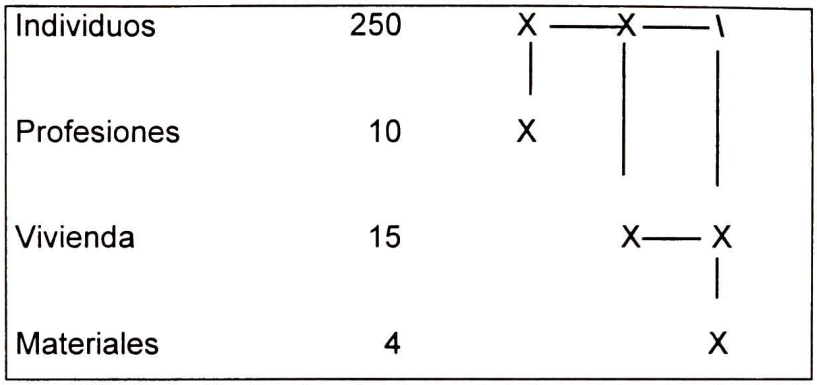


Figura 3.13.- Relación funcional entre materiales, vivienda e individuos

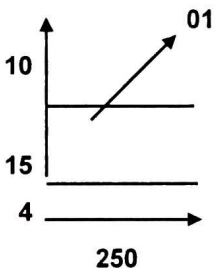


Figura 3.14.- Representación gráfica de la figura 3.13

Datos inaplicables a ciertas categorías de un componente.- Observando los componentes de la siguiente figura, obviamente, las profesiones no son aplicables a los niños, nosotros podemos representar este caso como se muestra a continuación.

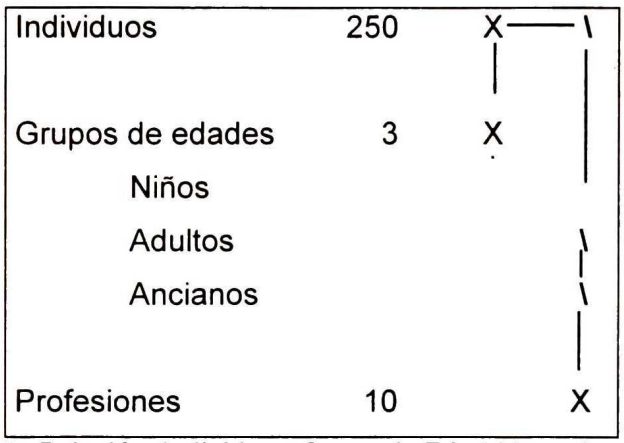


Figura3.15.- Relación Individuos-Grupo de Edad (datos inaplicables)

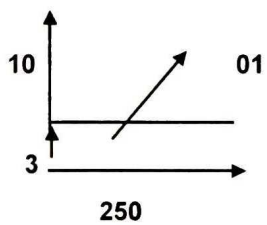


Figura 3.16.- Representación gráfica de la figura 3.15

Datos ausentes.- En el siguiente ejemplo se tiene que solamente se conoce la edad de 100 de los 250 individuos. Se debe reservar una fila para observar las intersecciones concernientes a este subgrupo de individuos. Información de este tipo crea un área desconocida en el esquema de homogeneidad.

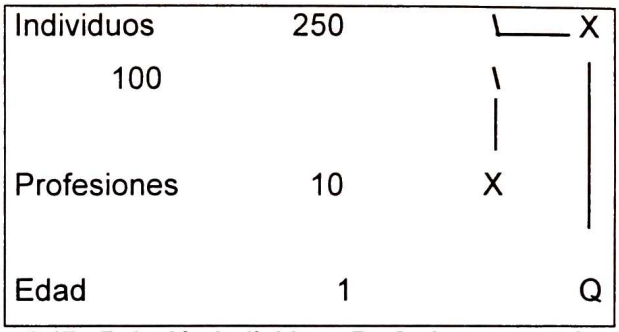


Figura 3.17.- Relación Individuos Profesiones cuando solo se conoce la información de 100 individuos

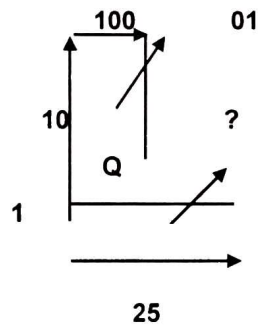


Figura 3.18.- Representación Gráfica de la figura 3.17

En la Tabla de Distribución, los datos de tipo carácter se representarán con una X y los datos que implican cantidad se representan con una Q.

3.3.2. Esquema de Homogeneidad

Una vez construida la tabla de datos en donde se detectan los componentes de estudio, se procede a identificar si los datos son homogéneos, es decir, se detecta el punto en común existente entre los componentes del estudio. Para ello se construye el **Esquema de Homogeneidad**.

El esquema de homogeneidad describe a grandes rasgos la tabla de la información. Nos permite definir una tabla homogénea para plantear problemas de metodología de procesamiento y concebir una tabla de datos utilizable. Esquematiza la matriz de datos construida por estos componentes.

El esquema de homogeneidad es un esquema *x y z* de la tabla construida por los datos. Un estudio es homogéneo si todos los elementos tienen un punto de comparación, este punto de comparación tiene una forma gráfica: *es el componente el cual es común a todos los demás componentes del estudio.* Esto es, es aquel componente que si se coloca en *x* se convierte en el OBJETO ESTADÍSTICO para el cual todos los demás componentes son sus CARACTERÍSTICAS las cuales pueden colocarse en el eje *y*.

Este esquema nos permite definir un punto de comparación para el estudio completo, esto es, encontrar el componente que será colocado en *x* lo cual permite colocar a todos los demás componentes del estudio en *y* . Es decir, con el esquema de homogeneidad se localizan los ***objetos estadísticos*** (que se colocarán sobre el eje *x*) y sus respectivas ***características*** o ***atributos*** (que se colocarán a lo largo del eje *y*).

En el esquema de homogeneidad, los *objetos estadísticos* se distinguen fácilmente cuando se observa una línea horizontal de cruces como se observa en la figura 3.19.2A.

A continuación se presenta una serie de ejemplos que permiten visualizar lo que es una tabla de distribución y un esquema de homogeneidad. Se debe tomar en cuenta las siguientes notaciones para el eje *z*:

- **01** Define una tabla cuyos valores son SI o NO.
- **Q** Define una tabla que contiene cantidades o números ordenados.
- **Qob** Define una tabla que contiene solamente cantidades de una población simple de objetos estadísticos, es decir, cuyo total general es significativo

A continuación se presenta un ejemplo para observar la tabla de distribución y el esquema de homogeneidad, la notación será la siguiente: I: Individuos, P:Profesiones, R: Sueldo, A: Edad.

I	250	X	X	X
		|	|	|
P	10	X	|	|
			|	|
R	1		Q	|
				|
A	1			Q

Figura 3.19.1A

I	250	X	X	X
		|	|	|
P	10	X	|	|
			|	|
R	3		X	|
				|
A	4			X

Figura 3.19.2A

I	250	Q
		|
P	10	X
		|
R	3	X
		|
A	4	X

Figura 3.19.3A

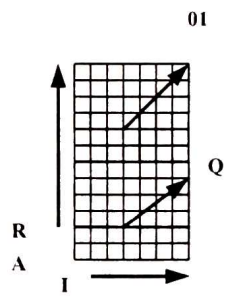


Figura 3.19.1B

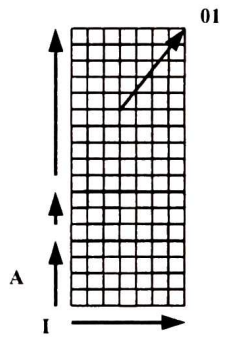


Figura 3.19.2B

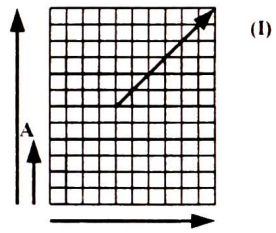


Figura 3.19.3B

I	---	Q Q
		| |
P	10	X X
R	3	X
A	4	X

Figura 3.19.4 A

I	---	Q
P	10	X |
		| |
R	3	| X
		| |
A	4	X 4 X

Figura 3.19.5A

I	250	Q Q Q
		| | |
P	10	X | |
		| |
R	3	X |
		|
A	4	X

Figura 3.19.6A

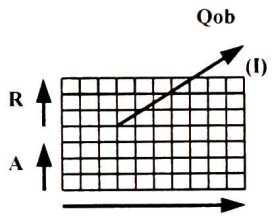


Figura 3.19.4B

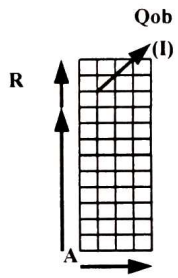


Figura 3.19.5B

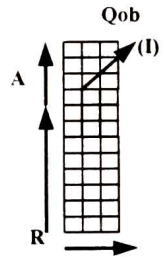


Figura 3.19.6B

I	---	Q Q Q
		| | |
P	10	X | |
		| |
R	3	X |
		|
A	4	X

Figura 3.19. 7A

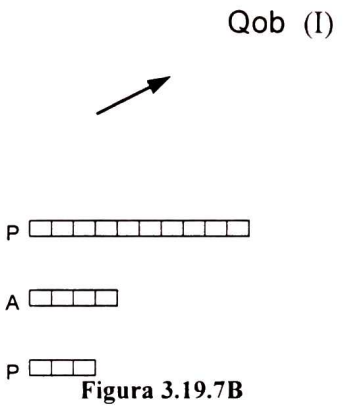


Figura 3.19.7B

Cuando hay solamente una línea vertical como se muestra en la figura 3.19.3 A, cada uno de los componentes con una marca de cruz puede colocarse a lo largo del eje X como se observa en la figura 3.19.3 B). Cuando hay varias líneas verticales pero ninguna línea de cruces horizontal tal como se observa en la figura 3.19.7 A) se tiene un conjunto heterogéneo sin ningún punto de comparación (Figura 3.19.7 B).

La tabla de distribución nos permite grabar el texto estadístico linealmente y en cualquier orden.

Con este esquema se puede definir cual es el objeto estadístico, cuáles las características relacionadas a dicho objeto y cuáles son sus intersecciones, una vez realizada este proceso, se comienza a analizar la tabla determinando el tipo de objeto estadístico y la cantidad de características. Una vez identificados estos elementos, se procede a realizar la construcción gráfica.

Cuando se tiene una línea con X y éstas se encuentran ligadas con una X en cada línea sucesiva, se puede decir que el elemento de la primera línea es el OBJETO y los elementos de las líneas restantes son sus CARACTERÍSTICAS. Las líneas verticales que unen a cada X se denominan intersecciones. Si se encuentra la notación Q quiere decir que esta característica tiene un valor único para el objeto.

3.4 CONSTRUCCIÓN DE GRÁFICAS

Una vez ordenados los datos dentro de la tabla de datos, el siguiente paso es determinar qué representación gráfica es la más apropiada para los datos tomando en cuenta los siguientes aspectos:

1) El número de renglones de la tabla, es decir el número de características;

2) La naturaleza de las series de objetos la cual es, por ejemplo, **ordenada** (0) para meses y días; **reordenable** (≠) para individuos o cantidades y **topográfico** (T) para pueblos.

Las dos grandes divisiones para la clasificación de las representaciones gráficas son :

a) Representaciones Gráficas para Objetos Reordenables (Datos de tipo numérico)

b) Representaciones Gráficas para Objetos Ordenados (Datos de tipo fecha)

Bertin [11] considera que dentro del primer grupo se encuentran las siguientes representaciones gráficas:

1. Matrices de Permutación
2. Tablas Ordenadas

Dentro de la clasificación de las Matrices de Permutación se encuentran las siguientes estructuras:

- Matriz Reordenable
- Arreglo de Curvas

3.4.1 Matrices de Permutación

Matriz Reordenable

La matriz reordenable es una construcción básica. Esta construcción es útil cuando la tabla de datos toma la forma ($\neq \neq$) (Objetos Reordenables) y no excede del tamaño x x y = 10,000.

En esta construcción, todas las filas son equivalentes a las columnas las cuales permiten la aplicación de permutaciones.

El proceso que permite la construcción de una matriz reordenable es el siguiente:

1) Verificar que todos los componentes de la tabla sean reordenables.

2) Elegir las filas con valores similares y colocarlas juntas, esto dará como resultado una imagen mas simple.

3) Elegir las columnas con valores similares y colocarlas juntas de manera que puedan observarse las relaciones completas y la interpretación es más simple.

Observemos el siguiente ejemplo que corresponde a un conjunto de poblaciones A, B, C,... P y los elementos que existen en cada una de ellas como son escuelas, estación de policía, veterinario, doctor, etc. La secuencia del proceso de permutación se presenta a continuación con el siguiente ejemplo:

A) Matriz de datos inicial

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P		
							■			■						1	Escuelas Secundarias
	■	■	■			■					■			■		2	Uniones de Agricultura
							■			■						3	Estación de tren
■	■			■	■			■	■			■	■		■	4	Escuelas Rurales
	■	■	■			■					■			■		5	Veterinarios
■	■			■	■			■	■			■	■		■	6	Sin Servicio Médico
									■				■			7	Sin Agua Potable
		■					■			■				■		8	Estación de Policia
	■	■	■			■					■				■	9	Terrenos urbanizados

Figura 3.20.- Tabla inicial de datos

B) Se colocan juntas las líneas similares.

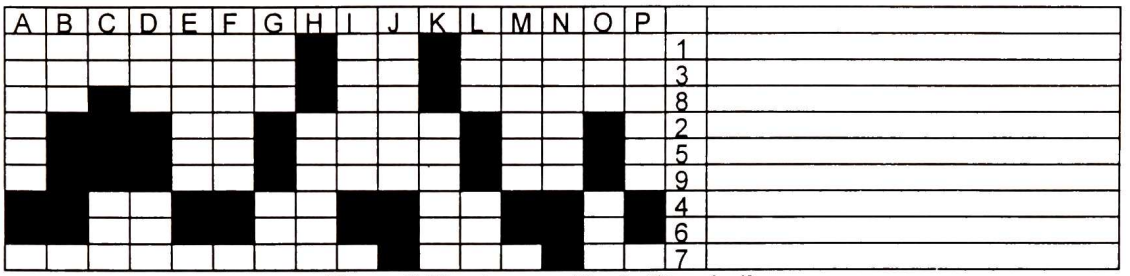


Figura 3.21.- Agrupación de filas similares

C) Se colocan juntas las columnas similares.

Figura 3.22.- Agrupación de columnas similares

De esta manera, una vez realizado el proceso de permutación, se descubren las relaciones completas entre los objetos y se pueden aplicar las mismas decisiones a los pueblos H y K que se basan en la presencia de las mismas características 1,3,8 y la ausencia de otras. Así mismo, las mismas decisiones no pueden aplicarse a las poblaciones N, J, P, M, I, F, E, A, B. Finalmente, las poblaciones O, L, G, D, C justifican un tercer conjunto de decisión.

La reclasificación visual, nos permite definir grupos característicos con situaciones particulares, tal como se observa la población B, la cual puede pertenecer a dos grupos diferentes.

Procedimiento

1o. Elegir una fila, si es necesario hacerlo de forma aleatoria o preferentemente una que incluya pocos valores desconocidos y similares, una que represente un fenómeno cuya influencia general pueda ser censada o medida. En este ejemplo se tomó como referencia la fila 1. Se coloca en la parte superior. Se toma la matriz verticalmente. La matriz entera será entonces clasificada a lo largo de x de acuerdo al perfil de la fila 1 este proceso se observa en la figura 3.20 y en la figura 3.21.

2o. Una vez que se ha obtenido la matriz con filas similares juntas, se procede a determinar cuáles son las columnas similares y estas se agrupan.

Exclusiones

Las permutaciones crean agrupaciones. Sin embargo, ciertas filas o columnas no caen dentro de estas agrupaciones y pueden quedar fuera. Ellas representan casos especiales en relación al conjunto considerado o a un nuevo sistema el cual puede ser simplificado posteriormente.

Arreglo de Curvas

El arreglo de curvas es una construcción estándar para una tabla $\neq \mathbf{0}$ (Objetos Reordenables-Ordenados). La curva tiene una ventaja, puesto que esta muestra cuestas y nos permite analizarlas en todos los niveles. Sin embargo, las cantidades están representadas por y . Las filas en la matriz, son por lo tanto irregulares y la técnica gráfica es más compleja. Dentro de esta clasificación entran las gráficas de línea y progresiones descritas en el capítulo 1.

3.4.2 Tablas Ordenadas

Estas representaciones gráficas involucran datos que lleven una secuencia (horas, meses, etc.)

3.2.2.1 Tablas con 1, 2 o 3 características.

Las tablas de una a tres características ofrecen dos construcciones posibles dependiendo de cómo se encuentren colocados los objetos A, B, C a lo largo de *x* o en *z*: las *Construcciones de Matriz* que acomodan los objetos a lo largo de *x* y las características a lo largo de *y* . Si los objetos son reordenables, deben ser reordenados.

Los *Scatter Plots* son "Tablas ordenadas". Los objetos se colocan en *z*, lo cual nos permite registrar directamente sin necesidad de permutaciones. Los Scatter Plots se utilizan cuando las relaciones completas son las consideraciones más pertinentes, cuando el orden lineal de los objetos no es indispensable y cuando el número de objetos es muy largo.

Objetos Reordenables (≠) con una característica.

El límite es descubrir grupos similares o vecinos de objetos y/o grupos de cantidades.

Si el número de objetos es muy extenso, los objetos pueden ser agrupados en un diagrama de distribución, o un diagrama de líneas.

Si el número de objetos es menor a 10, la representación gráfica sería a través de un diagrama de pastel.

Objetos Reordenables (≠) con dos características

Objetivo: Descubrir agrupaciones de objetos y/o la relación entre dos características. Aquí es donde dominan los diagramas de correlación o scatter plots.

La representación de objetos con dos características, se puede realizar a través de Scatter Plots, pues se considera a gráfica más adecuada para ello.

Objetos Reordenables (≠) con tres características

Las metas aquí son: a) descubrir grupos de objetos similares a lo largo de las tres características y b) descubrir, si se da el caso, dos características similares en relación a una tercera.

Las gráficas que se pueden generar son: gráficas de barras y Scatter Plots.

Objetos Ordenados (0) con una característica

Cuando se tiene una tabla cuyos objetos son ordenados (fechas) y sólo tienen una característica, entonces se recomienda representar esta información utilizando las series de tiempo o cronogramas, ya que nos permite observar las variaciones de los datos a través del tiempo.

Objetos Ordenados (0) con dos características

Objetivo: descubrir "períodos" definidos por dos características.

Cuando no es significativa la diferencia, entonces se utiliza una escala logarítmica. Las características de este tipo de gráfica se han descrito en el capítulo 1.

Cuando las dos características tienen una probabilidad de reactivarse directamente una a otra con o sin un intervalo de tiempo, se utilizan Scatter Plots.

Objetos Ordenados (0) con tres características

El objetivo es 1) descrubrir períodos definidos por tres características y 2) descubrir, donde sea posible, dos características que son aproximadas a una tercera.

Las representaciones gráficas para este tipo de datos son: el arreglo de curvas en escala logarítmica y el scatter plot.

El arreglo de curvas en una escala logarítmica compara las pendientes. Dos curvas similares, se deben colocar juntas y así poder observar como se asemejan las características.

3.5 CONCLUSION DEL CAPÍTULO

Dadas estas condiciones, el Sistema DALI realiza lo siguiente :

a) Identifica el objeto y el número de características relacionadas a ese objeto.

   Esto se realiza a través del procedimiento de la construcción de la tabla de datos y del esquema de homogeneidad.

b) Identifica el tipo de los datos sobre los cuales se realizará la construcción gráfica. A través del descriptor de archivos para identificar el tipo de dato que se va a graficar.

c) Determina el número de datos que se van a representar gráficamente.

d) Elige mediante una estructura CASE qué gráfica es la más apropiada para ese conjunto de datos.

CAPITULO 4

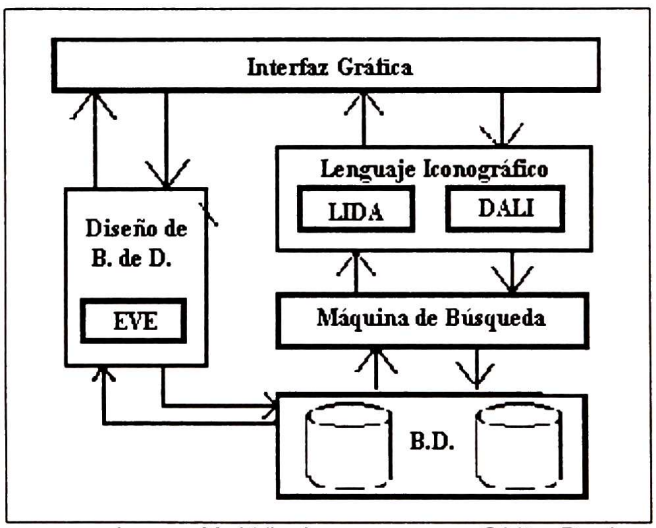
DALI: herramienta para la representación gráfica de información.

4.1. ANTECEDENTES: PROYECTOS DE PROGRAMACIÓN VISUAL

El sistema DALI forma parte del proyecto de "Programación automática a partir de Descriptores de Flujo de Información". El proyecto visual junto con otros trabajos de tesis de maestría y doctorado incluye dos tópicos principales de investigación: Visualización y Lenguajes Visuales y; Bases de Datos.

La propuesta original es a partir del lenguaje LIDA [4], el cual puede servir como lenguaje de simulación para negocios en donde DALI es un sistema incorporado que permitirá mostrar escenarios visuales de las bases de datos.

El objetivo general es un sistema de bases de datos con todo un entorno visual desde el diseño de las bases de datos hasta su explotación, incluyendo Minería de Datos.



El sistema tiene como base X Windows como GUI. Dado el objetivo general, el proyecto ha incluido los siguientes sistemas: EVE, Sistema para el diseño conceptual, lógico y físico de bases de datos, Entidad-Vínculo-Extendido [14]; LIDA, lenguaje visual que interacciona con una base de datos para resolver problemas de aplicaciones, Lenguaje Iconográfico para el Desarrollo de Aplicaciones[4]; DALI, ambiente visual para la presentación automática de gráficas de la información. HEVICOP, Herramienta Visual basada en el Paradigma 'BLOX' para construir programas en lenguaje C y C++

LIDA, desarrollado por el Dr. Sergio V. Chapa [4] es un lenguaje de flujo de datos que se describe de una forma gráfica; en donde se tienen símbolos gráficos que representan operaciones sobre los datos de entrada. Los objetos de datos, pasan a través de las líneas de flujo para entrar a módulos de transformación que son los íconos que tienen asociada una semántica a su imagen y con el uso de reglas sintácticas se disponen en una carta gráfica para conformar un flujograma.

EVE actualmente tiene dos versiones, la primera desarrollada en ambiente Windows de Microsoft y la segunda que corresponde más a este proyecto [3]. El sistema EVE permite diseñar bases de datos mediante gráficas del método extendido de Chen Entidad-Vínculo. El sistema deja un modelo lógico en Tercera Forma Normal normalizado bajo el algoritmo de Berstein [4]. Usa un esquema de descriptor de LIDA y DALI.

4.2 INTERACCIÓN DE LIDA CON DALI

El resultado de las operaciones efectuadas por LIDA sobre la Base de Datos, se registra en forma de tablas. Como se mencionó anteriormente, la información puede interpretarse mas fácilmente si se representa en forma gráfica que en una lista de datos; por lo tanto, es aquí donde el sistema DALI interviene para la representación de la información.

Una vez realizadas las operaciones indicadas en el flujograma, el usuario podrá invocar a DALI para que se le presente la información de manera gráfica.

4.2.1 Flujogramas

Los flujogramas son una herramienta gráfica que sirven para representar las funciones concernientes al manejo de la información, dentro de la estructura de una organización y representar la transformación de la información. De esta forma, los flujogramas se pueden utilizar para dos propósitos:

- Describir la gestión de una organización
- Procesar la información

Cuando los flujogramas se usan para describir la estructura operacional de la organización administrativa, el usuario representa en un lenguaje gráfico de dos dimensiones, las actividades y/o funciones que los órganos desempeñan.

El flujo de la información se representa gráficamente por flechas que consideran el tránsito de la información que fluye entre las unidades organizacionales, con la finalidad de ser transformada en cada una de ellas. De esta manera se tiene que los flujogramas son un excelente modelo para la representación de la gestión administrativa, sirviendo para responder preguntas acerca de los procedimientos administrativos.

Cada cuadro pictórico del flujograma está constituido de símbolos gráficos que representan objetos o actividades que son esenciales en cada uno de los elementos de la organización. Los símbolos son íconos que se ligan con flechas etiquetadas con el nombre de la "forma" de los documentos que contiene la información. En cada etapa se tiene una transición que se efectúa bajo una actividad o un conjunto de actividades, que suelen transformar y procesar la información para dar como resultado el segundo propósito de procesado de la información.

Los flujogramas como un modelo para describir el procesamiento de datos, consiste en la automatización de los procedimientos que manejan la información en alguna parte del sistema. El flujograma en esta parte expresa precisamente la relación funcional entre los documentos y el sistema. Esto quiere decir cómo el sistema va a efectuar las transformaciones a los documentos con un conjunto de procesos específicos.

En un flujograma, los procesos están descritos en una reseña gráfica que tiene íconos y líneas de flujo de datos. En los íconos se tiene la transformación de la información, y en los arcos, el flujo de la misma información que sale de algunos íconos para entrar a otros.

Las principales características de LIDA son:

a) Se basa en el modelo de datos relacional.

b) Es un lenguaje de flujo de datos.

c) Su programación es visual.

En el sistema LIDA, a través del descriptor de archivos los esquemas que se definen son relaciones, siendo estas los objetos de datos que principalmente fluyen en las líneas de flujo de datos. El modelo de datos relacional es el enfoque de abstracción de datos de LIDA que pretende deliberadamente omitir ciertos detalles al usuario final, manteniendo solo los que son relevantes en las aplicaciones.

El modelo de flujo de datos que tiene LIDA da un enfoque de abstracción procedural debido a que el usuario solo trata con íconos que representan procesos, además que el flujo de datos determina el control del flujo. Los arcos conectan a íconos que determinan la abstracción procedural como transformaciones de objetos de entrada a salidas. Las representaciones icónicas en LIDA, permiten invocar a los procedimientos que encapsulan operaciones, sin tomar en cuenta la representación de los objetos ni la implementación de las operaciones. Los íconos admiten argumentos que pueden ser atributos, expresiones aritméticas o expresiones booleanas. Estas últimas se organizarán como argumentos en íconos que son constructores condicionales que en su ejecución determinarán bifurcaciones en el flujo de datos mostrando una vista en paralelo como se observa en la figura 4.1.

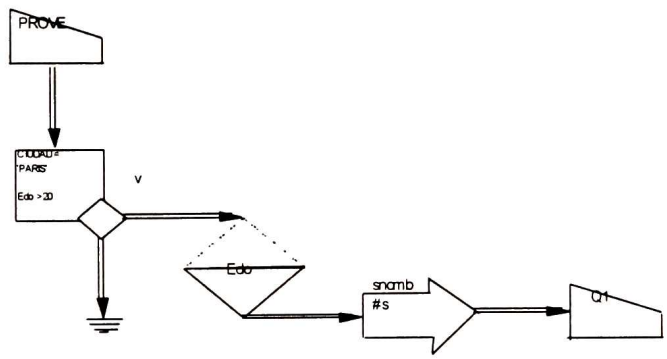


Figura 4.1.- Ejemplo de flujograma en LIDA

Esta figura es un flujograma que representa el planteamiento de un problema de consulta a una base de datos. El problema es el siguiente:

Dados los siguientes esquemas de relaciones:

PROVE(#S, SNOMB, EDO, CIUDAD)

PARTE(#P, PNOMB, COLOR, PESO, CIUDAD)

ORDEN(#S,#P, CANTIDAD).

Queremos obtener el nombre del proveedor y su número, para los proveedores que se encuentran en la ciudad de París y tengan un estado mayor que 20.

```
SQL>  SELECT SNOMB, #S
      FROM PROVE
      WHERE CIUDAD = 'Paris' AND EDO >20
      ORDER BY EDO DESC
```

En el flujograma se puede observar que de la relación Orden y Parte, fluyen simultáneamente para entrar a dos íconos de selección. En estos se seleccionan las n-adas que cumplen con ciudad igual a "París" y Estado mayor a 20. Posteriormente, como la operación de enmedio es una unión y requiere que los esquemas de relación sean iguales, solo proyectamos el atributo #P con el ícono flecha para que el ícono Union procese las dos relaciones unarias y arroje el resultado en la relación Q1.

Un rasgo distintivo de LIDA es su programación visual, que se establece en la edición y puede persistir hasta la documentación. El usuario solo trata con la disposición de íconos en la pantalla y conexiones de líneas. Considerando además, cuando la semántica del ícono lo requiere, añadir texto que son argumentos. Dadas estas características de edición gráfica, se puede decir que el sistema LIDA puede ser visto como un sistema de Diseño Asistido por Computadora para la programación.

En LIDA, la sintaxis está dada por las líneas de flujo que conectan a los íconos. Los íconos que maneja son los siguientes:

- Iconos terminales de los cuales solo entran líneas de flujo de datos e iniciales de las cuales salen. Estos son *escribe relación* y *lee relación*.
- Iconos que representan operadores binarios de los cuales siempre están esperando dos relaciones de entrada y arrojan una salida.
- Iconos unarios que aceptan una relación y dan como resultado otra relación.
- Iconos que aceptan una relación y bajo una condición establecida con una expresión booleana dividen las trayectorias de flujo en dos, una verdadera y otra falsa.
- Iconos funciones que aceptan en su entrada una relación y arrojan un valor que se mandan a un ícono rectángulo. Esta será identificada por un nombre.

4.2.2 Descripción general del sistema LIDA

Los componentes de LIDA son los siguientes:

- Módulo denominado Descriptor que permite capturar la definición de los datos y estructurar los esquemas de relación. El procedimiento es por despliegue de ventanas, manejo del cursor y actualización de datos. Este proceso es equivalente al llevado a cabo por un Lenguaje de Definición de Datos (DDL)

- Módulo capturador de datos que permite construir la base de datos. El enfoque también es de ventanas y el formato de columnas lo proporciona la definición de los datos dados por el módulo descriptor.

- Un editor gráfico para los flujogramas

- Un intérprete de código de cadena. Que es el lenguaje intermedio del sistema denominado ISBL extendido[4].

4.3 ARQUITECTURA GENERAL DEL SISTEMA DALI

Definición: DALI es un sistema de visualización de datos que tiene como objetivo representar de una manera gráfica, los resultados obtenidos por las operaciones realizadas en LIDA, se considera una herramienta visual para la representación gráfica de datos. Está desarrollada bajo un ambiente de XWindows que proporciona facilidades gráficas.

El sistema DALI está integrado por los siguientes módulos:

a) Módulo para la Definición y Manejo de archivos.

b) Analizador de Datos

c) Constructor gráfico

La interrelación de estos módulos se observa en la figura 4.2.

a) El módulo para la Definición de Archivos crea descriptores en donde se indica el nombre, tipo y tamaño de cada campo que compone el archivo. Una vez definido el archivo, el sistema permite introducir información. También permite leer y escribir desde y hacia el disco los archivos que se desean representar gráficamente.

b) El módulo de Clasificación y Análisis verifica el tipo de los datos que se desean visualizar, el número de campos o características del archivo, crea la Tabla de Distribución y la Matriz de Datos que se describen en el capítulo 3. Una vez creada la matriz, se analizan los datos para definir el tipo de gráfica más apropiado para su representación. El analizador está integrado por una serie de procedimientos que realizan el análisis que se describió en el capítulo 3.

c) El Módulo de Construcción Gráfica crea una estructura de datos para la representación gráfica denominada *Graff,* ésta contiene toda la información que generó el módulo de Análisis referente a la gráfica más apropiada. Toma de esta estructura la información y construye la gráfica especificada por el análisis utilizando las funciones de Xlib y Motif para su despliegue en la pantalla.

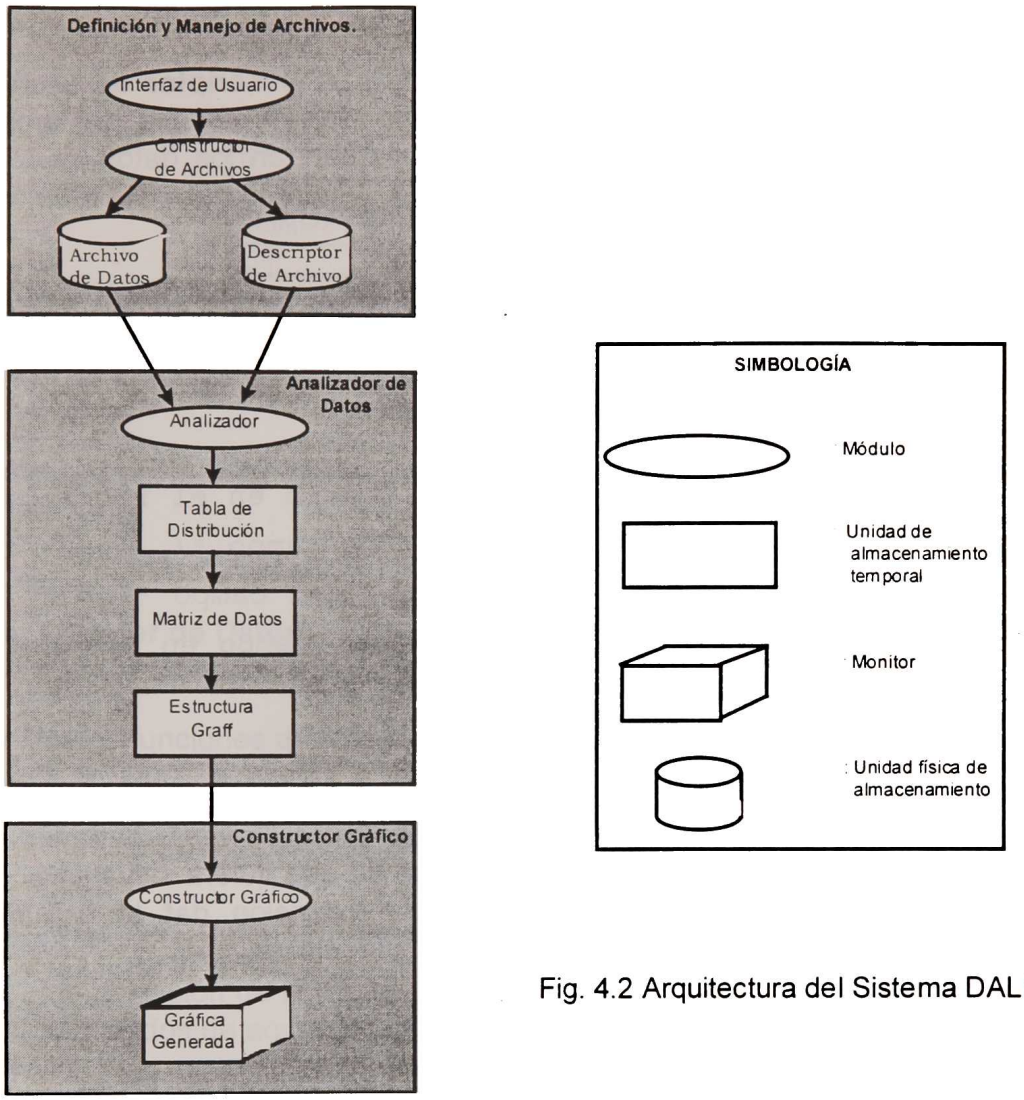


Fig. 4.2 Arquitectura del Sistema DALI

4.3.1 Definición de la Base de Datos

Debido a que aún continúa en desarrollo como proyecto de tesis la construcción del sistema LIDA en ambiente Xwindows, el sistema DALI permite definir archivos de datos para que a partir de ellos se lleve a cabo la construcción gráfica de manera independiente. En el momento en que se integre LIDA y DALI, la información será leída desde las tablas que genera LIDA.

Para definir los archivos de datos que serán representados, el sistema DALI presenta un Menú de Opciones en donde el usuario podrá elegir el proceso que desea realizar. Este menú se describe en el siguiente capítulo.

El módulo de definición de la base de datos genera un **descriptor de archivos** en donde se guardan las características de cada uno de los campos del archivo, se maneja un arreglo de descriptores en donde se encuentran los datos de todos los campos.

A continuación se presenta la estructura **descriptor** en donde se lee y escribe la información de los campos:

```
struct Descriptor {
     int posición,  /* Posición del campo en el registro */
   enum Tipo_camp Tipo,      /* Tipo de campo */
    char nombre[TAM_NOM],    /* Nombre del campo */
   int tam,                /* Tamaño del campo en bytes */
 } Campos [MAX_CAMPS]
```

Una vez definidos los campos con sus correspondientes tipos y tamaños, se introducen los datos a través de una de las opciones del menú. Estos se van ordenando en una lista doblemente ligada y posteriormente, en el caso de que quiera salvarse el archivo en disco, la información se guarda en un archivo con extensión **.dat**. La estructura de la lista ligada se muestra a continuación:

```
struct Db_Tipo {
  char *Datos;     /* apuntador a los datos */
  struct Db_Tipo *Ant;  /* apuntador al reg. anterior */
  struct Db_Tipo *Sig;  /* apuntador al siguiente reg.*/
} *Primero, *Ultimo, *cur_rec;
```

Se crea un archivo para el descriptor de cada .dat con extensión **.def** y este también se graba en disco.

En el momento en que se desee realizar el análisis de un archivo, se realiza el proceso inverso: se lee del archivo con extensión **.def** la información correspondiente al tipo y tamaño de los campos se guarda en el arreglo de estructuras **Campos** y los datos contenidos en el archivo con extensión **.dat** se van colocando en la lista ligada para que a partir de ahí puedan ser leídos en el proceso de Análisis. La estructura en donde quedará guardada la información del descriptor se muestra en la figura 4.3.

Estructura Campos

0	30	
Char	int	
Nombre_emp	Edad_emp	
30	2	
[0]	[1]	....

Figura 4.3.- Arreglo Campos (Guarda datos del descriptor)

Los pasos para la realización del análisis se describen en la siguiente sección.

4.3.2 Analizador de Datos

Las principales funciones del analizador son:

a) Creación de la tabla de distribución y verificación de la homogeneidad de los datos.

b) Creación de la matriz de datos.

c) Identificación de la gráfica más adecuada para la representación de los datos analizados de acuerdo al número de características encontradas.

d) Creación de la estructura Graff.

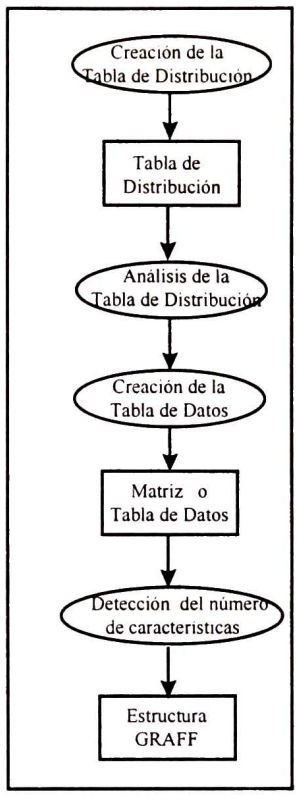


Figura 4.4.- Procesos del Análisis de Datos

a) **Creación de la Tabla de Distribución.** Este proceso consiste en clasificar los campos de acuerdo a su tipo y asignarles a cada uno de ellos una nomenclatura ("O", "Q", "X") para determinar si son datos ordenados, reordenables o de tipo texto. Los campos se leen del descriptor de archivos, se clasifican y se almacenan en una estructura de datos a la que se denomina Tabla de Distribución la cual se describió en el capítulo 3.

b) Una vez construida la tabla de distribución, se procede a realizar la verificación de los datos, es decir, verificar si en el conjunto de datos existe un punto en común el cual tenga relación con el resto de los datos, esto es lo que se denomina Esquema de Homogeneidad (Capítulo 3). A partir de aquí, se determina el **objeto** y las **características** del conjunto de datos.

c) La matriz de datos es un arreglo de apuntadores de tipo char en donde se van a almacenar temporalmente los datos del archivo. Se almacenan los datos **objeto** y sus correspondientes datos **característica** de acuerdo a la información contenida en la Tabla de Distribución.

d) En la primera fila se colocan los datos **objeto** y en las restantes, los datos **característica**.

e) Se determina el número de características de la matriz y el tipo de las mismas y esta información se considera para la creación de la estructura Graff.

f) Se crea la estructura Graff la cual contiene toda la información que dará origen a una gráfica.

Creación de la Tabla de Distribución y Verificación la homogeneidad de los datos

Para la creación de la Tabla de Distribución se siguen los siguientes procesos:

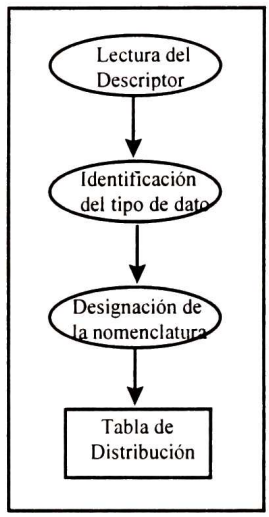


Figura 4.5.- Creación de la Tabla de Distribución

Lectura del descriptor.- Se lee la información contenida en el archivo con extensión **.def** y se va almacenando en el arreglo de registros denominado "Campos", como se indica en la sección anterior.

Identificación del tipo de dato.- Se comienza a analizar el tipo de cada uno de los campos para asignarle su correspondiente nomenclatura.

Designación de la Nomenclatura.- Se asigna la nomenclatura dependiendo del tipo de dato:

X si es de tipo texto

Q si es numérico

O si es tipo fecha

Se escribe en la estructura Reg_Tabdis el nombre del campo, su nomenclatura y el valor de dimensión.

```
struct Registro {
     char Elemento[40];
     int Dimension;
     char Nomen[50][3];
};

struct Registro Reg_TabDis;
```

`Elemento` .- Nombre del campo.

`Dimensión`.- Número de elementos de este campo.

`Nomen` .- Arreglo de nomenclaturas.

Se abre el archivo TabDis y se almacenan ahí los registros de Reg_Tabdis.

Creación de la tabla o matriz de datos

Una vez creada la tabla de distribución se realizan los procesos que a continuación se indican:

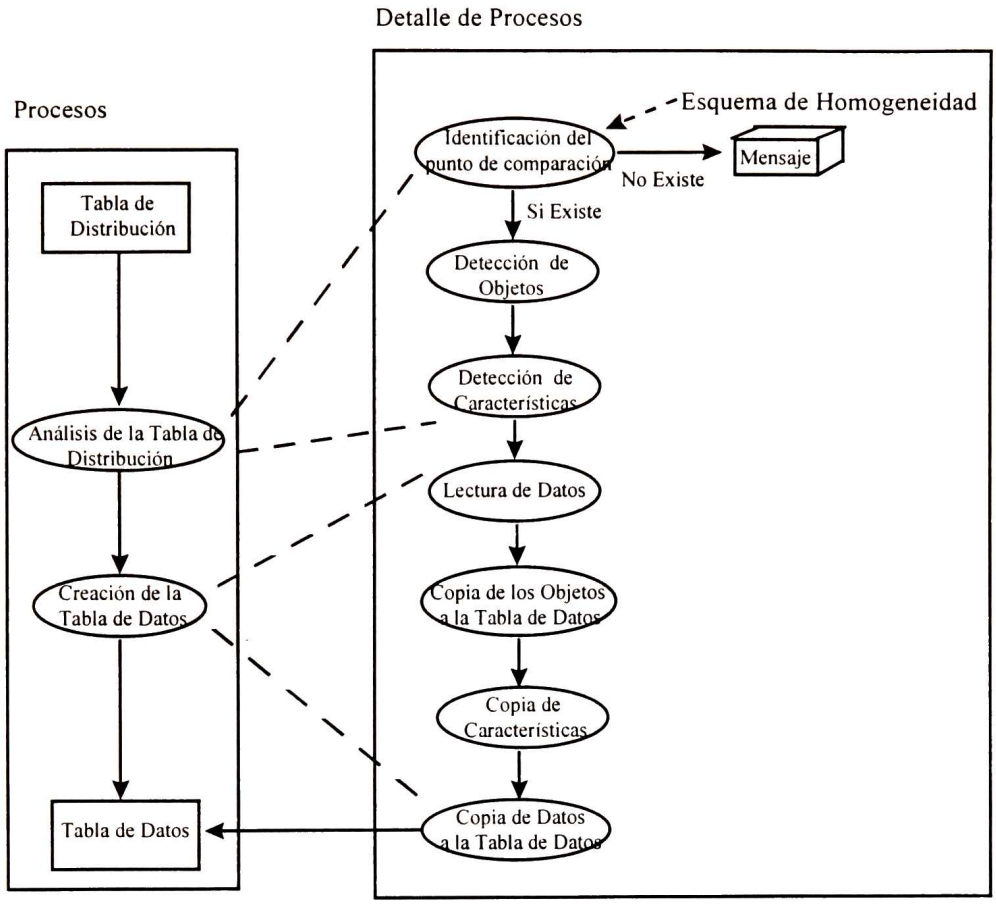


Figura 4.6.- Construcción de Tabla de Datos

Esquema de Homogeneidad.- Es aquí donde se debe identificar si existe un punto de comparación para definir si los datos son homogéneos y proseguir el análisis de la información.

Detección de objetos y características.- Los datos se leen de la lista ligada y se van guardando en vectores de tipo apuntador a char.

Se verifican todos los datos para identificar el dato **objeto** y los datos **característica**. Esto se realiza identificando qué datos se repiten:

Si todos los datos de un campo son diferentes entonces este campo se considera el campo **objeto**.

Si algún dato de un campo se repite entonces este campo se considera una **característica**.

Lectura de datos e integración a la Tabla de Datos.- Los vectores de datos se integran en sus correspondientes renglones a la matriz de datos de acuerdo a su tipo: **objeto** o **característica.**

Obtención del número de características.- Este procedimiento consiste en determinar el número de campos del registro relacionados al campo llave, para que así se obtenga el número de características.

Creación de la estructura graff.- Una vez determinado el número de características se crea la estructura graf cuya definición se presenta a continuación.

```
typedef struct {
     int g_id; /* Número de identificación de la gráfica */
      int tipo; /* Tipo de gráfica */
     int ncarac; /* Número de características */
     int sep; /* si la estructura pastel esta separada, en
            3d o   normal */
     int x;    /* Posición x en la pantalla  */
     int y;    /* Posición y en la pantalla  */
     char tit[30];     /* Título de la gráfica */
     char stit1[30];
     char stit2[30];
     char nomy[30];    /* Nombre del eje y */
     char nomx[30];    /* Nombre del eje y */
     char ser1[30];    /* Nombre de la serie 1 */
     char ser2[30];    /* Nombre de la serie 2 */
     char ser3[30];    /* Nombre de la serie 3 */
     char ser4[30];    /* Nombre de la serie 4 */
     char unid1[30];
     char unid2[30];
     char unid3[30];
     char unid4[30];
     int ndatx;          /* Número de datos en el eje x */
     char *arrx[100];  /* Datos del eje y */
     char *arr1[100]; /* Datos del eje y */
     char *arr2[100]; /* Datos del eje y */
```

```
        char *arr3[100]; /* Datos del eje y */
        char *arr4[100]; /* Datos del eje y */
        int tamdx;          /* Tamaño en bytes de los datos en
x */
        int tamd1;     /* Tamaño en bytes de los datos en x */
        int tamd2;     /* Tamaño en bytes de los datos en x */
        int tamd3;     /* Tamaño en bytes de los datos en x */
        int tamd4;     /* Tamaño en bytes de los datos en x */
        int tamcampX;
        int tamcamp1;
        int tamcamp2;
        int tamcamp3;
        int tamcamp4;
        } graf_ ,  *graf;
```

4.3.3 Constructor Gráfico

El constructor gráfico está integrado por los siguientes procesos:

a) Determinación de la gráfica más apropiada
b) Actualización de la estructura Graff
c) Rutinas para la creación del ambiente gráfico.
d) Rutinas para la generación de gráficas.

a) Determinación de la gráfica más apropiada

Un elemento fundamental en el análisis de los datos es el número de características de la matriz de datos y el tipo de los datos que se desean graficar. Una vez determinado el tipo de datos y habiendo obtenido el número de características, se realiza el procedimiento de definición de la gráfica apropiada a través de la rutina Objetos_reordenables que se basa principalmente en asignar la función gráfica correspondiente al conjunto de datos analizado dependiendo del tipo de datos y el número de **características** asociadas al **objeto.**

b) Llenado de la estructura graff

En esta estructura se guardan todas las características de la gráfica como son:

- Número de características o variables
- Título de eje x
- Título de eje y
- Nombre de etiquetas
- Datos en x
- Datos en y

que serán necesarias en el momento de generar la gráfica.

c) **Rutinas para la creación del ambiente gráfico**

El sistema DALI fue desarrollado bajo el ambiente Xwindows en una estación de trabajo DEC. Para la interfaz gráfica se utilizaron funciones de las bibliotecas de Motif y Xlib las cuales permiten el manejo de ventanas y objetos visuales como push button, menus, scroll bars, etc., que se describen en el siguiente capítulo.

Las gráficas son generadas dentro de ventanas o **formas** las cuales contienen ciertas características y están en función de los **eventos** que se presentan a su alrededor. Cada forma tendrá un contexto gráfico el cual deberá ser inicializado antes de generar la gráfica.

Para la creación de una forma que va a contener la gráfica se utiliza la siguiente función

```
shell[indi]    =    XtAppCreateShell(sheln[indi],"Grafica",
                applicationShellWidgetClass,
                XtDisplay(toplevel), al, ac);
```

Para la inicialización del contexto gráfico se utiliza la función

```
drawing_area[indi]              =XmCreateDrawingArea(formg[indi],
                    dwn[indi],    al,ac);
get_colors(drawing_area[indi]);
```

```
setup_gc(drawing_area[indi]);
```

d) Rutinas para la generación de gráficas

Dentro de la estructura de control CASE se encuentran las funciones gráficas que reciben como parámetro la estructura de datos Graff. La función que genera las gráficas se denomina **genera_g**.

A continuación se enlistan las diferentes gráficas que se generan de acuerdo a los parámetros que el análisis toma en cuenta:

Tabla 4.1 Correspondencia del número de características con su gráfica representativa

Número de Características	Número de elementos	Ausencia	Progresión Geométrica	Bandera	Tipo Gráfica
1	<=12	No			Proporcional
		Sí			Scatter Plot
		No	No		Barras
			Sí		Gráfica Logarítmica
	>12	Sí			Scatter Plot
		No	No		Gráfica de línea
			Sí		Gráfica Logarítmica
2	<12	Sí			Scatter Plot
		No	No		Barras
			Sí		Gráfica Logarítmica
	>12	Sí			Scatter plot
		No	No		Línea
			Sí		Gráfica Logarítmica
3	<=12			No	Barras
	>12			No	Gráfica de Línea
4 o 5					Matriz de Permutación
>5					Cara de Chernoff

Número de características.- Es el número de campos relacionados a un campo objeto, es decir, el número de atributos de un campo llave.

Número de elementos .- Es el número de renglones de la matriz de datos.

Ausencia.- Se comprueba si existe ausencia de datos, es decir, si algún valor de algún campo no es conocido.

Progresión Geométrica.- Este parámetro indica si los valores que se encuentran registrados en la matriz de datos siguen una progresión geométrica.

Con base en los anteriores conceptos es como se elige la gráfica apropiada para el conjunto de datos que se encuentra almacenado en la matriz o tabla de datos.

Una vez determinado el tipo de la gráfica a generarse, se ejecuta el procedimiento de la gráfica correspondiente. El parámetro principal para cada uno de estos procedimientos es la estructura **Graff** la cual contiene toda la información necesaria para la generación de la gráfica.

4.4 CONCLUSIONES AL CAPÍTULO

- El sistema DALI es la herramienta que genera gráficas estadísticas y multivariadas para la representación de datos.

- El siguiente paso es unir esta herramienta al sistema LIDA que actualmente se encuentra en desarrollo bajo un ambiente Xwindows.

- Como extensión al presente trabajo se pueden desarrollar rutinas para gráficas estadísticas en 3D.

Capítulo 5

SISTEMA DALI Y SU AMBIENTE GRÁFICO XWINDOWS

5.1 ANTECEDENTES

Desde Macintosh de Apple y PCs de MS-DOS hasta las estaciones de trabajo gráficas como Sun y DEC Station, ha ido cambiado la manera en que el usuario interactúa con las computadoras. Las capacidades gráficas han hecho posible interfaces gráficas para usuarios como la Macintosh User Interface y Microsoft Windows. Con estas interfaces, en lugar de ejecutar comandos en línea, se puede utilizar el dispositivo apuntador mouse para ejecutar programas, editar, copiar y borrar archivos. Adicionalmente, las interfaces gráficas dividen la pantalla de despliegue físico en regiones (generalmente rectangulares) denominadas *ventanas* donde aparece la salida de diferentes aplicaciones.

Para ambientes con sistema operativo UNIX, la herramienta que permite la generación de aplicaciones a través del manejo de ventanas es el sistema X Windows[15].

X fue desarrollado conjuntamente por el proyecto Athena de MIT y la Corporación Digital Equipment con contribuciones de muchas otras compañías. Fue dirigido por Robert Scheifler y alumnos del MIT. La versión sobre la que se desarrolló el sistema DALI es la versión X11 registrada en enero de 1990. Los principales conceptos que se manejan en este ambiente se describen brevemente a continuación.

5.2 CONCEPTOS GENERALES DE X WINDOWS

5.2.1 Displays y pantallas

La principal característica de X es que es un sistema de ventanas para despliegues gráficos de mapas de bits. En X windows, un *display* se define como una estación de trabajo constituida por un teclado, un dispositivo apuntador (mouse) y una o más pantallas. Las pantallas múltiples pueden trabajar juntas.

5.2.2 El Modelo Cliente-Servidor

El sistema X es un sistema orientado a red. Una aplicación no necesariamente puede correr en el mismo sistema que soporta al display. Mientras que muchas aplicaciones se ejecutan localmente, otras pueden ejecutarse en otras máquinas enviando solicitudes a través de la red para un despliegue particular y recibir eventos del teclado y del ratón desde el sistema que controla el display.

El ***servidor*** proporciona un servicio a las solicitudes del ***cliente***. Generalmente, los clientes se comunican con el servidor a través de la red y el cliente y el servidor intercambian datos usando un protocolo entendible por ambas estaciones de trabajo. Así como un servidor de archivos almacena archivos y permite que los clientes los accesen y manipulen, el servidor de display X ofrece servicios de despliegue gráfico para clientes que envían las solicitudes del protocolo X al servidor.

El servidor actúa como intermediario entre los programas de usuario que corren en cualquier sistema ya sea local o remoto y los recursos del sistema local. El servidor desarrolla las siguientes tareas:

- Permite el acceso al display por múltiples clientes.
- Interpreta los mensajes de red de los clientes.
- Pasa la entrada del usuario a los clientes enviando mensajes de red.

  Mantiene estructuras de datos complejas incluyendo ventanas, cursores, fonts y contextos gráficos.

En contraste con los servidores de bases de datos o servidores de archivos (los cuales por lo general son procesos que se ejecutan en máquinas remotas) el servidor de display X es un proceso que se ejecuta en la estación de trabajo mientras que los clientes pueden estar corriendo en computadoras remotas.

A continuación se presenta un esquema del modelo **cliente/servidor**.

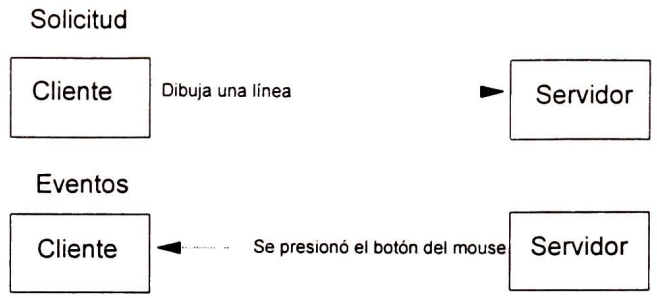


Figura 5.1.- Representación del modelo **cliente/servidor**

5.2.3 Manejo de Eventos

El servidor X considera cualquier cosa que se realice con el teclado o con el mouse como **eventos** que serán reportados a los clientes. Cuando se corren aplicaciones en X, todo en la pantalla aparece dentro de ventanas y cada ventana está asociada con un cliente específico. Cuando se presiona y se suelta el botón de mouse, el servidor X envía estos eventos de entrada al cliente que originalmente creó la ventana que contiene el apuntador del mouse. Para los eventos del teclado la tecla presionada siempre pertenecerá a la ventana actual designada.

El servidor X envía otro tipo de evento a los clientes, un **evento expose.** Este evento indica al cliente si algo sucede en su ventana.

5.2.4 Las capas del Sistema X

Protocolo X: El lenguaje de máquina de X

El protocolo define la manera en que el servidor y el cliente van a intercambiar información y como va a ser interpretada. En el protocolo X, los datos se intercambian en forma asíncrona sobre una comunicación de dos sentidos que permite la transmisión en ocho bytes, esta capa se considera el lenguaje de máquina de X.

Xlib: Lenguaje ensamblador del sistema X

El sistema X Windows viene con una biblioteca de rutinas en C denominada *Xlib*. Xlib permite el acceso al protocolo X a través de más de 300 rutinas de utilería. Si el protocolo X se considera el lenguaje máquina de X, entonces Xlib es el lenguaje ensamblador, es decir, a través de él se pueden ejecutar instrucciones del protocolo X.

X Toolkits: Lenguajes de alto nivel del sistema X

Este es un conjunto de rutinas que permite crear botones, listas y menús las cuales pueden utilizarse para construir una interfaz gráfica. El sistema X contiene las herramientas *X Toolkits Intrinsics*, el cual utiliza la implementación orientada a objetos para crear bloques de construcción básicos llamados **widgets**. Existen otros toolkits como Motif creado por la Open Software Fundation y OPEN LOOK creado por SUN los cuales usan un alto nivel de abstracción. La herramienta Motif está construída sobre X Toolkits Intrinsics y se consideran como el lenguaje de alto nivel del sistema X, ya que es a través del cual el usuario puede desarrollar aplicaciones en el ambiente Xwindows.

La siguiente figura muestra la estructura general de una aplicación X. La aplicación primeramente llama rutinas desde un toolkit. El toolkit llama rutinas desde Xt Intrinsics las cuales a su vez llaman a Xlib. La aplicación puede también hacer llamadas directamente a rutinas de Xlib para generar salidas de texto y gráficas en una ventana.

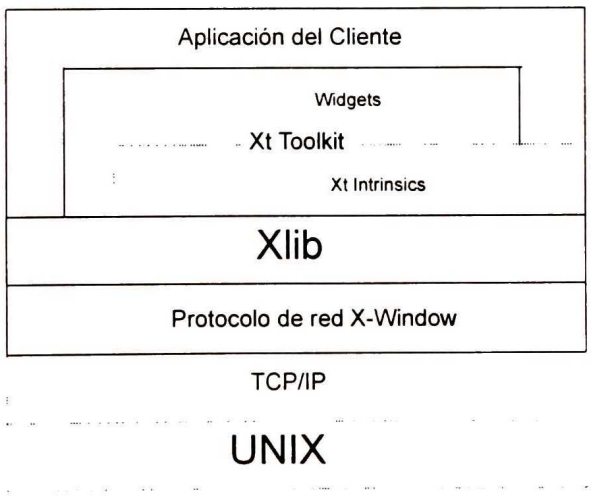


Figura 5.2.- Capas del Sistema Xwindows

Para el desarrollo del sistema DALI, se utilizaron las bibliotecas de Xlib para la generación de gráficas, manejo de color y texto.

Para la creación de la interface gráfica (ventanas, menús, botones, scroll bars, etc.), se utilizó el Toolkit MOTIF.

5.3 MOTIF Y SU NATURALEZA ORIENTADA A OBJETOS

Esta es una herramienta que permite a los programadores diseñar conjuntos de widgets para crear la interfaz gráfica en ambiente UNIX.

El conjunto de widgets es típico: contiene los widgets de scroll bar, de botón, de menu, de texto, etc. Un widget en MOTIF es equivalente a un objeto gráfico en Visual Basic.

Por su diseño, el widget aparece muy orientado a objetos para el programador. Debido a que la programación de estas bibliotecas es en lenguaje C, no es completamente orientado a objetos. Este toolkit proporciona las ventajas de la programación orientada a objetos sin que el programador requiera aprender un nuevo lenguaje.

En la programación de MOTIF, cada objeto de la interface de usuario (o widget) está controlado por un conjunto de variables llamadas **recursos.** Al cambiar los recursos, se puede controlar la apariencia y comportamiento del widget. Al leer los recursos, se puede conocer el estado del widget. El widget también puede enviar mensajes, conocidos como **callbacks** cuando quiera comunicarse con el código restante.

El ambiente de programación orientada a objetos soporta jerarquía de objetos. Para construir un nuevo objeto, se usa y se agrega uno existente en un proceso llamado **herencia.** El nuevo objeto puede hacer cualquier cosa que el objeto original haga así como cualquier otro proceso que se le adicione. Se pueden combinar objetos existentes dentro de un nuevo objeto. Similarmente, Motif usa la herencia para construir widgets como antecesores de otros widgets o fuera de grupos de widgets. Todos los widgets de MOTIF usan la herencia internamente.

Cada widget puede mandar mensajes de salida llamados **callbacks** para las funciones del programa cuando un usuario manipula un widget.

Existen dos grandes ventajas para manejar objetos de interface de usuario en esta forma. Primero, se puede modificar la apariencia y comportamiento del widget. Segundo, el widget maneja todo lo referente al manejo de eventos de bajo nivel. Motif permite que el usuario desarrolle interfaces de usuario gráficas de una manera más sencilla.

Algunos de los widgets más comunes en la programación de la interfaz gráfica son los siguientes:

XmForm.- Manejador de widgets que funge como contenedor de widgets hijos.

XmPushButton .- Permite al usuario ejecutar un comando tecleando un botón.

XmText .- Proporciona capacidades para la edición de texto.

XmLabel .- Despliega una cadena de caracteres.

XmList .- Permite al usuario elegir una o múltiples opciones desde una lista.

Para la creación de estos widgets se utiliza la función:

XtCreateWidget(

String nombre,

WidgetClass class,

Widget padre,

ArgList args,

Cardinal num_args);

Donde

nombre es el nombre del widget

class es la clase del widget

padre el padre del widget que se va a crear

args lista de argumentos

num_args número de argumentos

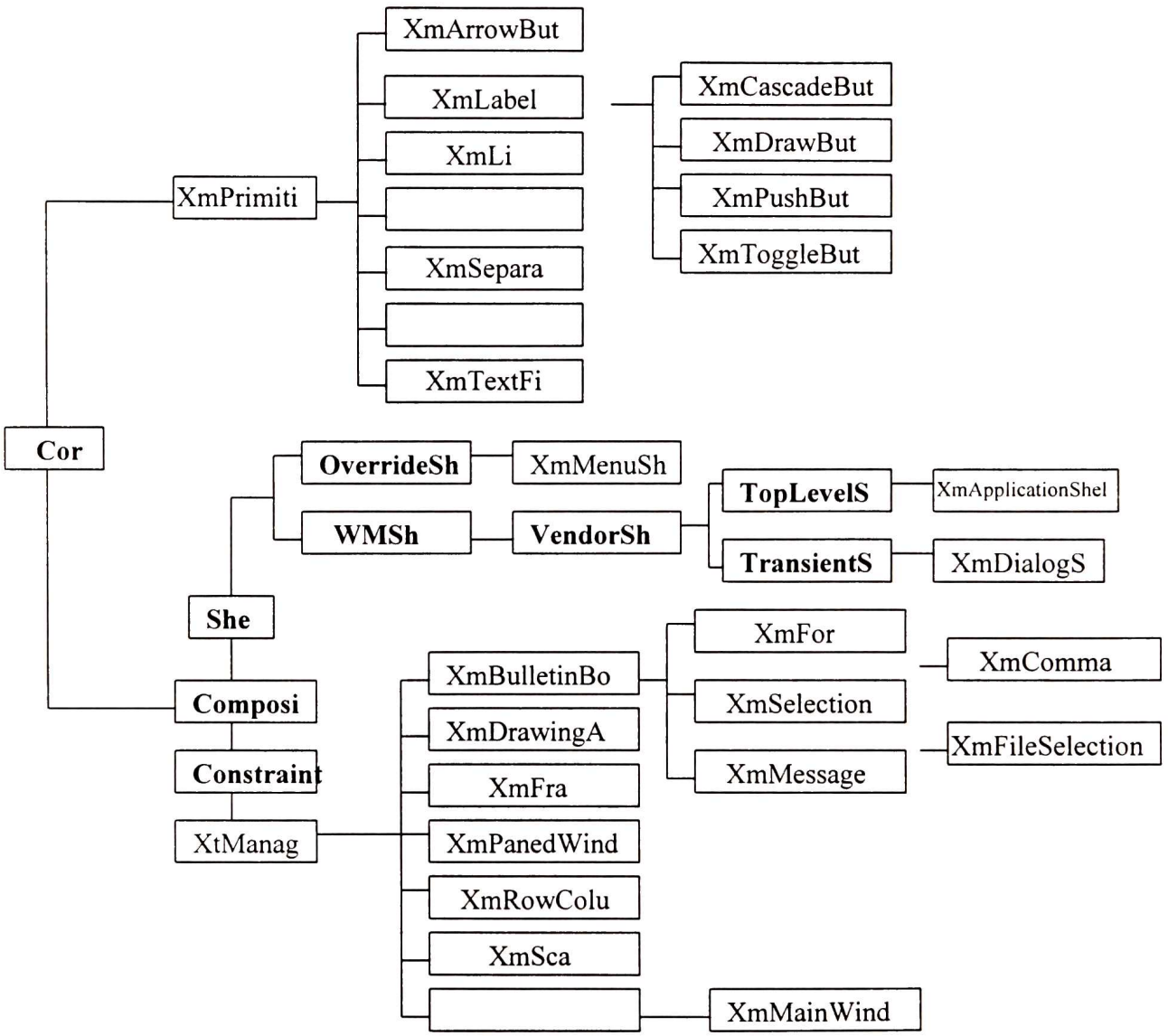


Figura 5.3.- Jerarquía de Herencia de los Widgets de Motif y Xt

5.4 CREACIÓN DEL AMBIENTE GRÁFICO EN EL SISTEMA X WINDOWS

En esta sección se describen las bases para la creación de aplicaciones gráficas en el ambiente Xwindows así como los eventos que comúnmente intervienen en estas aplicaciones.

En cualquier lenguaje de programación utilizado para la generación de gráficos, es necesario inicializar el ambiente gráfico sobre el cual se desea trabajar, en el caso de turbo C, Borland C o cualquier otro tipo de compilador, existen ya funciones específicas para este fin, estas funciones las proporciona el mismo lenguaje de programación. En el caso de Motif, una de las ironías más interesantes es que aunque es una herramienta diseñada para implementar interfaces gráficas, no tiene funciones gráficas elementales para dibujar en pantalla líneas, puntos, .etc. Si se desean crear gráficas por computadora, es necesario bajar dos niveles dentro del ambiente Xwindows y llamar directamente a las bibliotecas de X (Xlib) que son las que contienen las funciones gráficas elementales.

A continuación se presenta un diagrama en el que se observa cómo se relacionan las bibliotecas que intervienen en la programación al utilizar Motif.

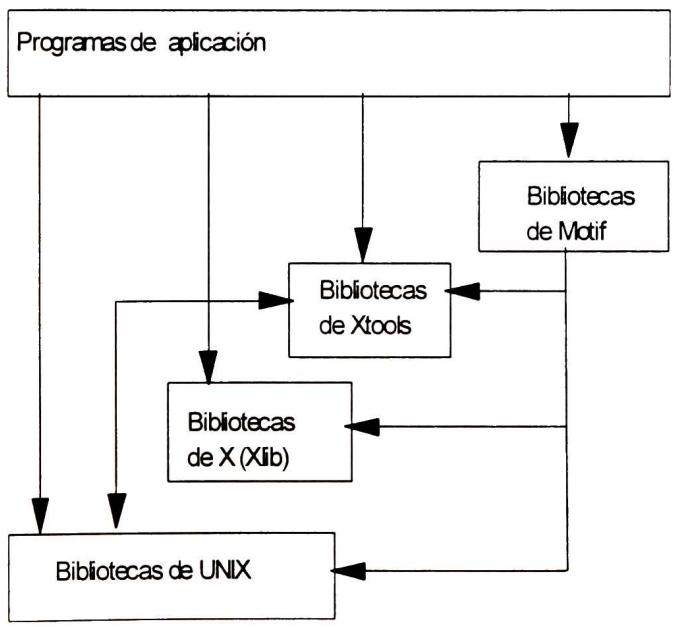


Figura 5.4.- Relaciones entre las bibliotecas utilizadas en la Programación de Motif

Las bibliotecas UNIX son las bibliotecas standard como stdio, strings y math. Las bibliotecas X (Xlib), son aquellas a través de las cuales se accesan funciones y variables de X. Las bibliotecas Xt proporcionan el acceso a las funciones y variables de los Toolkits Por último, las bibliotecas de Motif proporcionan el acceso al conjunto de widgets de Motif. Las flechas del diagrama muestran una jerarquía estricta : las bibliotecas de Motif utilizan las bibliotecas de X, Toolkits y UNIX, y las bibliotecas X utilizan solo las de UNIX. Cualquier aplicación de Motif puede accesar cualquiera de estas bibliotecas en cualquier momento. Esta accesibilidad permite a Motif omitir las funciones gráficas.

En este capítulo se describe la forma en que se realiza la programación de gráficos utilizando Motif y Xlib.

5.4.1 Conceptos básicos de graficación por computadora.

Un *display* de computadora está formado por un conjunto de puntos llamados *pixeles* colocados en un arreglo de dos dimensiones. Los pixeles sobre la pantalla dada, tienen una cierta profundidad. Una pantalla monocromática puede solamente tener pixeles de color blanco y negro, ningún otro color es posible, por lo tanto, su profundidad es 1, es decir que utiliza un bit para determinar el valor o color de cualquier pixel. Este tipo de display es comúnmente llamado bitmap. Por otro lado, un pixmap, es una imagen con una profundidad mayor o igual a 1. Un monitor típico de color, podría tener una profundidad de 8 bits o 256 colores por pixel.

Un display gráfico tiene dos sistemas de coordenadas : las coordenadas de la pantalla y las coordenadas de la ventana como se muestra en la figura 4.4. Las coordenadas de la pantalla comienzan en la esquina superior izquierda de la pantalla misma. Muchas veces, se es necesario hacer referencia al sistema de coordenadas de la ventana sobre la cual se está actualmente dibujando. Cada ventana tiene su propio sistema de coordenadas, comenzando en 0,0 en la esquina superior izquierda. Este punto se denomina *origen*. Los valores positivos de X se extienden hacia la derecha y los de Y se extienden hacia abajo.

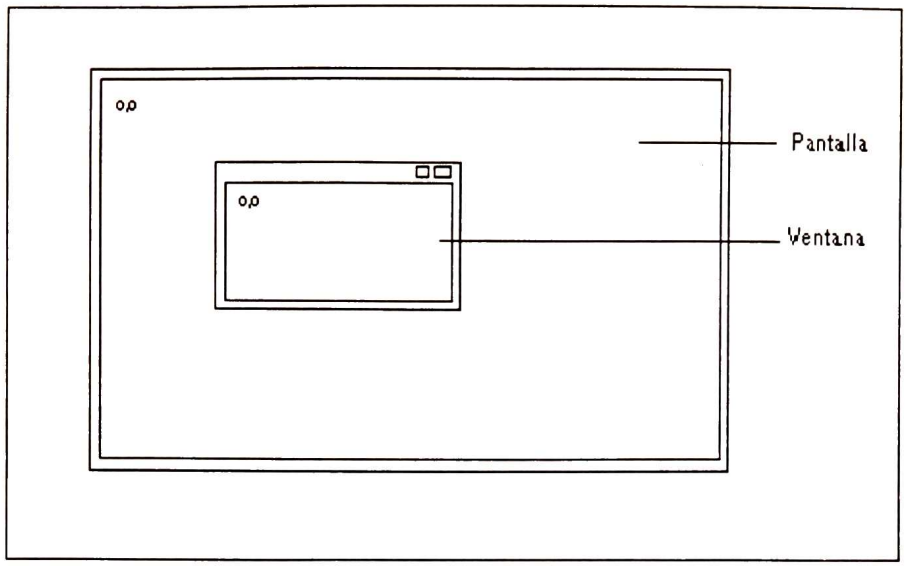


Figura 5.5.- Coordenadas de la pantalla y la ventana

5.4.2 El Contexto Gráfico de X Windows

El servidor X realiza un número de pasos antes de alterar los pixeles apropiados en una ventana o pixmap para generar una solicitud de salida gráfica como una línea con su anchura y estilo específico. El contexto gráfico es el depósito de información necesaria para completar el proceso de graficación, es una estructura de datos que contiene toda la información necesaria para controlar la apariencia de cualquier objeto dibujado en X.

En X, un *"drawable"* se refiere a una ventana o a un pixmap, ambos representan un raster de pixeles, uno sobre-pantalla (ventana) y el otro fuera-de-pantalla (pixmap). Conceptualmente, cada *"drawable"* tiene un arreglo de dos dimensiones en memoria en el cual cada lugar guarda un número de bits de información que representan un valor de pixel. El número de bits en cada valor de pixel da al *"drawable"* su profundidad.

El concepto de contexto gráfico se introdujo en X11. En la versión X10, las funciones gráficas proporcionaban todos los atributos gráficos (tales como ancho de línea, y pixel de foreground) en cada llamada. Esto era ineficiente debido a que los parámetros tenían que enviarse al servidor en cada solicitud, aún si no eran cambiados los atributos. X11 toma una ventaja más eficiente llamando a GC solo una vez y refiriéndose a él por un simple recurso ID en cada llamada a una función gráfica.

La estructura de datos que define el Contexto Gráfico, se describe a continuación:

```
typedef struct {
     int function ;
     unsigned long plane_mask ;
     unsigned long foreground ;/*color del pixel del foreground*/
     unsigned long background ;/*color del pixel del background*/
     int line_width ;
     int line_style ;/*LineSolid, LineOnOffDash, LineDoubleDash*/
     int cap_style ;
     int join_style ;
     int fill_style ;
     int fill_rule ;
     int arc_mode ;
     Pixmap title ;
     Pixmap Stipple ;
     int ts_x_origin ;
     int ts_y_origin ;
     Font font ;
     int subwindow_mode ;
     Bool graphics _exposures ;
     int clip_x_origin ;
     int clip_x_origin ;
     Pixmap clipmask ;
     int dash_offset ;
     char dashes ;
} XGCValues ;
```

Cada uno de estos elementos interviene en las características de los gráficos generados en X Windows. A continuación se describe cada uno de ellos :

Function.- La función de dibujo determina la operación lógica que coloca los pixeles fuente dentro del destino : una ventana o pixmap, las posibles funciones son las siguientes (src = fuente, dst = destino): Estas funciones se utilizan principalmente para animación.

Gxclear	0 (Se copia un 0 dentro dst para todos los puntos en src)
Gxand	src AND dst
GXandReverse	src AND NOT dst
Gxcopy	src(pixels en src reemplazan aquellos en dst)
GXanfInverted	(NOT src) AND dst
Gxnoop	dst (do nothing)
Gxxor	src XOR dst
Gxor	src OR dst
Gxnor	(NOT src) AND (NOT dst)
Gxequiv	(NOT src) XOR dst
Gxinvert	NOT dst (la inversa de dst para todos los puntos en src)
GXorReverse	src OR (NOT dst)
GXcopyInverted	NOT src
GXorInverted	(NOT src) OR dst
Gxnand	(NOT src) OR (NOT dst)
Gxset	1 (Se copia un 1 dentro de dst para todos los puntos en src)

plane mask.- Este campo controla que planos de pixmap destino se afectan en la operación de dibujo. Si el bit en la máscara de plano es 1, ese plano es modificado. Si el bit es 0, la operación de dibujo no afecta al plano.

foreground , background.- Estos campos contienen los colores del background y foreground para dibujar.

line_width.- Este campo controla el ancho de la figura que se desea dibujar (rectángulo, línea, etc).

line_style.- Este campo determina el tipo de línea. LineSolid dibuja una línea sólida. LineDoubleDash dibuja odd dashes usando el **fill_style** actual. LineOnOffDash dibuja como LineSolid pero no odd dashes.

cap_style.- Este campo afecta el trazo de los extremos de líneas y arcos. **CapButt** dibuja extremos cuadrados. **CapRound** dibuja extremos redondeados. **CapProjecting** extiende el punto extremo de la línea a una distancia igual a un medio de l ancho de la línea. **CapNotLast**. En líneas de ancho 0, no dibuja el pixel de punto final.

join_miter.- Este campo afecta la curvatura de las esquinas. **Join_Miter** une las líneas normalmente, **Join_Round** une las líneas con esquinas redondeadas y **JoinBevel** une líneas con ejes **beveled**.

fill_style.- Este campo determina los patrones del llenado de las figuras. **FillTiled**, **FillStippled** y **FillOpaqueStipled**.

fill_rule.- Determina qué partes de un polígono llenar.

arc_mode.- Este campo indica cómo se va a manejar la porción no dibujada de un arco. **ArcPieSlice** ocasiona que la parte no dibujada del arco aparezca como una rebanada de pastel. **ArcChord** ocasiona que sea tratado como un eje recto como la cuerda entre los ángulos inicial y final de un arco.

Stipples, Tiles y Estilos de llenado.- La apariencia de un área rellena, depende de su estilo de relleno o patrón de relleno. El estilo por default es el **FillSolid**, que especifica que los pixeles dentro de la figura serán pintados con el color del fondo actual. Los otros estilos como **FillStippled, FillOpaqueStippled** y **FillTiled,** usan una cuadrícula llamada *stipple* o *tile* para la operación de relleno de figuras. El término *stipple* se refiere a un pixmap de profundidad 1 que sirve como máscara a través de la cual los colores del fondo y frente se aplican durante la operación de relleno. El pixmap stipple es de tamaño fijo, generalmente de 8x8 o 16x16.

El *tile* es un arreglo de pixeles de dos dimensiones con la misma profundidad que el *drawable* para el cual fue creado el contexto gráfico. Mientras que el *stipple* tiene una profundidad de uno, un *tile* tiene la misma profundidad que el drawable y se aplica directamente al área que se va a rellenar.

Clip Mask.- Se puede llamar como máscara de corte. Por default, el servidor corta todas las salidas gráficas en los límites del drawable (ventana o pixmap). El contexto gráfico puede alterar el comportamiento por default. Si se desea dibujar dentro de sub-ventanas, se puede hacer alterando el atributo **subwindow_mode,** con el valor **IncludeInferiors** en lugar de **ClipByChildren**.

5.4.3 Funciones Gráficas Básicas.

Xlib incluye funciones para dibujar puntos, rectángulos, líneas, arcos y polígonos. Los primeros tres parámetros para todas las funciones de dibujo son los siguientes:

- Apuntador al display
- El ID del drawable

- Un GC (Contexto Gráfico) que controla la apariencia de la figura que se va dibujar.

- Display display ;
- Window window ;
- GC GC_1 ;

Los tipos de datos de estos parámetros (**Display**, **GC** y **Window**) están definidos dentro de Xlib :

XDrawPoint (display, window, GC_1, x, y) .- Dibuja un punto en las coordenadas x, y de la ventana *window*.

DrawPoints (display, window, GC_1, pt, numpt, CoordModeOrigin).- Dibuja un conjunto de puntos con una simple solicitud.

Xpoint pt Es un arreglo de puntos x,y

int numpt Es el número de puntos

CoordModeOrigin Indica que las coordenadas son relativas al origen de la ventana o del pixmap.

CoordModePrevious Se da cuando la coordenada de cada punto se da en términos

de los desplazamientos en x y y de los puntos previos.

XDrawLine (display, window, GC_1, x1, y1, x2, y2).- Dibuja una línea recta desde el punto inicial x1,y1 al punto final x2,y2.

int x1,y1 Coordenadas del extremo inicial de la línea.

int x2, y2 Coordenadas del extremo final de la línea.

XDrawSegments (display, window, GC_1, line, numsegs).- Esta función para dibujar varios segmentos de línea disjuntos utilizando los mismos atributos gráficos. Los segmentos son especificados usando la estructura especificada Xsegment :

```
typedef struct {
        short x1,y1 ;
        short x2,y2 ;
} Xsegment ;

Display display ;
Window window ;
GC  CG_1 ;
Xsegment  lines[ ] = { [100,100,200,200}, {10, 20, 35, 60}, {300,10,300,100}} ;
int numsegs = sizeof(lines) / sizeof(Xsegment) ;
```

XDrawRectangle (display, window, GC, x, y, width, height).- Dibuja un rectángulo a partir de las coordenadas x, y de su esquina superior izquierda indicando el ancho (width) y el largo (height) del rectángulo.

XFillPolygon (display, window, GC_1, points, numpoints, shape, mode) .-

```
int shape ; /*Convex, NonConvex, or Complex */
int mode ; /*CoordModeOrigin or CoordModePrevious */
```

Esta función llena un polígono con el patrón deseado. Se especifican los vértices del polígono en un arreglo de puntos (points), el argumento shape, ayuda al servidor a optimizar el algoritmo de relleno, Convex para figuras convexas. NonConvex para figuras no convexas y Complex para polígonos con ejes de intersección.

XDrawArc (display, window, GC_1, x, y, width, height, grad1, grad2).- Es similar a la función XDrawRectangle pues se pide la coordenada de la esquina superior izquierda y el largo y ancho del rectángulo dentro del cual se dibujará el arco ; los parámetros adicionales son los ángulos inicial y final del arco, esto se expresa en sesenta y cuatroavos de grado (de 0 a 360*64 equivale a un círculo)

XDrawArcs (display, window, GC_1, arcs, numarcs).- Con esta función se dibujan varios arcos.

Arcs es un arreglo de estructuras tipo Xarc, este tipo se define a continuación :

```
typedef struct
{     short x,y ;
      unsigned short width, height ;
      short     angle1, angle2 ;
}Xarc ;
```

numarcs es el número de arcos en el arreglo

5.4.3 Programación de gráficos en el ambiente Xwindow

Como en un programa normal escrito en turbo C, Pascal o Borland C, es necesario inicializar el ambiente gráfico para poder generar gráficos.

El primer paso es inicializar el contexto gráfico, esto se realiza mediante la siguiente función:

```
void setup_gc()
{
      int foreground, background ;
      XGCValues vals ;
      Arg al[10] ;
      int ac ;

/*  Para  obtener  los  valores  actuales  del  background  y
foreground*/

ac =0 ;
XtSetArg(al[ac], XmNforeground, &foreground) ; ac++ ;
XtSetArg(al[ac], XmNbackground, &background) ; ac++ ;
XtGetValues(drawing_area, al, ac) ;

/*crea el contexto gráfico*/
vals.foreground = foreground ;
vals.background = background ;

gc = XtGetGc(drawing_area, GCForeground, GCBackground, &vals) ;
}
```

La función **XtGetGc** inicializa el contexto gráfico.

Una vez inicializado el ambiente, se pueden utilizar las funciones gráficas.

A continuación se presenta un ejemplo de la utilización de funciones gráficas:

```
/*draw1.c*/

#include <Xm/Xm.h>
#include <Xm/DrawingA.h*/

XtAppContext  context ;
GC GC ;
Widget toplevel ;
Widget drawing_area ;

void setup_gc()
...

void exposureCB(w, client_data, call_data)

Widget w ;
XtPointer client_data ;
XtPointer call_data ;
{
   XDrawLine(XtDisplay(drawing_area), XtWindow(drawing_area), gc,
0, 0, 300,300) ;
}

void main(argc,argv)
int argc ;
char *argv[ ] ;
{
      Arg al[10] ;
```

```
      int ac ;
     /*crea el shell de alto nivel */
      toplevel = XtAppInitializa(&context, ▒▒ , NULL, 0, &argc,
argv, NULL, NULL,0) ;

      /*Obtiene el tamaño de la ventana*/
      ac = 0 ;
      XtSetArg(al[ac[ ], XmNheight, 300) ; ac++ ;
      XtSetArg(al[ac[ ], XmNwidth, 300) ; ac++ ;
}
```

5.5 CONCLUSIÓN DEL CAPÍTULO

Xwindows es un ambiente de desarrollo poderoso para la representación de gráficos ya que cuenta con funciones gráficas básicas contenidas en las bibliotecas de Xlib , permite el desarrollo de aplicaciones bajo el ambiente cliente-servidor., además de que está diseñado para elaborar interfaces amigables para el usuario sobre un sistema operativo UNIX, lo cual permite aprovechar los recursos que máquinas que con este sistema operativo ofrecen.

Capítulo 6

Estudio de casos

6.1 EL ENTORNO DE PROGRAMACIÓN DE DALI

El objetivo de este capítulo es presentar algunos casos ilustrativos de cómo DALI presenta en gráficas información de un problema de base de datos resuelta por LIDA. Los casos de estudio son principalmente dos: un problema de planeación de ventas y otro de análisis de ventas de petróleo.

En ambos casos los problemas se desarrollan desde el planteamiento conceptual de la base de datos, resolviendo algunas consultas con LIDA para después mostrar resultados en un ambiente gráfico con DALI.

6.1.1 **Interfaz Gráfica**

Si se ejecuta la aplicación de manera independiente a LIDA, el usuario deberá invocar al sistema tecleando el comando *dali* en la estación de trabajo. Se presenta en pantalla una ventana de inicio y al oprimir el botón derecho del ratón aparece un menú Pop-up con las siguientes opciones:

Lectura de Datos

Generación de Gráficos

Salir

3.5 CONCLUSION DEL CAPÍTULO

Dadas estas condiciones, el Sistema DALI realiza lo siguiente :

a) Identifica el objeto y el número de características relacionadas a ese objeto.

   Esto se realiza a través del procedimiento de la construcción de la tabla de datos y del esquema de homogeneidad.

b) Identifica el tipo de los datos sobre los cuales se realizará la construcción gráfica. A través del descriptor de archivos para identificar el tipo de dato que se va a graficar.

c) Determina el número de datos que se van a representar gráficamente.

d) Elige mediante una estructura CASE qué gráfica es la más apropiada para ese conjunto de datos.

CAPITULO 4

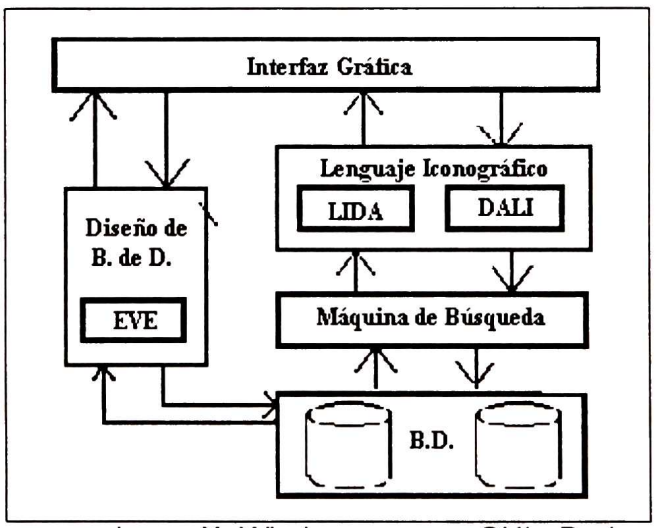
DALI: herramienta para la representación gráfica de información.

4.1. ANTECEDENTES: PROYECTOS DE PROGRAMACIÓN VISUAL

El sistema DALI forma parte del proyecto de "Programación automática a partir de Descriptores de Flujo de Información". El proyecto visual junto con otros trabajos de tesis de maestría y doctorado incluye dos tópicos principales de investigación: Visualización y Lenguajes Visuales y; Bases de Datos.

La propuesta original es a partir del lenguaje LIDA [4], el cual puede servir como lenguaje de simulación para negocios en donde DALI es un sistema incorporado que permitirá mostrar escenarios visuales de las bases de datos.

El objetivo general es un sistema de bases de datos con todo un entorno visual desde el diseño de las bases de datos hasta su explotación, incluyendo Minería de Datos.



El sistema tiene como base X Windows como GUI. Dado el objetivo general, el proyecto ha incluido los siguientes sistemas: EVE, Sistema para el diseño conceptual, lógico y físico de bases de datos, Entidad-Vínculo-Extendido [14]; LIDA, lenguaje visual que interacciona con una base de datos para resolver problemas de aplicaciones, Lenguaje Iconográfico para el Desarrollo de Aplicaciones[4]; DALI, ambiente visual para la presentación automática de gráficas de la información. HEVICOP, Herramienta Visual basada en el Paradigma 'BLOX' para construir programas en lenguaje C y C++

LIDA, desarrollado por el Dr. Sergio V. Chapa [4] es un lenguaje de flujo de datos que se describe de una forma gráfica; en donde se tienen símbolos gráficos que representan operaciones sobre los datos de entrada. Los objetos de datos, pasan a través de las líneas de flujo para entrar a módulos de transformación que son los íconos que tienen asociada una semántica a su imagen y con el uso de reglas sintácticas se disponen en una carta gráfica para conformar un flujograma.

EVE actualmente tiene dos versiones, la primera desarrollada en ambiente Windows de Microsoft y la segunda que corresponde más a este proyecto [3]. El sistema EVE permite diseñar bases de datos mediante gráficas del método extendido de Chen Entidad-Vínculo. El sistema deja un modelo lógico en Tercera Forma Normal normalizado bajo el algoritmo de Berstein [4]. Usa un esquema de descriptor de LIDA y DALI.

4.2 INTERACCIÓN DE LIDA CON DALI

El resultado de las operaciones efectuadas por LIDA sobre la Base de Datos, se registra en forma de tablas. Como se mencionó anteriormente, la información puede interpretarse mas fácilmente si se representa en forma gráfica que en una lista de datos; por lo tanto, es aquí donde el sistema DALI interviene para la representación de la información.

Una vez realizadas las operaciones indicadas en el flujograma, el usuario podrá invocar a DALI para que se le presente la información de manera gráfica.

4.2.1 Flujogramas

Los flujogramas son una herramienta gráfica que sirven para representar las funciones concernientes al manejo de la información, dentro de la estructura de una organización y representar la transformación de la información. De esta forma, los flujogramas se pueden utilizar para dos propósitos:

- Describir la gestión de una organización
- Procesar la información

Cuando los flujogramas se usan para describir la estructura operacional de la organización administrativa, el usuario representa en un lenguaje gráfico de dos dimensiones, las actividades y/o funciones que los órganos desempeñan.

El flujo de la información se representa gráficamente por flechas que consideran el tránsito de la información que fluye entre las unidades organizacionales, con la finalidad de ser transformada en cada una de ellas. De esta manera se tiene que los flujogramas son un excelente modelo para la representación de la gestión administrativa, sirviendo para responder preguntas acerca de los procedimientos administrativos.

Cada cuadro pictórico del flujograma está constituido de símbolos gráficos que representan objetos o actividades que son esenciales en cada uno de los elementos de la organización. Los símbolos son íconos que se ligan con flechas etiquetadas con el nombre de la "forma" de los documentos que contiene la información. En cada etapa se tiene una transición que se efectúa bajo una actividad o un conjunto de actividades, que suelen transformar y procesar la información para dar como resultado el segundo propósito de procesado de la información.

Los flujogramas como un modelo para describir el procesamiento de datos, consiste en la automatización de los procedimientos que manejan la información en alguna parte del sistema. El flujograma en esta parte expresa precisamente la relación funcional entre los documentos y el sistema. Esto quiere decir cómo el sistema va a efectuar las transformaciones a los documentos con un conjunto de procesos específicos.

En un flujograma, los procesos están descritos en una reseña gráfica que tiene íconos y líneas de flujo de datos. En los íconos se tiene la transformación de la información, y en los arcos, el flujo de la misma información que sale de algunos íconos para entrar a otros.

Las principales características de LIDA son:

a) Se basa en el modelo de datos relacional.

b) Es un lenguaje de flujo de datos.

c) Su programación es visual.

En el sistema LIDA, a través del descriptor de archivos los esquemas que se definen son relaciones, siendo estas los objetos de datos que principalmente fluyen en las líneas de flujo de datos. El modelo de datos relacional es el enfoque de abstracción de datos de LIDA que pretende deliberadamente omitir ciertos detalles al usuario final, manteniendo solo los que son relevantes en las aplicaciones.

El modelo de flujo de datos que tiene LIDA da un enfoque de abstracción procedural debido a que el usuario solo trata con íconos que representan procesos, además que el flujo de datos determina el control del flujo. Los arcos conectan a íconos que determinan la abstracción procedural como transformaciones de objetos de entrada a salidas. Las representaciones icónicas en LIDA, permiten invocar a los procedimientos que encapsulan operaciones, sin tomar en cuenta la representación de los objetos ni la implementación de las operaciones. Los íconos admiten argumentos que pueden ser atributos, expresiones aritméticas o expresiones booleanas. Estas últimas se organizarán como argumentos en íconos que son constructores condicionales que en su ejecución determinarán bifurcaciones en el flujo de datos mostrando una vista en paralelo como se observa en la figura 4.1.

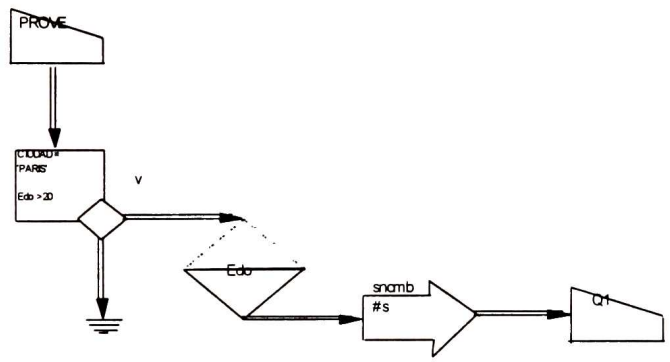


Figura 4.1.- Ejemplo de flujograma en LIDA

Esta figura es un flujograma que representa el planteamiento de un problema de consulta a una base de datos. El problema es el siguiente:

Dados los siguientes esquemas de relaciones:

PROVE(#S, SNOMB, EDO, CIUDAD)

PARTE(#P, PNOMB, COLOR, PESO, CIUDAD)

ORDEN(#S,#P, CANTIDAD).

Queremos obtener el nombre del proveedor y su número, para los proveedores que se encuentran en la ciudad de París y tengan un estado mayor que 20.

```
SQL>  SELECT SNOMB, #S
      FROM PROVE
      WHERE CIUDAD = 'Paris' AND EDO >20
      ORDER BY EDO DESC
```

En el flujograma se puede observar que de la relación Orden y Parte, fluyen simultáneamente para entrar a dos íconos de selección. En estos se seleccionan las n-adas que cumplen con ciudad igual a "París" y Estado mayor a 20. Posteriormente, como la operación de enmedio es una unión y requiere que los esquemas de relación sean iguales, solo proyectamos el atributo #P con el ícono flecha para que el ícono Union procese las dos relaciones unarias y arroje el resultado en la relación Q1.

Un rasgo distintivo de LIDA es su programación visual, que se establece en la edición y puede persistir hasta la documentación. El usuario solo trata con la disposición de íconos en la pantalla y conexiones de líneas. Considerando además, cuando la semántica del ícono lo requiere, añadir texto que son argumentos. Dadas estas características de edición gráfica, se puede decir que el sistema LIDA puede ser visto como un sistema de Diseño Asistido por Computadora para la programación.

En LIDA, la sintaxis está dada por las líneas de flujo que conectan a los íconos. Los íconos que maneja son los siguientes:

- Iconos terminales de los cuales solo entran líneas de flujo de datos e iniciales de las cuales salen. Estos son *escribe relación* y *lee relación*.
- Iconos que representan operadores binarios de los cuales siempre están esperando dos relaciones de entrada y arrojan una salida.
- Iconos unarios que aceptan una relación y dan como resultado otra relación.
- Iconos que aceptan una relación y bajo una condición establecida con una expresión booleana dividen las trayectorias de flujo en dos, una verdadera y otra falsa.
- Iconos funciones que aceptan en su entrada una relación y arrojan un valor que se mandan a un ícono rectángulo. Esta será identificada por un nombre.

4.2.2 Descripción general del sistema LIDA

Los componentes de LIDA son los siguientes:

- Módulo denominado Descriptor que permite capturar la definición de los datos y estructurar los esquemas de relación. El procedimiento es por despliegue de ventanas, manejo del cursor y actualización de datos. Este proceso es equivalente al llevado a cabo por un Lenguaje de Definición de Datos (DDL)

- Módulo capturador de datos que permite construir la base de datos. El enfoque también es de ventanas y el formato de columnas lo proporciona la definición de los datos dados por el módulo descriptor.

- Un editor gráfico para los flujogramas

- Un intérprete de código de cadena. Que es el lenguaje intermedio del sistema denominado ISBL extendido[4].

4.3 ARQUITECTURA GENERAL DEL SISTEMA DALI

Definición: DALI es un sistema de visualización de datos que tiene como objetivo representar de una manera gráfica, los resultados obtenidos por las operaciones realizadas en LIDA, se considera una herramienta visual para la representación gráfica de datos. Está desarrollada bajo un ambiente de XWindows que proporciona facilidades gráficas.

El sistema DALI está integrado por los siguientes módulos:

a) Módulo para la Definición y Manejo de archivos.

b) Analizador de Datos

c) Constructor gráfico

La interrelación de estos módulos se observa en la figura 4.2.

a) El módulo para la Definición de Archivos crea descriptores en donde se indica el nombre, tipo y tamaño de cada campo que compone el archivo. Una vez definido el archivo, el sistema permite introducir información. También permite leer y escribir desde y hacia el disco los archivos que se desean representar gráficamente.

b) El módulo de Clasificación y Análisis verifica el tipo de los datos que se desean visualizar, el número de campos o características del archivo, crea la Tabla de Distribución y la Matriz de Datos que se describen en el capítulo 3. Una vez creada la matriz, se analizan los datos para definir el tipo de gráfica más apropiado para su representación. El analizador está integrado por una serie de procedimientos que realizan el análisis que se describió en el capítulo 3.

c) El Módulo de Construcción Gráfica crea una estructura de datos para la representación gráfica denominada *Graff,* ésta contiene toda la información que generó el módulo de Análisis referente a la gráfica más apropiada. Toma de esta estructura la información y construye la gráfica especificada por el análisis utilizando las funciones de Xlib y Motif para su despliegue en la pantalla.

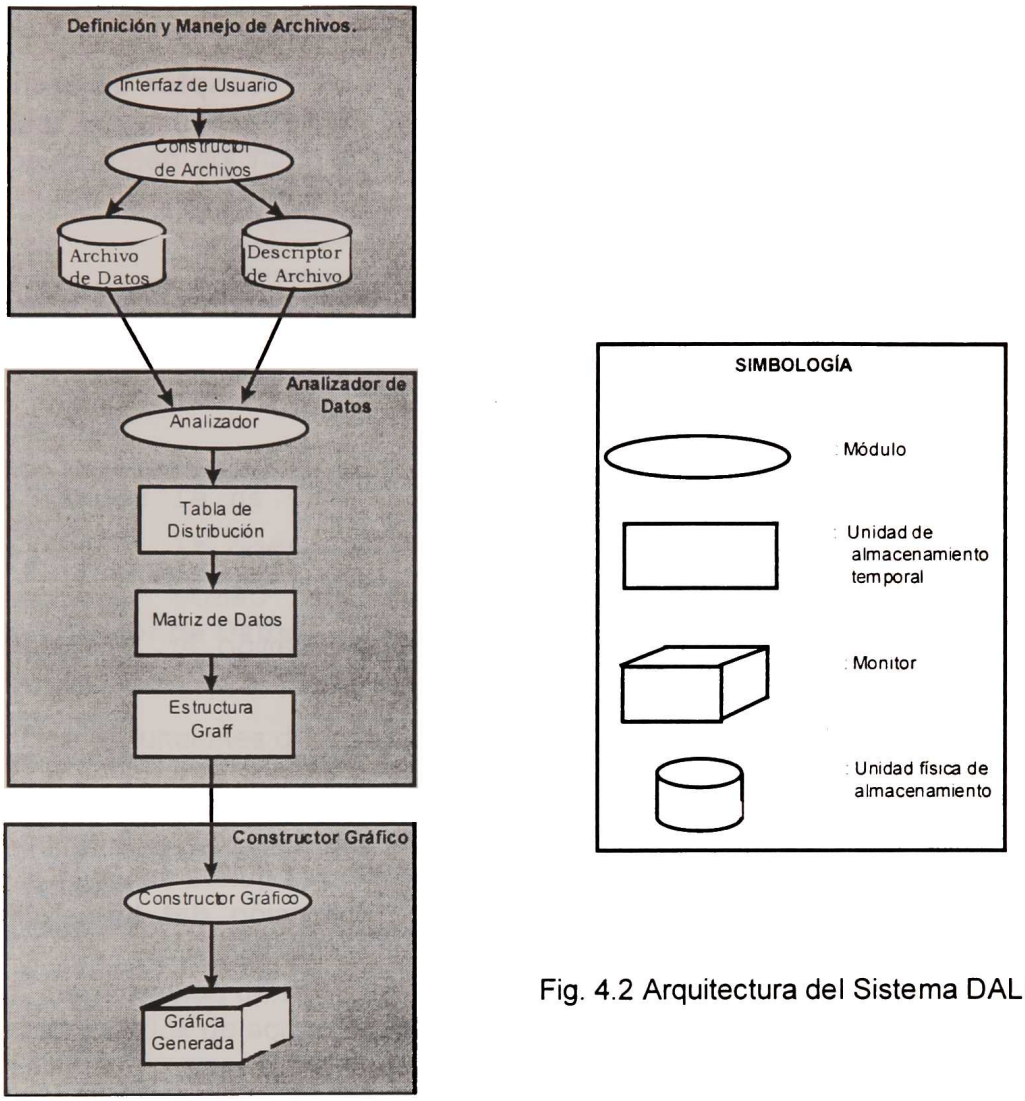


Fig. 4.2 Arquitectura del Sistema DALI

4.3.1 Definición de la Base de Datos

Debido a que aún continúa en desarrollo como proyecto de tesis la construcción del sistema LIDA en ambiente Xwindows, el sistema DALI permite definir archivos de datos para que a partir de ellos se lleve a cabo la construcción gráfica de manera independiente. En el momento en que se integre LIDA y DALI, la información será leída desde las tablas que genera LIDA.

Para definir los archivos de datos que serán representados, el sistema DALI presenta un Menú de Opciones en donde el usuario podrá elegir el proceso que desea realizar. Este menú se describe en el siguiente capítulo.

El módulo de definición de la base de datos genera un **descriptor de archivos** en donde se guardan las características de cada uno de los campos del archivo, se maneja un arreglo de descriptores en donde se encuentran los datos de todos los campos.

A continuación se presenta la estructura **descriptor** en donde se lee y escribe la información de los campos:

```
struct Descriptor {
     int posición,  /* Posición del campo en el registro */
   enum Tipo_camp Tipo,      /* Tipo de campo */
    char nombre[TAM_NOM],    /* Nombre del campo */
   int tam,                /* Tamaño del campo en bytes */
 } Campos [MAX_CAMPS]
```

Una vez definidos los campos con sus correspondientes tipos y tamaños, se introducen los datos a través de una de las opciones del menú. Estos se van ordenando en una lista doblemente ligada y posteriormente, en el caso de que quiera salvarse el archivo en disco, la información se guarda en un archivo con extensión **.dat**. La estructura de la lista ligada se muestra a continuación:

```
struct Db_Tipo {
  char *Datos;     /* apuntador a los datos */
  struct Db_Tipo *Ant;  /* apuntador al reg. anterior */
  struct Db_Tipo *Sig;  /* apuntador al siguiente reg.*/
} *Primero, *Ultimo, *cur_rec;
```

Se crea un archivo para el descriptor de cada .dat con extensión **.def** y este también se graba en disco.

En el momento en que se desee realizar el análisis de un archivo, se realiza el proceso inverso: se lee del archivo con extensión **.def** la información correspondiente al tipo y tamaño de los campos se guarda en el arreglo de estructuras **Campos** y los datos contenidos en el archivo con extensión **.dat** se van colocando en la lista ligada para que a partir de ahí puedan ser leídos en el proceso de Análisis. La estructura en donde quedará guardada la información del descriptor se muestra en la figura 4.3.

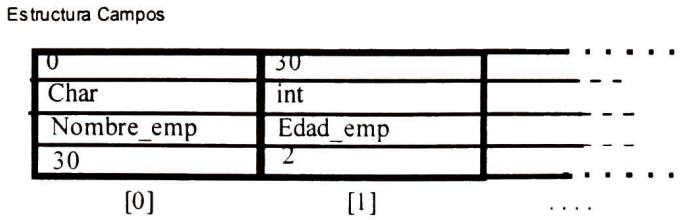


Figura 4.3.- Arreglo Campos (Guarda datos del descriptor)

Los pasos para la realización del análisis se describen en la siguiente sección.

4.3.2 Analizador de Datos

Las principales funciones del analizador son:

a) Creación de la tabla de distribución y verificación de la homogeneidad de los datos.

b) Creación de la matriz de datos.

c) Identificación de la gráfica más adecuada para la representación de los datos analizados de acuerdo al número de características encontradas.

d) Creación de la estructura Graff.

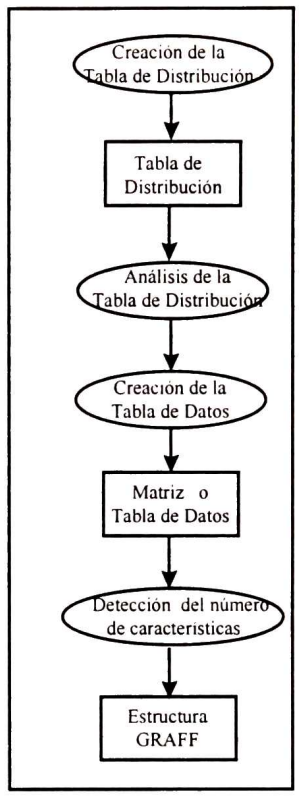


Figura 4.4.- Procesos del Análisis de Datos

a) **Creación de la Tabla de Distribución.** Este proceso consiste en clasificar los campos de acuerdo a su tipo y asignarles a cada uno de ellos una nomenclatura ("O", "Q", "X") para determinar si son datos ordenados, reordenables o de tipo texto. Los campos se leen del descriptor de archivos, se clasifican y se almacenan en una estructura de datos a la que se denomina Tabla de Distribución la cual se describió en el capítulo 3.

b) Una vez construida la tabla de distribución, se procede a realizar la verificación de los datos, es decir, verificar si en el conjunto de datos existe un punto en común el cual tenga relación con el resto de los datos, esto es lo que se denomina Esquema de Homogeneidad (Capítulo 3). A partir de aquí, se determina el **objeto** y las **características** del conjunto de datos.

c) La matriz de datos es un arreglo de apuntadores de tipo char en donde se van a almacenar temporalmente los datos del archivo. Se almacenan los datos **objeto** y sus correspondientes datos **característica** de acuerdo a la información contenida en la Tabla de Distribución.

d) En la primera fila se colocan los datos **objeto** y en las restantes, los datos **característica**.

e) Se determina el número de características de la matriz y el tipo de las mismas y esta información se considera para la creación de la estructura Graff.

f) Se crea la estructura Graff la cual contiene toda la información que dará origen a una gráfica.

Creación de la Tabla de Distribución y Verificación la homogeneidad de los datos

Para la creación de la Tabla de Distribución se siguen los siguientes procesos:

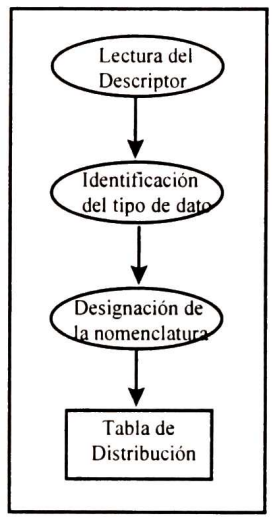


Figura 4.5.- Creación de la Tabla de Distribución

Lectura del descriptor.- Se lee la información contenida en el archivo con extensión **.def** y se va almacenando en el arreglo de registros denominado "Campos", como se indica en la sección anterior.

Identificación del tipo de dato.- Se comienza a analizar el tipo de cada uno de los campos para asignarle su correspondiente nomenclatura.

Designación de la Nomenclatura.- Se asigna la nomenclatura dependiendo del tipo de dato:

X si es de tipo texto

Q si es numérico

O si es tipo fecha

Se escribe en la estructura Reg_Tabdis el nombre del campo, su nomenclatura y el valor de dimensión.

```
struct Registro {
     char Elemento[40];
     int Dimension;
     char Nomen[50][3];
};

struct Registro Reg_TabDis;
```

`Elemento` .- Nombre del campo.

`Dimensión`.- Número de elementos de este campo.

`Nomen` .- Arreglo de nomenclaturas.

Se abre el archivo TabDis y se almacenan ahí los registros de Reg_Tabdis.

Creación de la tabla o matriz de datos

Una vez creada la tabla de distribución se realizan los procesos que a continuación se indican:

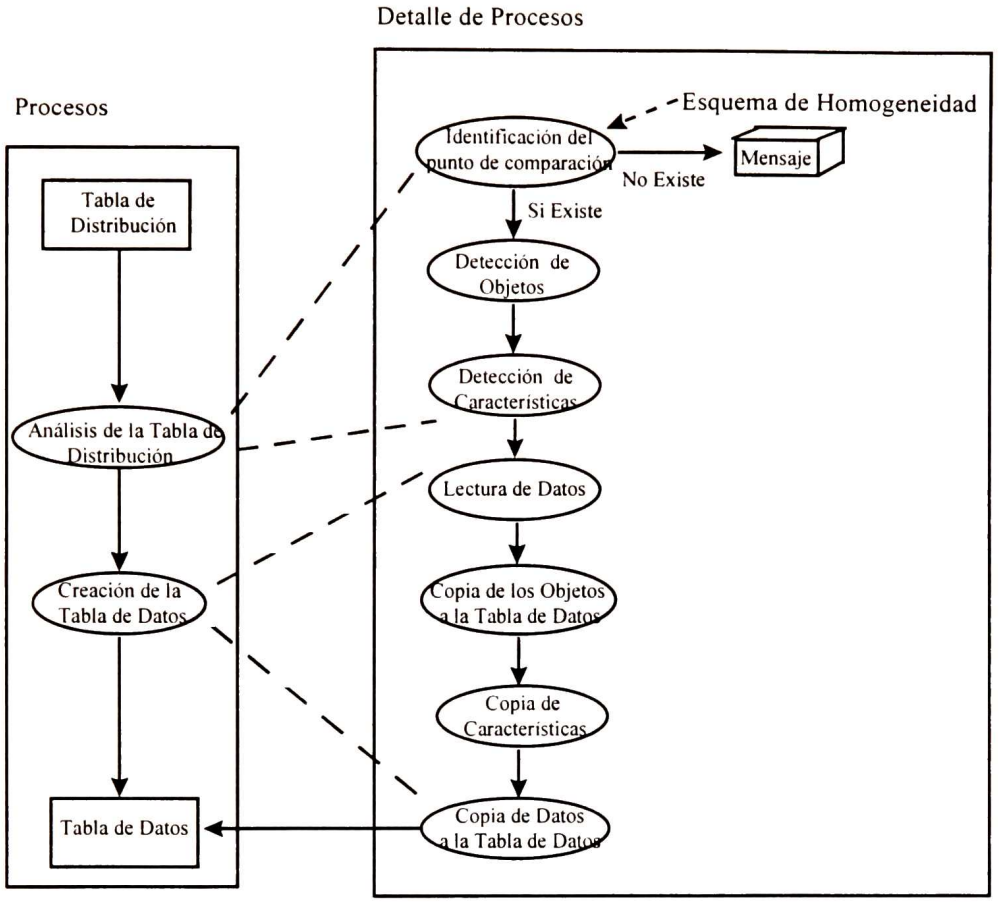


Figura 4.6.- Construcción de Tabla de Datos

Esquema de Homogeneidad.- Es aquí donde se debe identificar si existe un punto de comparación para definir si los datos son homogéneos y proseguir el análisis de la información.

Detección de objetos y características.- Los datos se leen de la lista ligada y se van guardando en vectores de tipo apuntador a char.

Se verifican todos los datos para identificar el dato **objeto** y los datos **característica**. Esto se realiza identificando qué datos se repiten:

Si todos los datos de un campo son diferentes entonces este campo se considera el campo **objeto**.

Si algún dato de un campo se repite entonces este campo se considera una **característica**.

Lectura de datos e integración a la Tabla de Datos.- Los vectores de datos se integran en sus correspondientes renglones a la matriz de datos de acuerdo a su tipo: **objeto** o **característica.**

Obtención del número de características.- Este procedimiento consiste en determinar el número de campos del registro relacionados al campo llave, para que así se obtenga el número de características.

Creación de la estructura graff.- Una vez determinado el número de características se crea la estructura graf cuya definición se presenta a continuación.

```
typedef struct {
     int g_id; /* Número de identificación de la gráfica */
      int tipo; /* Tipo de gráfica */
     int ncarac; /* Número de características */
     int sep; /* si la estructura pastel esta separada, en
            3d o   normal */
     int x;    /* Posición x en la pantalla  */
     int y;    /* Posición y en la pantalla  */
     char tit[30];     /* Título de la gráfica */
     char stit1[30];
     char stit2[30];
     char nomy[30];    /* Nombre del eje y */
     char nomx[30];    /* Nombre del eje y */
     char ser1[30];    /* Nombre de la serie 1 */
     char ser2[30];    /* Nombre de la serie 2 */
     char ser3[30];    /* Nombre de la serie 3 */
     char ser4[30];    /* Nombre de la serie 4 */
     char unid1[30];
     char unid2[30];
     char unid3[30];
     char unid4[30];
     int ndatx;          /* Número de datos en el eje x */
     char *arrx[100];  /* Datos del eje y */
     char *arr1[100]; /* Datos del eje y */
     char *arr2[100]; /* Datos del eje y */
```

```
        char *arr3[100]; /* Datos del eje y */
        char *arr4[100]; /* Datos del eje y */
        int tamdx;          /* Tamaño en bytes de los datos en
x */
        int tamd1;     /* Tamaño en bytes de los datos en x */
        int tamd2;     /* Tamaño en bytes de los datos en x */
        int tamd3;     /* Tamaño en bytes de los datos en x */
        int tamd4;     /* Tamaño en bytes de los datos en x */
        int tamcampX;
        int tamcamp1;
        int tamcamp2;
        int tamcamp3;
        int tamcamp4;
        } graf_ ,  *graf;
```

4.3.3 Constructor Gráfico

El constructor gráfico está integrado por los siguientes procesos:

a) Determinación de la gráfica más apropiada
b) Actualización de la estructura Graff
c) Rutinas para la creación del ambiente gráfico.
d) Rutinas para la generación de gráficas.

a) Determinación de la gráfica más apropiada

Un elemento fundamental en el análisis de los datos es el número de características de la matriz de datos y el tipo de los datos que se desean graficar. Una vez determinado el tipo de datos y habiendo obtenido el número de características, se realiza el procedimiento de definición de la gráfica apropiada a través de la rutina Objetos_reordenables que se basa principalmente en asignar la función gráfica correspondiente al conjunto de datos analizado dependiendo del tipo de datos y el número de **características** asociadas al **objeto.**

b) Llenado de la estructura graff

En esta estructura se guardan todas las características de la gráfica como son:

- Número de características o variables
- Título de eje x
- Título de eje y
- Nombre de etiquetas
- Datos en x
- Datos en y

que serán necesarias en el momento de generar la gráfica.

c) **Rutinas para la creación del ambiente gráfico**

El sistema DALI fue desarrollado bajo el ambiente Xwindows en una estación de trabajo DEC. Para la interfaz gráfica se utilizaron funciones de las bibliotecas de Motif y Xlib las cuales permiten el manejo de ventanas y objetos visuales como push button, menus, scroll bars, etc., que se describen en el siguiente capítulo.

Las gráficas son generadas dentro de ventanas o **formas** las cuales contienen ciertas características y están en función de los **eventos** que se presentan a su alrededor. Cada forma tendrá un contexto gráfico el cual deberá ser inicializado antes de generar la gráfica.

Para la creación de una forma que va a contener la gráfica se utiliza la siguiente función

```
shell[indi]    =    XtAppCreateShell(sheln[indi],"Grafica",
                    applicationShellWidgetClass,
                    XtDisplay(toplevel), al, ac);
```

Para la inicialización del contexto gráfico se utiliza la función

```
drawing_area[indi]              =XmCreateDrawingArea(formg[indi],
                    dwn[indi],    al,ac);
get_colors(drawing_area[indi]);
```

```
setup_gc(drawing_area[indi]);
```

d) Rutinas para la generación de gráficas

Dentro de la estructura de control CASE se encuentran las funciones gráficas que reciben como parámetro la estructura de datos Graff. La función que genera las gráficas se denomina **genera_g**.

A continuación se enlistan las diferentes gráficas que se generan de acuerdo a los parámetros que el análisis toma en cuenta:

Tabla 4.1 Correspondencia del número de características con su gráfica representativa

Número de Características	Número de elementos	Ausencia	Progresión Geométrica	Bandera	Tipo Gráfica
1	<=12	No			Proporcional
		Sí			Scatter Plot
		No	No		Barras
			Sí		Gráfica Logarítmica
	>12	Sí			Scatter Plot
		No	No		Gráfica de línea
			Sí		Gráfica Logarítmica
2	<12	Sí			Scatter Plot
		No	No		Barras
			Sí		Gráfica Logarítmica
	>12	Sí			Scatter plot
		No	No		Línea
			Sí		Gráfica Logarítmica
3	<=12			No	Barras
	>12			No	Gráfica de Línea
4 o 5					Matriz de Permutación
>5					Cara de Chernoff

Número de características.- Es el número de campos relacionados a un campo objeto, es decir, el número de atributos de un campo llave.

Número de elementos .- Es el número de renglones de la matriz de datos.

Ausencia.- Se comprueba si existe ausencia de datos, es decir, si algún valor de algún campo no es conocido.

Progresión Geométrica.- Este parámetro indica si los valores que se encuentran registrados en la matriz de datos siguen una progresión geométrica.

Con base en los anteriores conceptos es como se elige la gráfica apropiada para el conjunto de datos que se encuentra almacenado en la matriz o tabla de datos.

Una vez determinado el tipo de la gráfica a generarse, se ejecuta el procedimiento de la gráfica correspondiente. El parámetro principal para cada uno de estos procedimientos es la estructura **Graff** la cual contiene toda la información necesaria para la generación de la gráfica.

4.4 CONCLUSIONES AL CAPÍTULO

- El sistema DALI es la herramienta que genera gráficas estadísticas y multivariadas para la representación de datos.

- El siguiente paso es unir esta herramienta al sistema LIDA que actualmente se encuentra en desarrollo bajo un ambiente Xwindows.

- Como extensión al presente trabajo se pueden desarrollar rutinas para gráficas estadísticas en 3D.

Capítulo 5

SISTEMA DALI Y SU AMBIENTE GRÁFICO XWINDOWS

5.1 ANTECEDENTES

Desde Macintosh de Apple y PCs de MS-DOS hasta las estaciones de trabajo gráficas como Sun y DEC Station, ha ido cambiado la manera en que el usuario interactúa con las computadoras. Las capacidades gráficas han hecho posible interfaces gráficas para usuarios como la Macintosh User Interface y Microsoft Windows. Con estas interfaces, en lugar de ejecutar comandos en línea, se puede utilizar el dispositivo apuntador mouse para ejecutar programas, editar, copiar y borrar archivos. Adicionalmente, las interfaces gráficas dividen la pantalla de despliegue físico en regiones (generalmente rectangulares) denominadas *ventanas* donde aparece la salida de diferentes aplicaciones.

Para ambientes con sistema operativo UNIX, la herramienta que permite la generación de aplicaciones a través del manejo de ventanas es el sistema X Windows[15].

X fue desarrollado conjuntamente por el proyecto Athena de MIT y la Corporación Digital Equipment con contribuciones de muchas otras compañías. Fue dirigido por Robert Scheifler y alumnos del MIT. La versión sobre la que se desarrolló el sistema DALI es la versión X11 registrada en enero de 1990. Los principales conceptos que se manejan en este ambiente se describen brevemente a continuación.

5.2 CONCEPTOS GENERALES DE X WINDOWS

5.2.1 Displays y pantallas

La principal característica de X es que es un sistema de ventanas para despliegues gráficos de mapas de bits. En X windows, un *display* se define como una estación de trabajo constituida por un teclado, un dispositivo apuntador (mouse) y una o más pantallas. Las pantallas múltiples pueden trabajar juntas.

5.2.2 El Modelo Cliente-Servidor

El sistema X es un sistema orientado a red. Una aplicación no necesariamente puede correr en el mismo sistema que soporta al display. Mientras que muchas aplicaciones se ejecutan localmente, otras pueden ejecutarse en otras máquinas enviando solicitudes a través de la red para un despliegue particular y recibir eventos del teclado y del ratón desde el sistema que controla el display.

El ***servidor*** proporciona un servicio a las solicitudes del ***cliente***. Generalmente, los clientes se comunican con el servidor a través de la red y el cliente y el servidor intercambian datos usando un protocolo entendible por ambas estaciones de trabajo. Así como un servidor de archivos almacena archivos y permite que los clientes los accesen y manipulen, el servidor de display X ofrece servicios de despliegue gráfico para clientes que envían las solicitudes del protocolo X al servidor.

El servidor actúa como intermediario entre los programas de usuario que corren en cualquier sistema ya sea local o remoto y los recursos del sistema local. El servidor desarrolla las siguientes tareas:

Permite el acceso al display por múltiples clientes.

- Interpreta los mensajes de red de los clientes.
- Pasa la entrada del usuario a los clientes enviando mensajes de red.

Mantiene estructuras de datos complejas incluyendo ventanas, cursores, fonts y contextos gráficos.

En contraste con los servidores de bases de datos o servidores de archivos (los cuales por lo general son procesos que se ejecutan en máquinas remotas) el servidor de display X es un proceso que se ejecuta en la estación de trabajo mientras que los clientes pueden estar corriendo en computadoras remotas.

A continuación se presenta un esquema del modelo **cliente/servidor**.

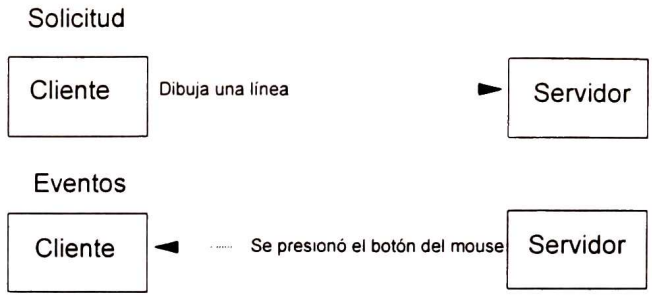


Figura 5.1.- Representación del modelo **cliente/servidor**

5.2.3 Manejo de Eventos

El servidor X considera cualquier cosa que se realice con el teclado o con el mouse como **eventos** que serán reportados a los clientes. Cuando se corren aplicaciones en X, todo en la pantalla aparece dentro de ventanas y cada ventana está asociada con un cliente específico. Cuando se presiona y se suelta el botón de mouse, el servidor X envía estos eventos de entrada al cliente que originalmente creó la ventana que contiene el apuntador del mouse. Para los eventos del teclado la tecla presionada siempre pertenecerá a la ventana actual designada.

El servidor X envía otro tipo de evento a los clientes, un **evento expose.** Este evento indica al cliente si algo sucede en su ventana.

5.2.4 Las capas del Sistema X

Protocolo X: El lenguaje de máquina de X

El protocolo define la manera en que el servidor y el cliente van a intercambiar información y como va a ser interpretada. En el protocolo X, los datos se intercambian en forma asíncrona sobre una comunicación de dos sentidos que permite la transmisión en ocho bytes, esta capa se considera el lenguaje de máquina de X.

Xlib: Lenguaje ensamblador del sistema X

El sistema X Windows viene con una biblioteca de rutinas en C denominada *Xlib*. Xlib permite el acceso al protocolo X a través de más de 300 rutinas de utilería. Si el protocolo X se considera el lenguaje máquina de X, entonces Xlib es el lenguaje ensamblador, es decir, a través de él se pueden ejecutar instrucciones del protocolo X.

X Toolkits: Lenguajes de alto nivel del sistema X

Este es un conjunto de rutinas que permite crear botones, listas y menús las cuales pueden utilizarse para construir una interfaz gráfica. El sistema X contiene las herramientas *X Toolkits Intrinsics*, el cual utiliza la implementación orientada a objetos para crear bloques de construcción básicos llamados **widgets**. Existen otros toolkits como Motif creado por la Open Software Fundation y OPEN LOOK creado por SUN los cuales usan un alto nivel de abstracción. La herramienta Motif está construída sobre X Toolkits Intrinsics y se consideran como el lenguaje de alto nivel del sistema X, ya que es a través del cual el usuario puede desarrollar aplicaciones en el ambiente Xwindows.

La siguiente figura muestra la estructura general de una aplicación X. La aplicación primeramente llama rutinas desde un toolkit. El toolkit llama rutinas desde Xt Intrinsics las cuales a su vez llaman a Xlib. La aplicación puede también hacer llamadas directamente a rutinas de Xlib para generar salidas de texto y gráficas en una ventana.

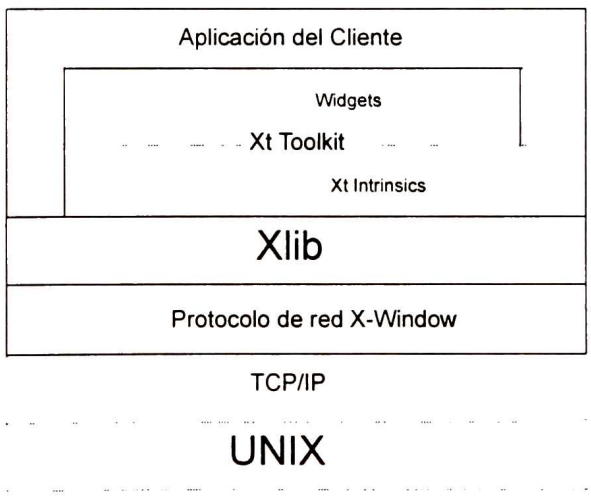


Figura 5.2.- Capas del Sistema Xwindows

Para el desarrollo del sistema DALI, se utilizaron las bibliotecas de Xlib para la generación de gráficas, manejo de color y texto.

Para la creación de la interface gráfica (ventanas, menús, botones, scroll bars, etc.), se utilizó el Toolkit MOTIF.

5.3 MOTIF Y SU NATURALEZA ORIENTADA A OBJETOS

Esta es una herramienta que permite a los programadores diseñar conjuntos de widgets para crear la interfaz gráfica en ambiente UNIX.

El conjunto de widgets es típico: contiene los widgets de scroll bar, de botón, de menu, de texto, etc. Un widget en MOTIF es equivalente a un objeto gráfico en Visual Basic.

Por su diseño, el widget aparece muy orientado a objetos para el programador. Debido a que la programación de estas bibliotecas es en lenguaje C, no es completamente orientado a objetos. Este toolkit proporciona las ventajas de la programación orientada a objetos sin que el programador requiera aprender un nuevo lenguaje.

En la programación de MOTIF, cada objeto de la interface de usuario (o widget) está controlado por un conjunto de variables llamadas **recursos.** Al cambiar los recursos, se puede controlar la apariencia y comportamiento del widget. Al leer los recursos, se puede conocer el estado del widget. El widget también puede enviar mensajes, conocidos como **callbacks** cuando quiera comunicarse con el código restante.

El ambiente de programación orientada a objetos soporta jerarquía de objetos. Para construir un nuevo objeto, se usa y se agrega uno existente en un proceso llamado **herencia.** El nuevo objeto puede hacer cualquier cosa que el objeto original haga así como cualquier otro proceso que se le adicione. Se pueden combinar objetos existentes dentro de un nuevo objeto. Similarmente, Motif usa la herencia para construir widgets como antecesores de otros widgets o fuera de grupos de widgets. Todos los widgets de MOTIF usan la herencia internamente.

Cada widget puede mandar mensajes de salida llamados **callbacks** para las funciones del programa cuando un usuario manipula un widget.

Existen dos grandes ventajas para manejar objetos de interface de usuario en esta forma. Primero, se puede modificar la apariencia y comportamiento del widget. Segundo, el widget maneja todo lo referente al manejo de eventos de bajo nivel. Motif permite que el usuario desarrolle interfaces de usuario gráficas de una manera más sencilla.

Algunos de los widgets más comunes en la programación de la interfaz gráfica son los siguientes:

XmForm.- Manejador de widgets que funge como contenedor de widgets hijos.

XmPushButton .- Permite al usuario ejecutar un comando tecleando un botón.

XmText .- Proporciona capacidades para la edición de texto.

XmLabel .- Despliega una cadena de caracteres.

XmList .- Permite al usuario elegir una o múltiples opciones desde una lista.

Para la creación de estos widgets se utiliza la función:

XtCreateWidget(

String nombre,

WidgetClass class,

Widget padre,

ArgList args,

Cardinal num_args);

Donde

nombre es el nombre del widget

class es la clase del widget

padre el padre del widget que se va a crear

args lista de argumentos

num_args número de argumentos

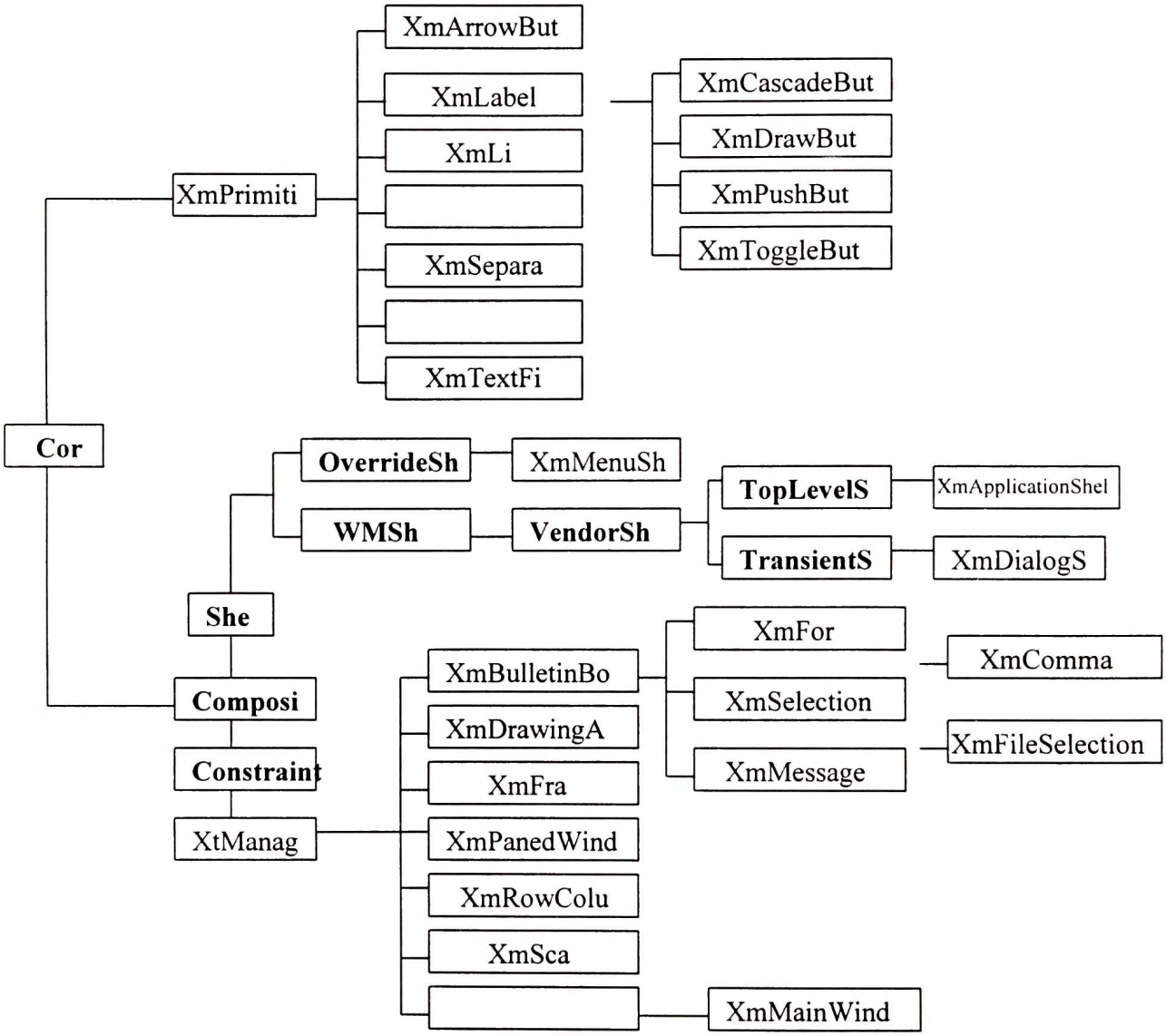


Figura 5.3.- Jerarquía de Herencia de los Widgets de Motif y Xt

5.4 CREACIÓN DEL AMBIENTE GRÁFICO EN EL SISTEMA X WINDOWS

En esta sección se describen las bases para la creación de aplicaciones gráficas en el ambiente Xwindows así como los eventos que comúnmente intervienen en estas aplicaciones.

En cualquier lenguaje de programación utilizado para la generación de gráficos, es necesario inicializar el ambiente gráfico sobre el cual se desea trabajar, en el caso de turbo C, Borland C o cualquier otro tipo de compilador, existen ya funciones específicas para este fin, estas funciones las proporciona el mismo lenguaje de programación. En el caso de Motif, una de las ironías más interesantes es que aunque es una herramienta diseñada para implementar interfaces gráficas, no tiene funciones gráficas elementales para dibujar en pantalla líneas, puntos, .etc. Si se desean crear gráficas por computadora, es necesario bajar dos niveles dentro del ambiente Xwindows y llamar directamente a las bibliotecas de X (Xlib) que son las que contienen las funciones gráficas elementales.

A continuación se presenta un diagrama en el que se observa cómo se relacionan las bibliotecas que intervienen en la programación al utilizar Motif.

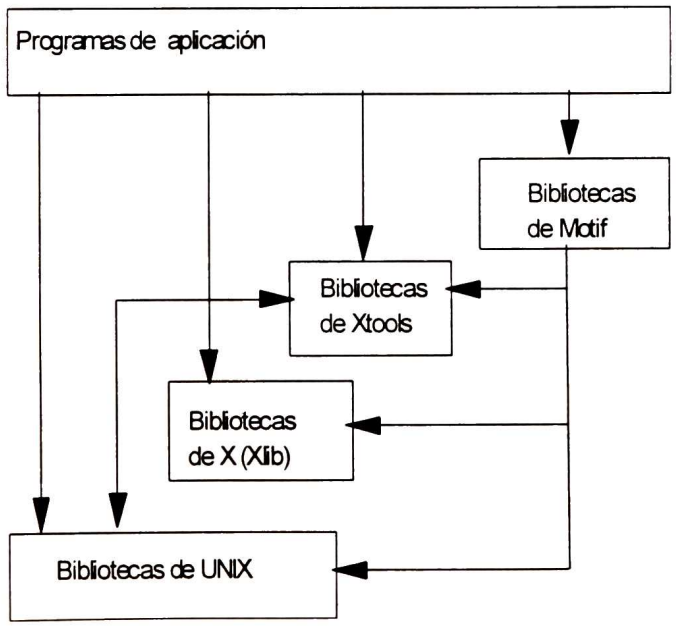


Figura 5.4.- Relaciones entre las bibliotecas utilizadas en la Programación de Motif

Las bibliotecas UNIX son las bibliotecas standard como stdio, strings y math. Las bibliotecas X (Xlib), son aquellas a través de las cuales se accesan funciones y variables de X. Las bibliotecas Xt proporcionan el acceso a las funciones y variables de los Toolkits Por último, las bibliotecas de Motif proporcionan el acceso al conjunto de widgets de Motif. Las flechas del diagrama muestran una jerarquía estricta : las bibliotecas de Motif utilizan las bibliotecas de X, Toolkits y UNIX, y las bibliotecas X utilizan solo las de UNIX. Cualquier aplicación de Motif puede accesar cualquiera de estas bibliotecas en cualquier momento. Esta accesibilidad permite a Motif omitir las funciones gráficas.

En este capítulo se describe la forma en que se realiza la programación de gráficos utilizando Motif y Xlib.

5.4.1 Conceptos básicos de graficación por computadora.

Un *display* de computadora está formado por un conjunto de puntos llamados *pixeles* colocados en un arreglo de dos dimensiones. Los pixeles sobre la pantalla dada, tienen una cierta profundidad. Una pantalla monocromática puede solamente tener pixeles de color blanco y negro, ningún otro color es posible, por lo tanto, su profundidad es 1, es decir que utiliza un bit para determinar el valor o color de cualquier pixel. Este tipo de display es comúnmente llamado bitmap. Por otro lado, un pixmap, es una imagen con una profundidad mayor o igual a 1. Un monitor típico de color, podría tener una profundidad de 8 bits o 256 colores por pixel.

Un display gráfico tiene dos sistemas de coordenadas : las coordenadas de la pantalla y las coordenadas de la ventana como se muestra en la figura 4.4. Las coordenadas de la pantalla comienzan en la esquina superior izquierda de la pantalla misma. Muchas veces, se es necesario hacer referencia al sistema de coordenadas de la ventana sobre la cual se está actualmente dibujando. Cada ventana tiene su propio sistema de coordenadas, comenzando en 0,0 en la esquina superior izquierda. Este punto se denomina *origen*. Los valores positivos de X se extienden hacia la derecha y los de Y se extienden hacia abajo.

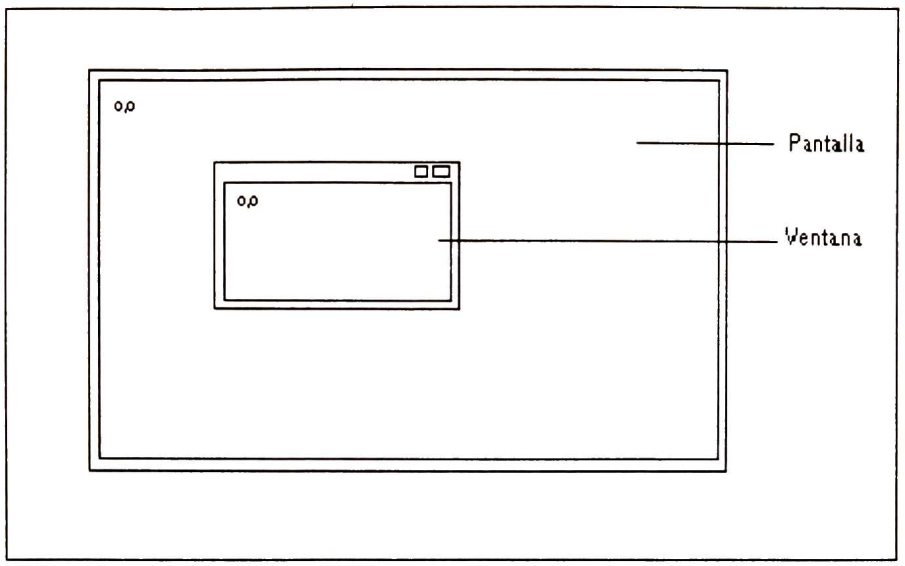


Figura 5.5.- Coordenadas de la pantalla y la ventana

5.4.2 El Contexto Gráfico de X Windows

El servidor X realiza un número de pasos antes de alterar los pixeles apropiados en una ventana o pixmap para generar una solicitud de salida gráfica como una línea con su anchura y estilo específico. El contexto gráfico es el depósito de información necesaria para completar el proceso de graficación, es una estructura de datos que contiene toda la información necesaria para controlar la apariencia de cualquier objeto dibujado en X.

En X, un *"drawable"* se refiere a una ventana o a un pixmap, ambos representan un raster de pixeles, uno sobre-pantalla (ventana) y el otro fuera-de-pantalla (pixmap). Conceptualmente, cada *"drawable"* tiene un arreglo de dos dimensiones en memoria en el cual cada lugar guarda un número de bits de información que representan un valor de pixel. El número de bits en cada valor de pixel da al *"drawable"* su profundidad.

El concepto de contexto gráfico se introdujo en X11. En la versión X10, las funciones gráficas proporcionaban todos los atributos gráficos (tales como ancho de línea, y pixel de foreground) en cada llamada. Esto era ineficiente debido a que los parámetros tenían que enviarse al servidor en cada solicitud, aún si no eran cambiados los atributos. X11 toma una ventaja más eficiente llamando a GC solo una vez y refiriéndose a él por un simple recurso ID en cada llamada a una función gráfica.

La estructura de datos que define el Contexto Gráfico, se describe a continuación:

```
typedef struct {
     int function ;
     unsigned long plane_mask ;
     unsigned long foreground ;/*color del pixel del foreground*/
     unsigned long background ;/*color del pixel del background*/
     int line_width ;
     int line_style ;/*LineSolid, LineOnOffDash, LineDoubleDash*/
     int cap_style ;
     int join_style ;
     int fill_style ;
     int fill_rule ;
     int arc_mode ;
     Pixmap title ;
     Pixmap Stipple ;
     int ts_x_origin ;
     int ts_y_origin ;
     Font font ;
     int subwindow_mode ;
     Bool graphics _exposures ;
     int clip_x_origin ;
     int clip_x_origin ;
     Pixmap clipmask ;
     int dash_offset ;
     char dashes ;
} XGCValues ;
```

Cada uno de estos elementos interviene en las características de los gráficos generados en X Windows. A continuación se describe cada uno de ellos :

Function.- La función de dibujo determina la operación lógica que coloca los pixeles fuente dentro del destino : una ventana o pixmap, las posibles funciones son las siguientes (src = fuente, dst = destino): Estas funciones se utilizan principalmente para animación.

Gxclear	0 (Se copia un 0 dentro dst para todos los puntos en src)
Gxand	src AND dst
GXandReverse	src AND NOT dst
Gxcopy	src(pixels en src reemplazan aquellos en dst)
GXanfInverted	(NOT src) AND dst
Gxnoop	dst (do nothing)
Gxxor	src XOR dst
Gxor	src OR dst
Gxnor	(NOT src) AND (NOT dst)
Gxequiv	(NOT src) XOR dst
Gxinvert	NOT dst (la inversa de dst para todos los puntos en src)
GXorReverse	src OR (NOT dst)
GXcopyInverted	NOT src
GXorInverted	(NOT src) OR dst
Gxnand	(NOT src) OR (NOT dst)
Gxset	1 (Se copia un 1 dentro de dst para todos los puntos en src)

plane mask.- Este campo controla que planos de pixmap destino se afectan en la operación de dibujo. Si el bit en la máscara de plano es 1, ese plano es modificado. Si el bit es 0, la operación de dibujo no afecta al plano.

foreground , background.- Estos campos contienen los colores del background y foreground para dibujar.

line_width.- Este campo controla el ancho de la figura que se desea dibujar (rectángulo, línea, etc).

line_style.- Este campo determina el tipo de línea. LineSolid dibuja una línea sólida. LineDoubleDash dibuja odd dashes usando el **fill_style** actual. LineOnOffDash dibuja como LineSolid pero no odd dashes.

cap_style.- Este campo afecta el trazo de los extremos de líneas y arcos. **CapButt** dibuja extremos cuadrados. **CapRound** dibuja extremos redondeados. **CapProjecting** extiende el punto extremo de la línea a una distancia igual a un medio de l ancho de la línea. **CapNotLast**. En líneas de ancho 0, no dibuja el pixel de punto final.

join_miter.- Este campo afecta la curvatura de las esquinas. **Join_Miter** une las líneas normalmente, **Join_Round** une las líneas con esquinas redondeadas y **JoinBevel** une líneas con ejes **beveled**.

fill_style.- Este campo determina los patrones del llenado de las figuras. **FillTiled**, **FillStippled** y **FillOpaqueStipled**.

fill_rule.- Determina qué partes de un polígono llenar.

arc_mode.- Este campo indica cómo se va a manejar la porción no dibujada de un arco. **ArcPieSlice** ocasiona que la parte no dibujada del arco aparezca como una rebanada de pastel. **ArcChord** ocasiona que sea tratado como un eje recto como la cuerda entre los ángulos inicial y final de un arco.

Stipples, Tiles y Estilos de llenado.- La apariencia de un área rellena, depende de su estilo de relleno o patrón de relleno. El estilo por default es el **FillSolid**, que especifica que los pixeles dentro de la figura serán pintados con el color del fondo actual. Los otros estilos como **FillStippled, FillOpaqueStippled** y **FillTiled,** usan una cuadrícula llamada *stipple* o *tile* para la operación de relleno de figuras. El término *stipple* se refiere a un pixmap de profundidad 1 que sirve como máscara a través de la cual los colores del fondo y frente se aplican durante la operación de relleno. El pixmap stipple es de tamaño fijo, generalmente de 8x8 o 16x16.

El *tile* es un arreglo de pixeles de dos dimensiones con la misma profundidad que el *drawable* para el cual fue creado el contexto gráfico. Mientras que el *stipple* tiene una profundidad de uno, un *tile* tiene la misma profundidad que el drawable y se aplica directamente al área que se va a rellenar.

Clip Mask.- Se puede llamar como máscara de corte. Por default, el servidor corta todas las salidas gráficas en los límites del drawable (ventana o pixmap). El contexto gráfico puede alterar el comportamiento por default. Si se desea dibujar dentro de sub-ventanas, se puede hacer alterando el atributo **subwindow_mode,** con el valor **IncludeInferiors** en lugar de **ClipByChildren**.

5.4.3 Funciones Gráficas Básicas.

Xlib incluye funciones para dibujar puntos, rectángulos, líneas, arcos y polígonos. Los primeros tres parámetros para todas las funciones de dibujo son los siguientes:

- Apuntador al display
- El ID del drawable

- Un GC (Contexto Gráfico) que controla la apariencia de la figura que se va dibujar.

- Display display ;
- Window window ;
- GC GC_1 ;

Los tipos de datos de estos parámetros (**Display**, **GC** y **Window**) están definidos dentro de Xlib :

XDrawPoint (display, window, GC_1, x, y) .- Dibuja un punto en las coordenadas x, y de la ventana *window*.

DrawPoints (display, window, GC_1, pt, numpt, CoordModeOrigin).- Dibuja un conjunto de puntos con una simple solicitud.

Xpoint pt Es un arreglo de puntos x,y

int numpt Es el número de puntos

CoordModeOrigin Indica que las coordenadas son relativas al origen de la ventana o del pixmap.

CoordModePrevious Se da cuando la coordenada de cada punto se da en términos

de los desplazamientos en x y y de los puntos previos.

XDrawLine (display, window, GC_1, x1, y1, x2, y2).- Dibuja una línea recta desde el punto inicial x1,y1 al punto final x2,y2.

int x1,y1 Coordenadas del extremo inicial de la línea.

int x2, y2 Coordenadas del extremo final de la línea.

XDrawSegments (display, window, GC_1, line, numsegs).- Esta función para dibujar varios segmentos de línea disjuntos utilizando los mismos atributos gráficos. Los segmentos son especificados usando la estructura especificada Xsegment :

```
typedef struct {
        short x1,y1 ;
        short x2,y2 ;
} Xsegment ;

Display display ;
Window window ;
GC  CG_1 ;
Xsegment  lines[ ] = { [100,100,200,200}, {10, 20, 35, 60}, {300,10,300,100}} ;
int numsegs = sizeof(lines) / sizeof(Xsegment) ;
```

XDrawRectangle (display, window, GC, x, y, width, height).- Dibuja un rectángulo a partir de las coordenadas x, y de su esquina superior izquierda indicando el ancho (width) y el largo (height) del rectángulo.

XFillPolygon (display, window, GC_1, points, numpoints, shape, mode) .-

```
int shape ; /*Convex, NonConvex, or Complex */
int mode ; /*CoordModeOrigin or CoordModePrevious */
```

Esta función llena un polígono con el patrón deseado. Se especifican los vértices del polígono en un arreglo de puntos (points), el argumento shape, ayuda al servidor a optimizar el algoritmo de relleno, Convex para figuras convexas. NonConvex para figuras no convexas y Complex para polígonos con ejes de intersección.

XDrawArc (display, window, GC_1, x, y, width, height, grad1, grad2).- Es similar a la función XDrawRectangle pues se pide la coordenada de la esquina superior izquierda y el largo y ancho del rectángulo dentro del cual se dibujará el arco ; los parámetros adicionales son los ángulos inicial y final del arco, esto se expresa en sesenta y cuatroavos de grado (de 0 a 360*64 equivale a un círculo)

XDrawArcs (display, window, GC_1, arcs, numarcs).- Con esta función se dibujan varios arcos.

Arcs es un arreglo de estructuras tipo Xarc, este tipo se define a continuación .

```
typedef struct
{     short x,y ;
      unsigned short width, height ;
      short     angle1, angle2 ;
}Xarc ;
```

numarcs es el número de arcos en el arreglo

5.4.3 Programación de gráficos en el ambiente Xwindow

Como en un programa normal escrito en turbo C, Pascal o Borland C, es necesario inicializar el ambiente gráfico para poder generar gráficos.

El primer paso es inicializar el contexto gráfico, esto se realiza mediante la siguiente función:

```
void setup_gc()
{
      int foreground, background ;
      XGCValues vals ;
      Arg al[10] ;
      int ac ;

/*  Para  obtener  los  valores  actuales  del  background  y
foreground*/

ac =0 ;
XtSetArg(al[ac], XmNforeground, &foreground) ; ac++ ;
XtSetArg(al[ac], XmNbackground, &background) ; ac++ ;
XtGetValues(drawing_area, al, ac) ;

/*crea el contexto gráfico*/
vals.foreground = foreground ;
vals.background = background ;

gc = XtGetGc(drawing_area, GCForeground, GCBackground, &vals) ;
}
```

La función **XtGetGc** inicializa el contexto gráfico.

Una vez inicializado el ambiente, se pueden utilizar las funciones gráficas.

A continuación se presenta un ejemplo de la utilización de funciones gráficas:

```
/*draw1.c*/

#include <Xm/Xm.h>
#include <Xm/DrawingA.h*/

XtAppContext  context ;
GC GC ;
Widget toplevel ;
Widget drawing_area ;

void setup_gc()
...

void exposureCB(w, client_data, call_data)

Widget w ;
XtPointer client_data ;
XtPointer call_data ;
{
   XDrawLine(XtDisplay(drawing_area), XtWindow(drawing_area), gc,
0, 0, 300,300) ;
}

void main(argc,argv)
int argc ;
char *argv[ ] ;
{
      Arg al[10] ;
```

```
      int ac ;
     /*crea el shell de alto nivel */
      toplevel = XtAppInitializa(&context, ▒▒ , NULL, 0, &argc,
argv, NULL, NULL,0) ;

      /*Obtiene el tamaño de la ventana*/
      ac = 0 ;
      XtSetArg(al[ac[ ], XmNheight, 300) ; ac++ ;
      XtSetArg(al[ac[ ], XmNwidth, 300) ; ac++ ;
}
```

5.5 CONCLUSIÓN DEL CAPÍTULO

Xwindows es un ambiente de desarrollo poderoso para la representación de gráficos ya que cuenta con funciones gráficas básicas contenidas en las bibliotecas de Xlib , permite el desarrollo de aplicaciones bajo el ambiente cliente-servidor., además de que está diseñado para elaborar interfaces amigables para el usuario sobre un sistema operativo UNIX, lo cual permite aprovechar los recursos que máquinas que con este sistema operativo ofrecen.

Capítulo 6

Estudio de casos

6.1 EL ENTORNO DE PROGRAMACIÓN DE DALI

El objetivo de este capítulo es presentar algunos casos ilustrativos de cómo DALI presenta en gráficas información de un problema de base de datos resuelta por LIDA. Los casos de estudio son principalmente dos: un problema de planeación de ventas y otro de análisis de ventas de petróleo.

En ambos casos los problemas se desarrollan desde el planteamiento conceptual de la base de datos, resolviendo algunas consultas con LIDA para después mostrar resultados en un ambiente gráfico con DALI.

6.1.1 **Interfaz Gráfica**

Si se ejecuta la aplicación de manera independiente a LIDA, el usuario deberá invocar al sistema tecleando el comando *dali* en la estación de trabajo. Se presenta en pantalla una ventana de inicio y al oprimir el botón derecho del ratón aparece un menú Pop-up con las siguientes opciones:

Lectura de Datos

Generación de Gráficos

Salir

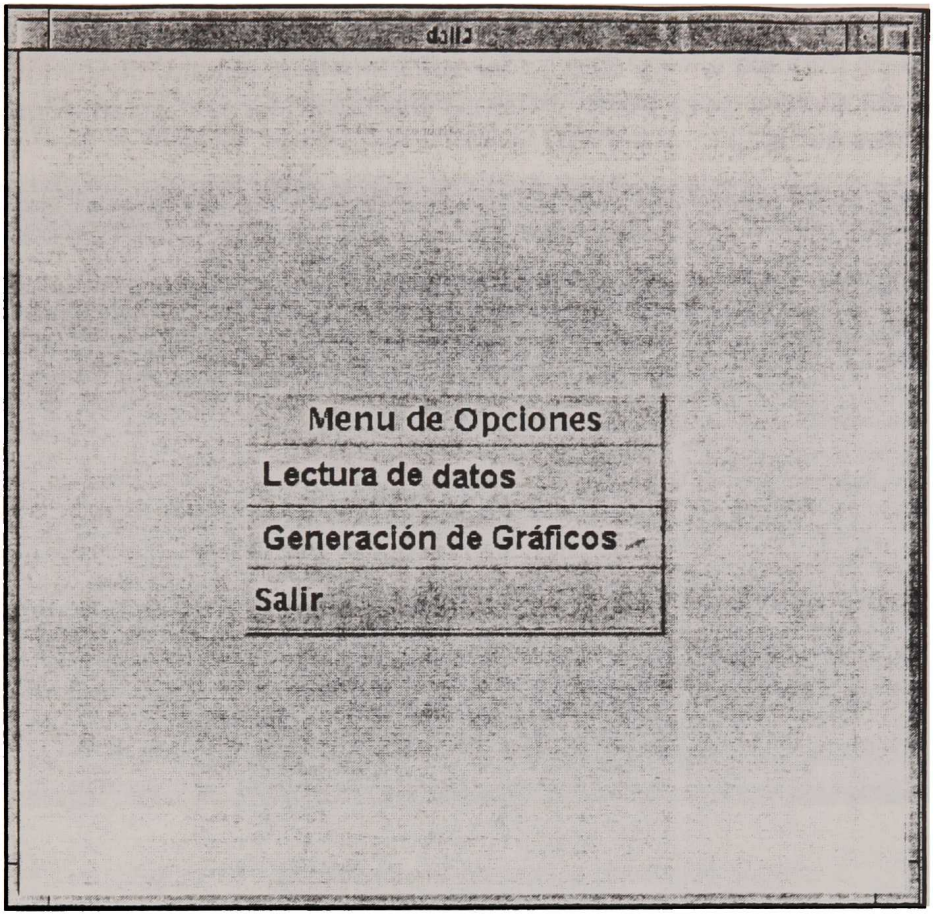


Figura 6.1.- Pantalla inicial de DALI como aplicación independiente.

1) **Lectura de datos** .- Con esta opción se realiza el proceso de análisis de datos. Se presenta en pantalla el siguiente menú que permite al usuario definir los campos y el nombre del archivo para posteriormente introducir la información. Figura 6.2 Menú de opciones de DALI

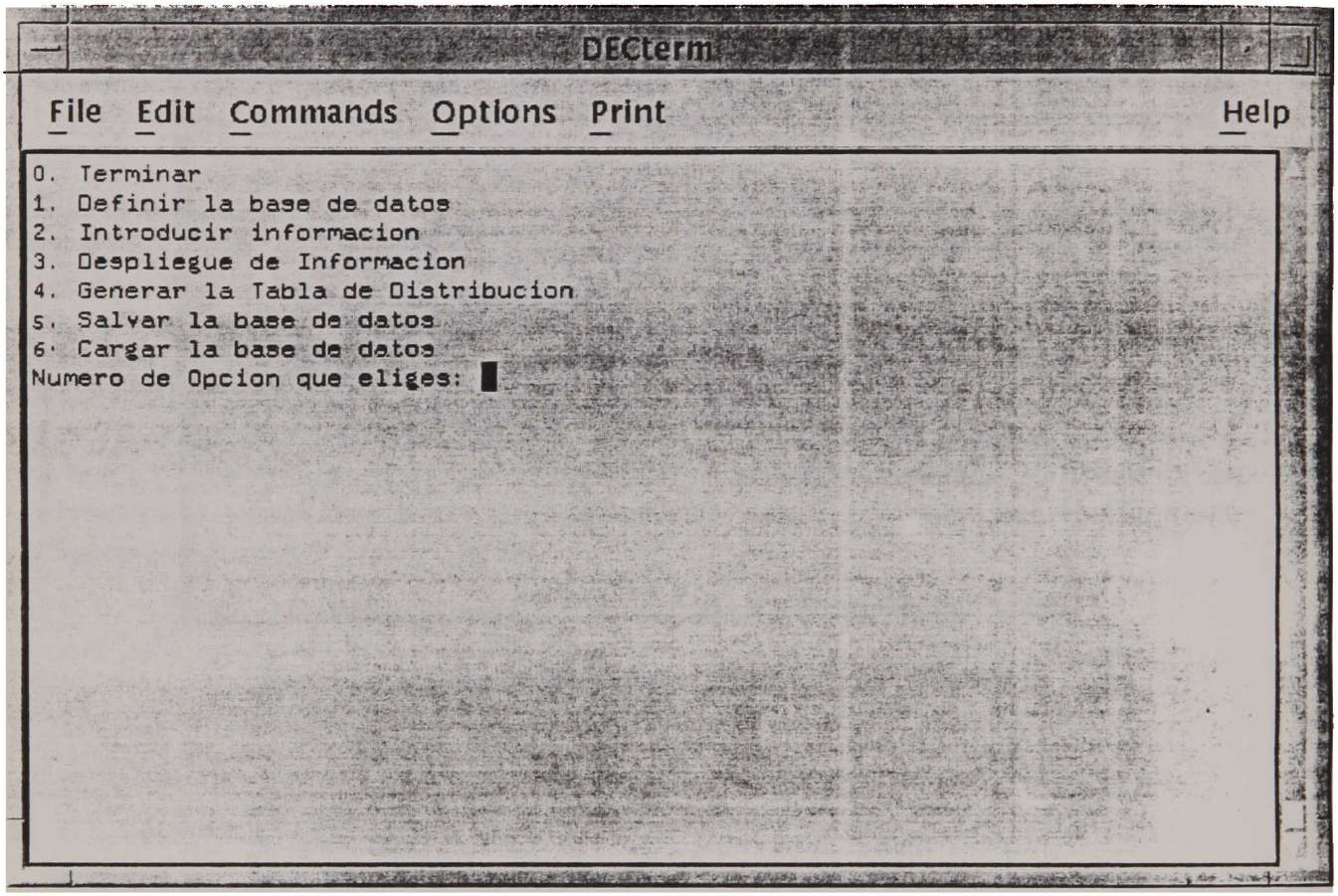


Figura 6.2.- Menú de opciones de DALI

Esta versión del sistema presenta un menú de opciones sencillo que trabaja bajo un ambiente UNIX.

1. Definir Base de Datos.- En esta opción el usuario podrá definir los campos de datos que desee indicando su nombre, tipo y tamaño en bytes. También aquí se define el nombre del archivo. Como primer paso se define el número de campos que tendrá el registro de la tabla como se muestra en la figura 6.3.

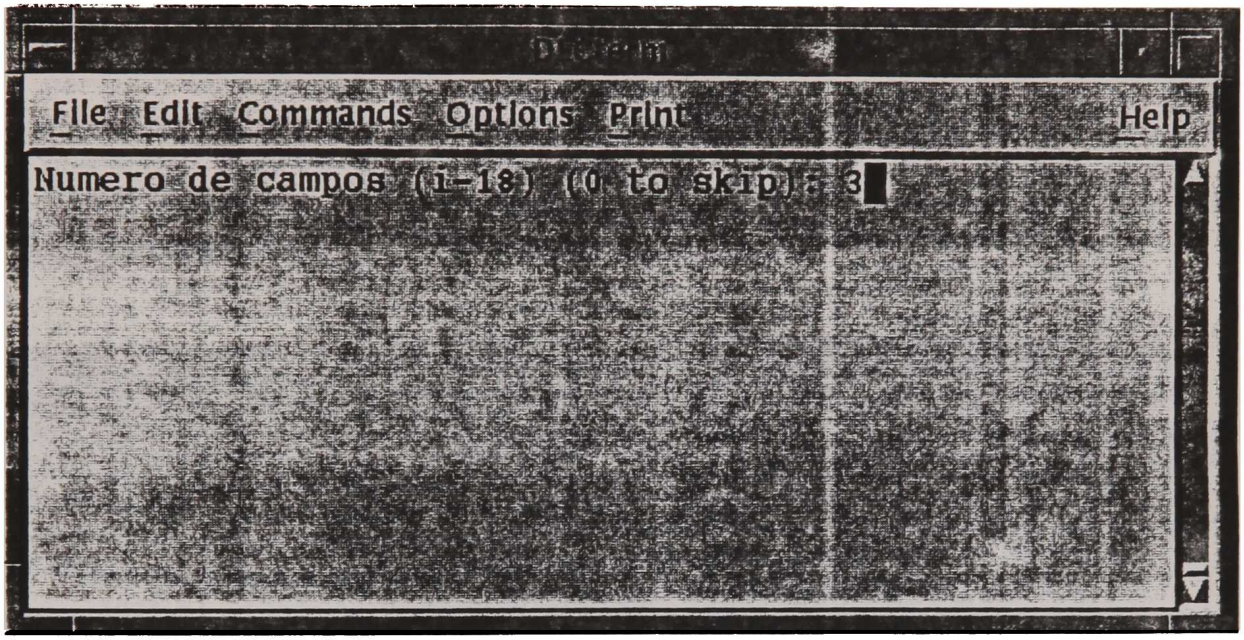


Figura 6.3.- Introducción del número de campos por registro.

Después se debe introducir nombre de cada uno de los campos que conforman la tabla:

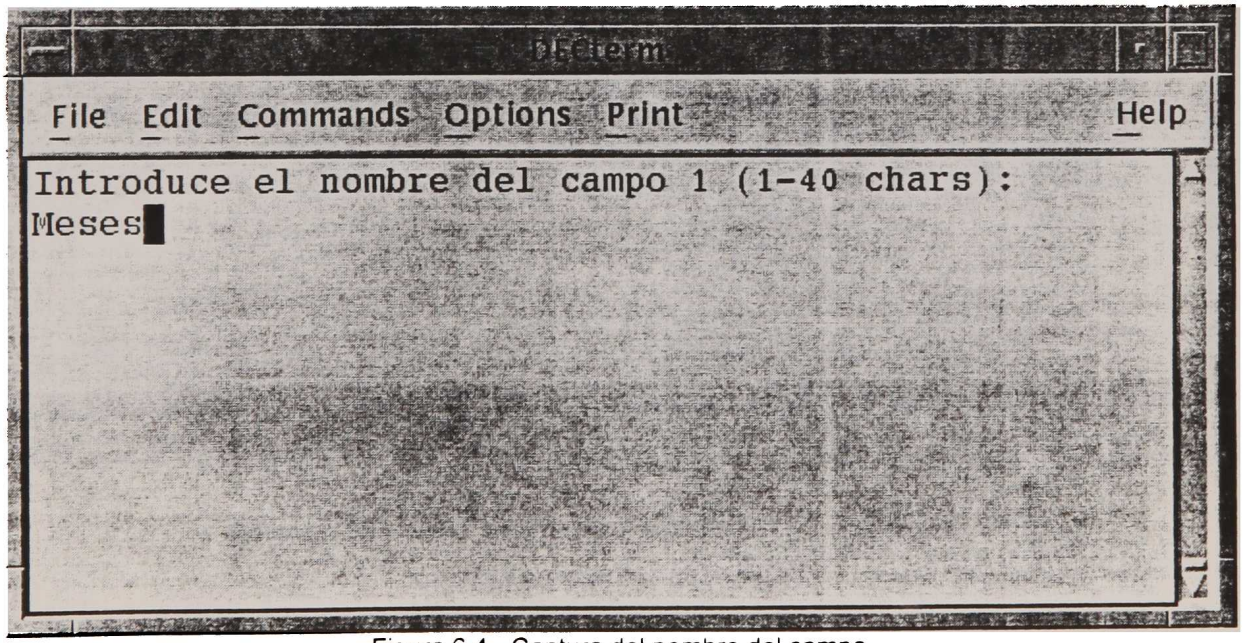


Figura 6.4.- Captura del nombre del campo

Después se debe introducir el tipo de cada uno de los campos y su longitud en bytes:

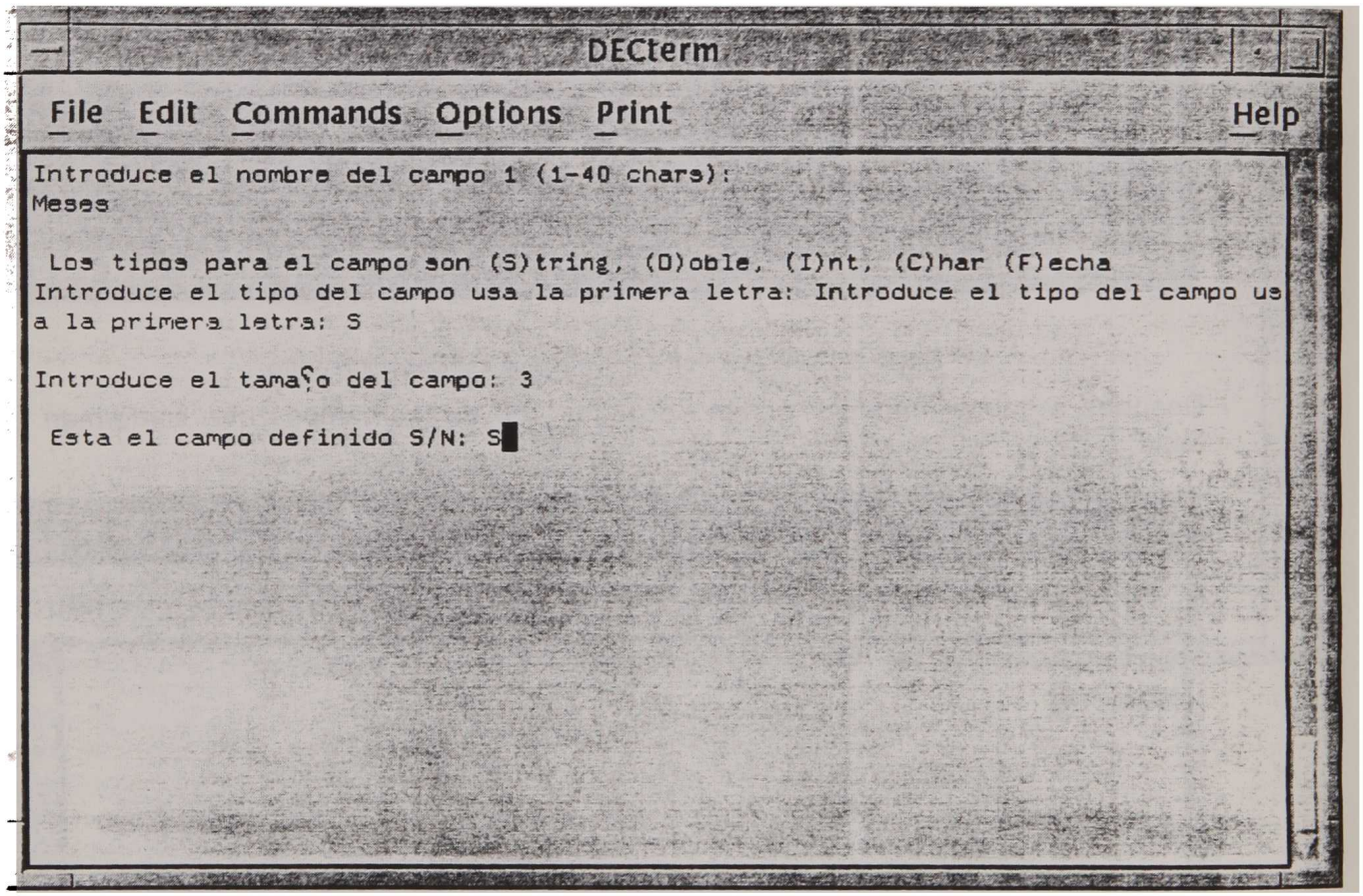


Figura 6.5.- Pantalla de captura del tipo y longitud de los campos

2. Introducir información.- Con esta opción se podrán introducir los valores de cada campo.
3. Despliegue de información.- Esta opción permite verificar que los datos fueron agregados correctamente.
4. Generar Tabla de Distribución.- Aquí es donde se inicia el Análisis de Datos. Se genera la tabla de distribución, y se realiza el análisis para generar la matriz o tabla de datos, se despliega en pantalla el resultado de los procesos que intervienen en el análisis. Una vez generada la tabla, el sistema regresa a la ventana de inicio. Como ejemplo se muestra la siguiente pantalla que despliega la tabla de distribución:

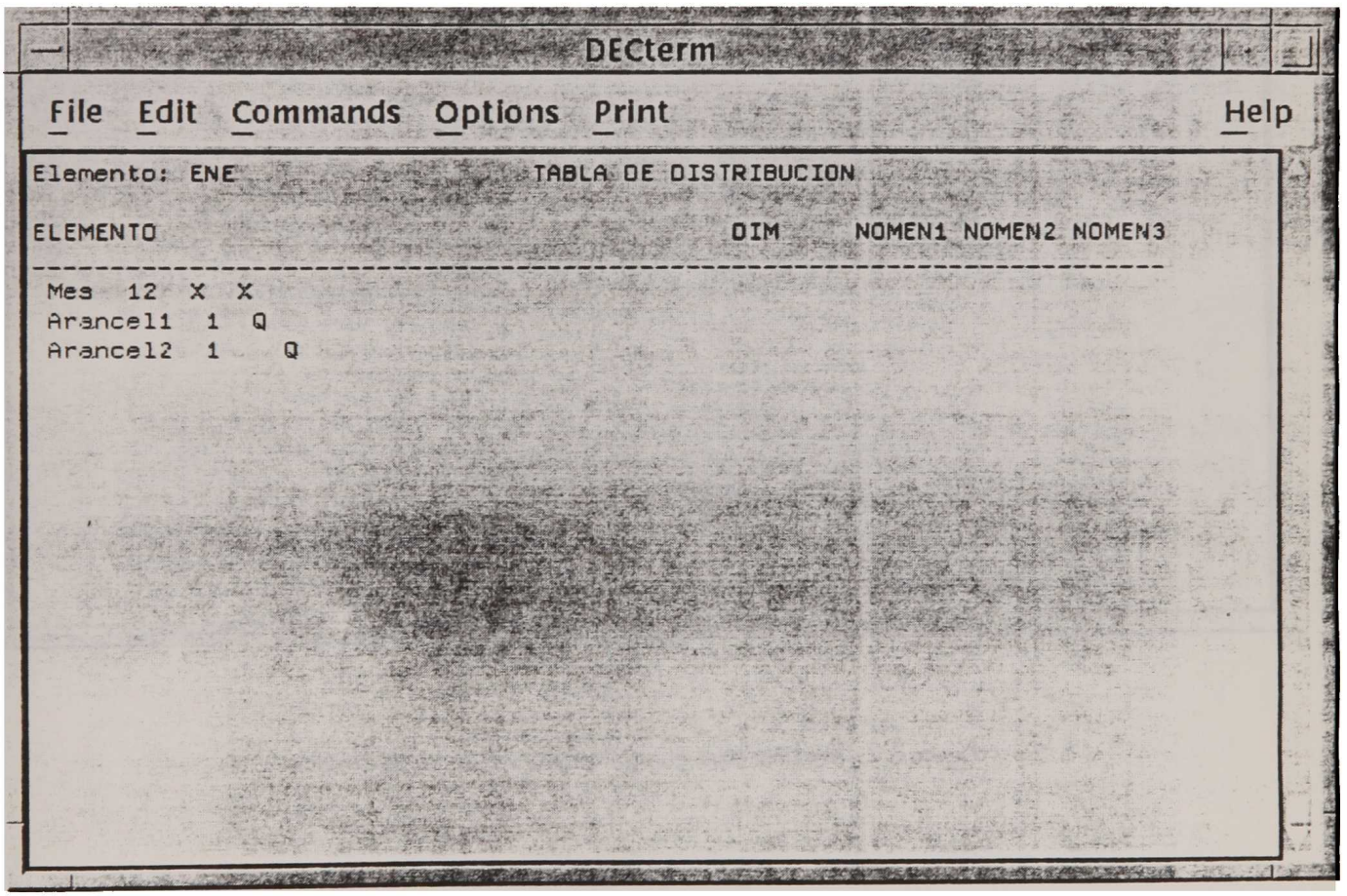


Figura 6.6.- Tabla de Distribución

5. Salvar la base de datos

Con esta opción se graba en disco la definición de la base de datos y su contenido. El usuario debe indicar el nombre con el que se va a salvar el archivo de datos.

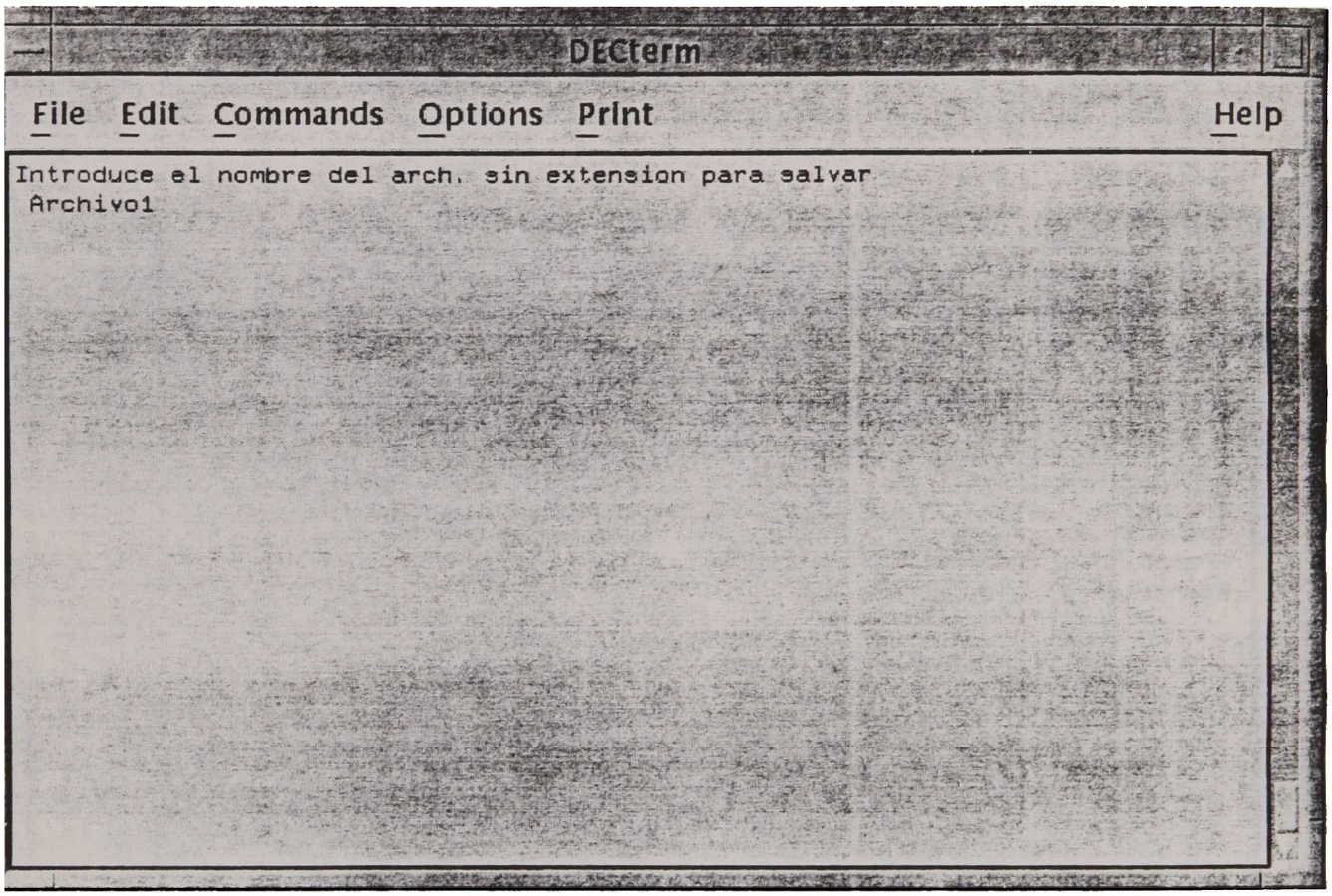


Figura 6.7.- Pantalla para guardar el archivo de datos creado

6. Cargar la base de datos

Es la opción que permite la lectura del archivo de datos sobre el cual se generará la gráfica automáticamente. Cabe hacer mención, que este archivo de datos equivale a la tabla de datos que resulta de la ejecución de alguna operación realizada en la base de datos a través de DALI.

2) Generación de Gráficos

Al elegir esta opción, comienza el proceso de construcción gráfica y se genera la gráfica apropiada. Cada una de las ventanas en donde se presentan las gráficas puede manipularse con el ratón, pueden moverse a través de la pantalla y cambiar de tamaño.

3) **Salir**

Con esta opción termina la ejecución del sistema.

6.2 ESTUDIO DE CASOS

Para ejemplificar la interacción de LIDA y DALI, se presentan a continuación los siguientes ejemplos:

6.2.1 Caso 1 Planeación de ventas

Un cliente coloca una orden de pedido a través de un vendedor particular para una cantidad de un producto específico que se embarca en cierta fecha. Los clientes se encuentran localizados en determinadas regiones de mercado y un vendedor sirve a los clientes dentro de una región particular.
El objetivo del sistema es que el usuario pueda tomar decisiones a tiempo en la planeación de ventas, análisis de mercado, embarques, etc. , tomando como base la visualización de la información para el seguimiento de las órdenes de pedido, clientes y vendedores para maximizar el servicio de venta.

Modelo Conceptual de la Base de Datos:

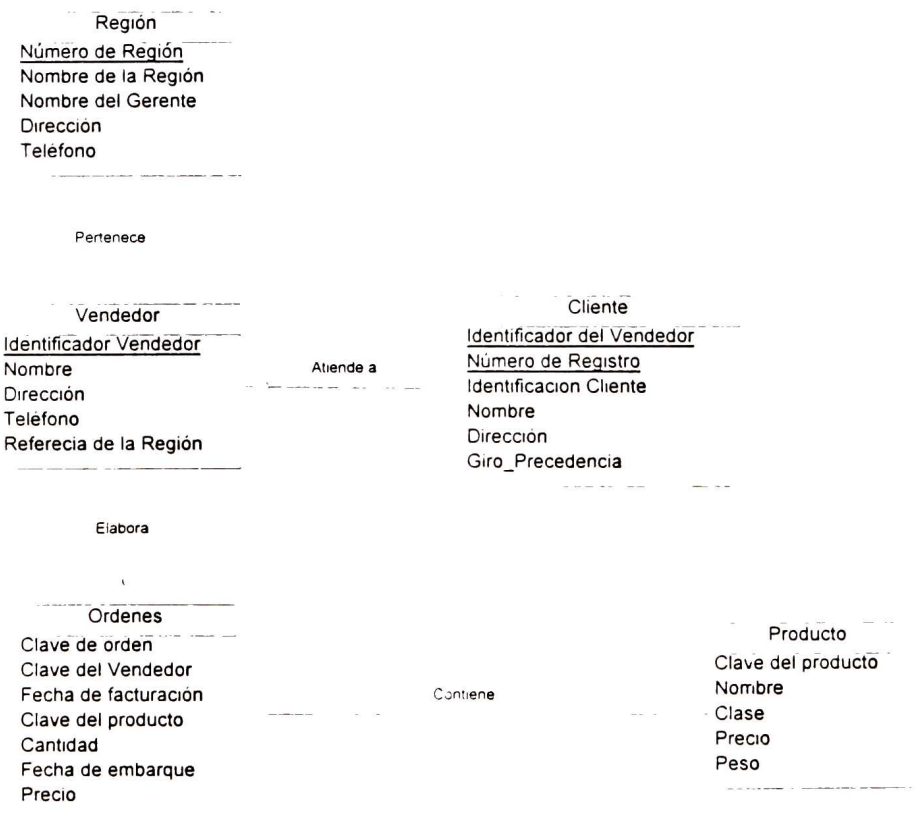


Figura 6.8.- Modelo Conceptual del caso 6.2.1

Modelo Físico:

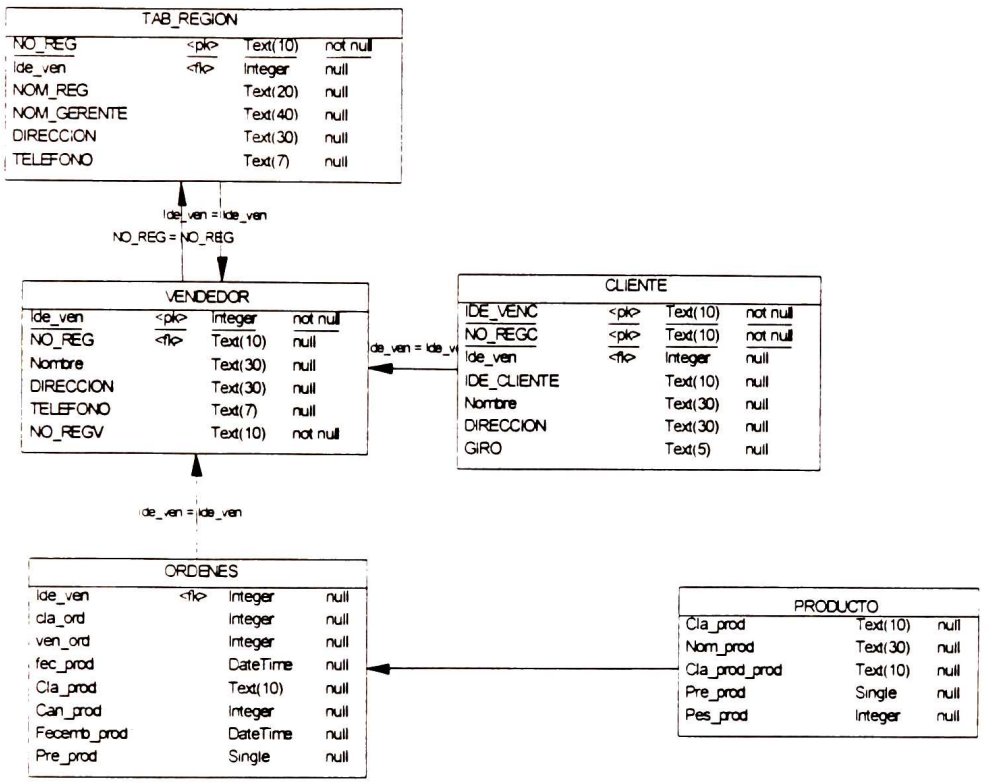


Figura 6.9.- Modelo Físico del caso 6.2.1

Tabla Cliente

Nombre Lógico	Nombre Físico	Tipo	P	M
Identificador del Vendedor	IDE_VENC	Text(10)	Si	Si
Número de Registro	NO_REGC	Text(10)	Si	Si
Identificador Vendedor	Ide_ven	Integer	No	No
Identificacion Cliente	IDE_CLIENTE	Text(10)	No	No
Nombre	Nombre	Text(30)	No	No
Dirección	DIRECCION	Text(30)	No	No
Giro_Precedencia	GIRO	Text(5)	No	No

Tabla Ordenes

Nombre Lógico	Nombre Físico	Tipo	P	M
Identificador Vendedor	Ide_ven	Integer	No	No
Clave de orden	cla_ord	Integer	No	No
Clave del Vendedor	ven_ord	Integer	No	No
Fecha de facturación	fec_prod	DateTime	No	No
Clave del producto	Cla_prod	Text(10)	No	No
Cantidad	Can_prod	Integer	No	No
Fecha de embarque	Fecemb_prod	DateTime	No	No
Precio	Pre_prod	Single	No	No

Tabla Producto

Nombre Lógico	Nombre Físico	Tipo	P	M
Clave del producto	Cla_prod	Text(10)	No	No
Nombre	Nom_prod	Text(30)	No	No
Clase	Cla_prod_prod	Text(10)	No	No
Precio	Pre_prod	Single	No	No
Peso	Pes_prod	Integer	No	No

Tabla Región

Nombre Lógico	Nombre Físico	Tipo	P	M
Número de Región	NO_REG	Text(10)	Si	Si
Identificador Vendedor	Ide_ven	Integer	No	No
Nombre de la Región	NOM_REG	Text(20)	No	No
Nombre del Gerente	NOM_GERENTE	Text(40)	No	No
Dirección	DIRECCION	Text(30)	No	No
Teléfono	TELEFONO	Text(7)	No	No

Tabla Vendedor

Nombre Lógico	Nombre Físico	Tipo	P	M
Identificador Vendedor	Ide_ven	Integer	Si	Si
Número de Región	NO_REG	Text(10)	No	No
Nombre	Nombre	Text(30)	No	No
Dirección	DIRECCION	Text(30)	No	No
Teléfono	TELEFONO	Text(7)	No	No
Referecia de la Región	NO_REGV	Text(10)	No	Si

El problema consiste en tomar las entidades *producto*, *órdenes* y *clientes* para poder ver cual es el total de ventas durante un período considerando los diversos aranceles de impuesto.

Para resolver este problema, se ejecuta la operación *junta* de las entidades *producto* y *órdenes* para proyectarlas con los atributos de interés como se indica a continuación:

Icono 1: Proyección(Clave del producto, clase, total, cantidad, fecha de facturación, $P/P)

Como resultado de esta proyección se obtiene una tabla de datos la cual pasa por un proceso de selecciòn en donde separamos la clase 1 correspondiente al arancel del 15% y la clase 2 correspondiente al arancel del 10%. En cada uno de los casos se invoca el proceso que calcula el total con base en el porcentaje mencionado:

Proceso A1: El pago por arancel de clase es de 15%.

Proceso A2: El pago por arancel de clase es de 10%.

Los resultados se agrupan en una nueva clase de datos (fecha de facturación) de acuerdo con el mes del año cuando se hace la facturación. Como la idea es comparar la tendencia de ventas según el arancel, se suman los totales y se despliegan las gráficas como se muestra en la figura.

Un segundo problema corresponde a tratar de encontrar las anomalías que se tienen en la fecha de embarque en contra de la fecha de facturación, en este caso, simplemente proyectamos la información según:

Icono 2: Proyección(Clave del producto, Clave de orden, fecha de facturación, fecha de emb, clase)

Para después simplemente graficar y visualizar lo antes mencionado. Además, por el siguiente ícono que denota encontrar la función promedio de fecha de embarque, podemos descubrir nueva información.

La figura siguiente representa el flujograma del modelo descrito.

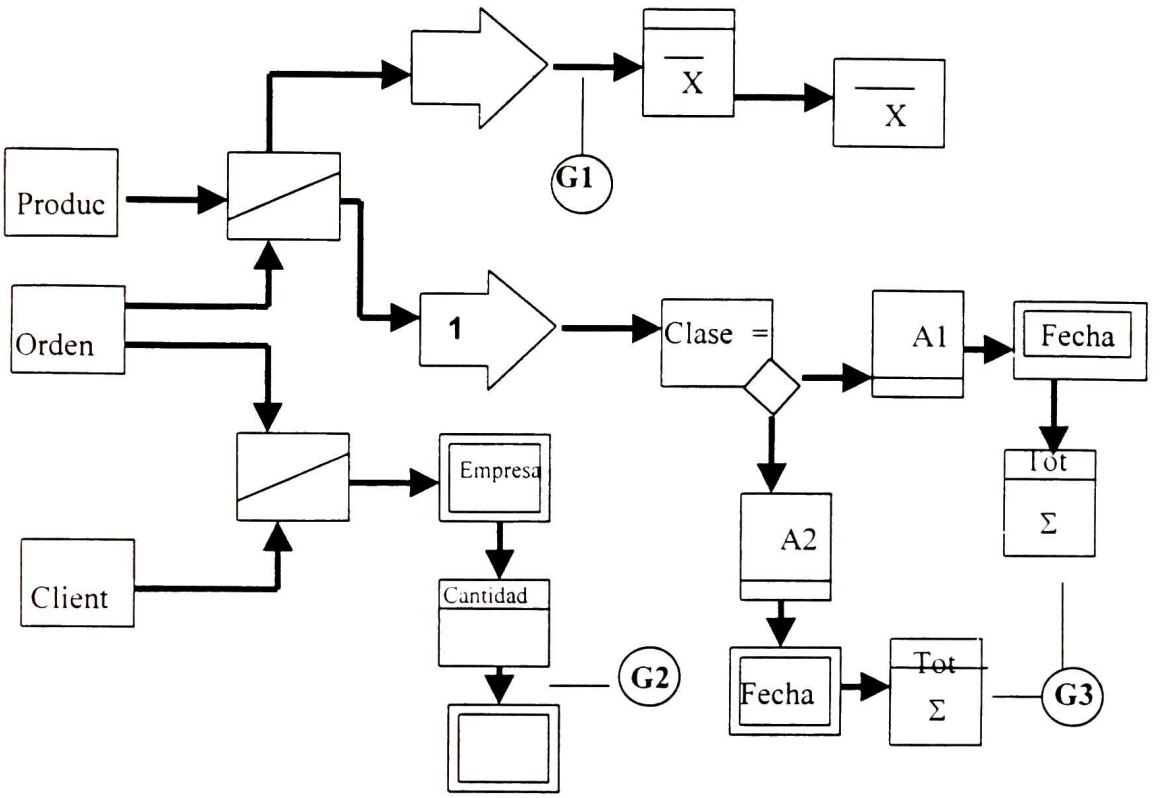


Figura 6.10.- Flujograma generado en LIDA

Cada círculo con la notación G1, G2 y G3, representan las gráficas resultantes de la operación realizada, por falta de espacio sólo se indica el lugar en donde aparecerá cada gráfica.

Las gráficas se presentan a continuación:

G1.- Esta gráfica representa la compra de un determinado producto por varias empresas:

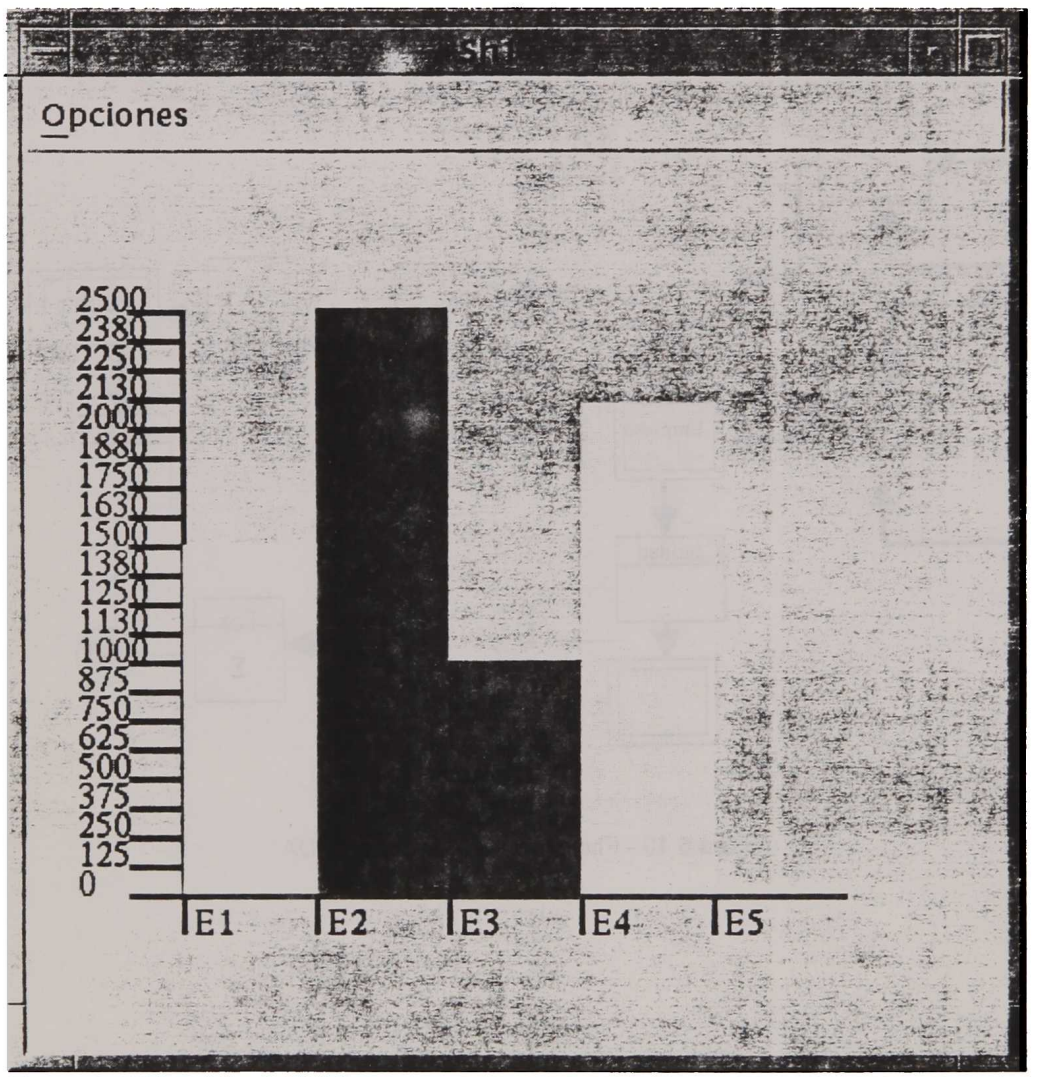


Figura 6.11 Gráfica representativa de la compra de un producto por varias empresas

G2.- Esta gráfica nos permite observar anomalías que existen con respecto a la fecha de facturación y la fecha de embarque, es decir, cuando no existe correspondencia entre la fecha de facturación y la fecha de embarque. El sistema DALI permitirá observar el número de facturas cuya fecha de embarque se encuentra fuera del tiempo especificado.

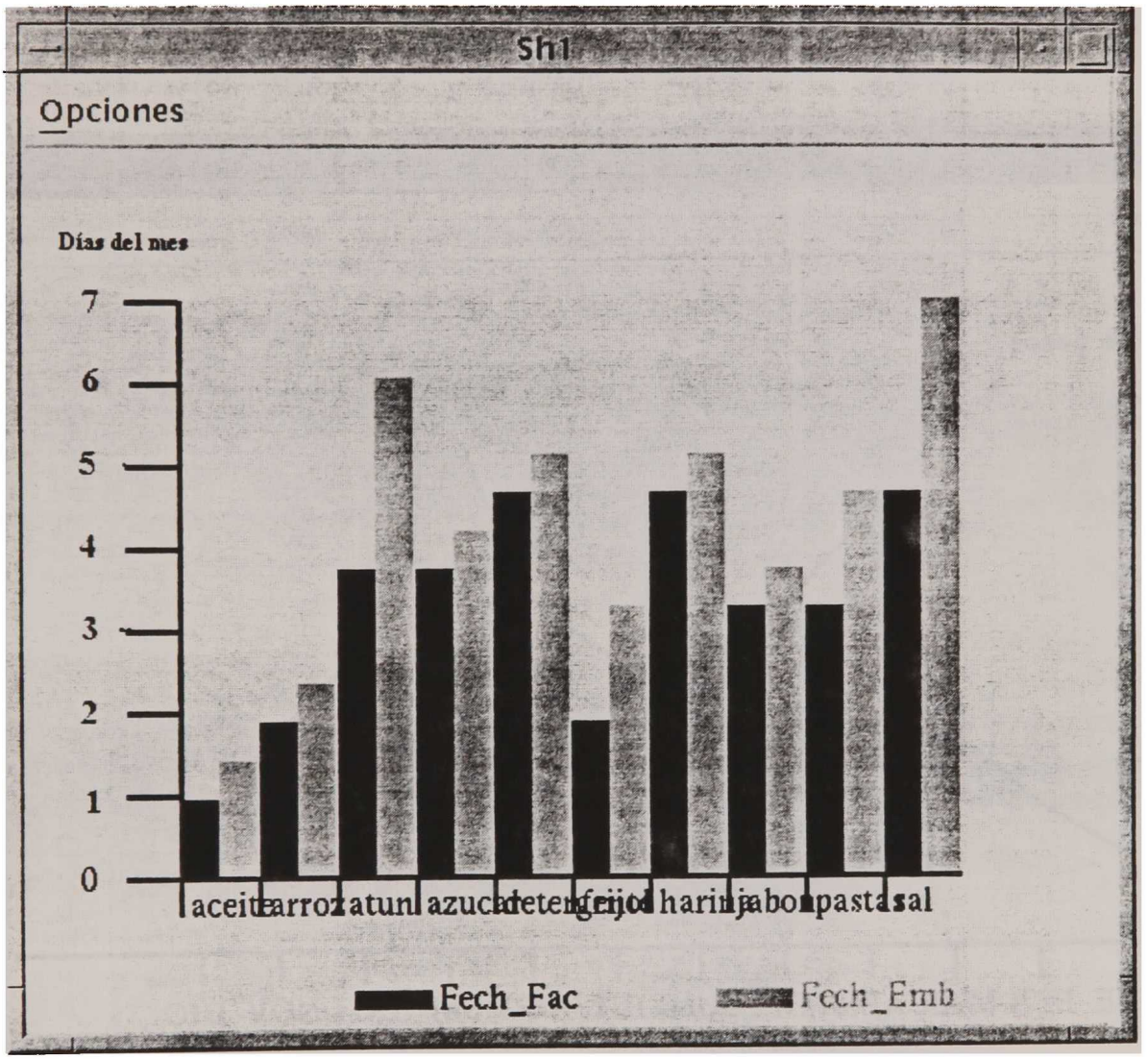


Figura 6.12.- Gráfica G2

La gráfica G2 permite observar las diferencias entre la fecha de facturación y la fecha de embarque de cada producto, con esto se puede comenzar a investigar las causas que ocasionan estas diferencias de tiempo.

G3.- Esta gráfica demuestra los totales de ventas de acuerdo a un arancel de 15% (A) y otro de 10% (B) a lo largo de un período de 12 meses.

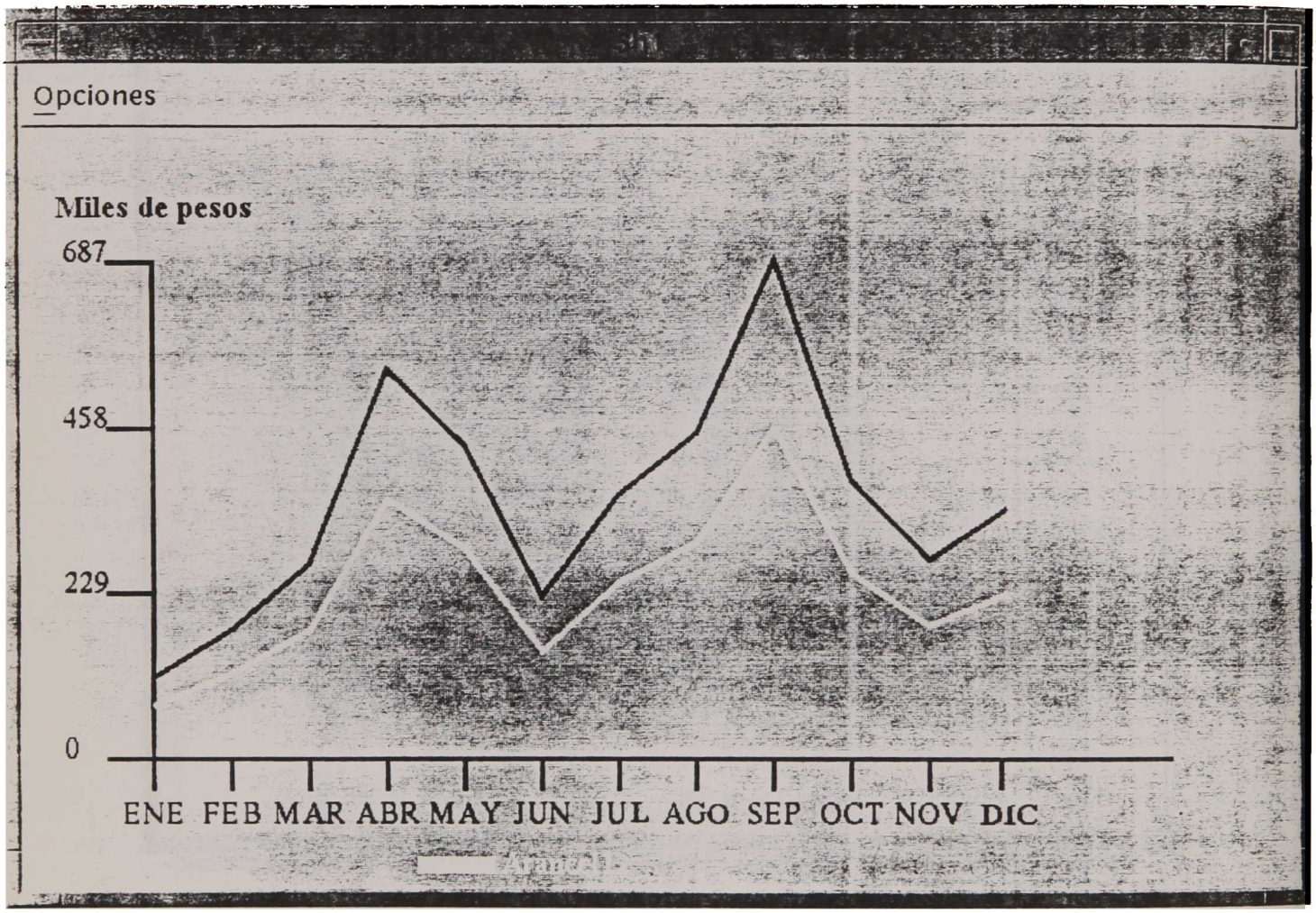


Figura 6.13.- Gráfica G3, Comparación de Aranceles

La información correspondiente a esta gráfica se muestra en la siguiente tabla:

Tabla 6.1.- Datos Aranceles de impuestos, según el importe de ventas

Mes	Importe	Arancel 1 (10%)	Arancel 2 (15%)
ENE	750.00	75.00	112.00
FEB	1200.00	120.00	180.00
MAR	1800.00	180.00	270.00
ABR	3600.00	360.00	540.00
MAY	2900.00	290.00	435.00
JUN	1500.00	150.00	225.00
JUL	2450.00	245.00	367.50
AGO	3000.00	300.00	45.00
SEP	4580.00	458.00	687.75
OCT	2570.00	257.00	385.5
NOV	184.00	184.00	276.00
DIC	2300.00	230.00	345.00

A través estas gráficas se puede obtener un mayor conocimiento acerca del comportamiento de los datos en un cierto período de tiempo o bajo ciertas circunstancias de manera que se puedan sacar conclusiones o realizar planeaciones a futuro.

6.2.2 Caso 2 Análisis de ventas de petróleo internacional

Este ejemplo es para poder representar la información multivariada. El modelo lógico y físico de la base de datos se presenta a continuación.
Los datos para este ejemplo se encuentran registrados en la tabla 1 del anexo 1.

Modelo Conceptual de la Base de Datos

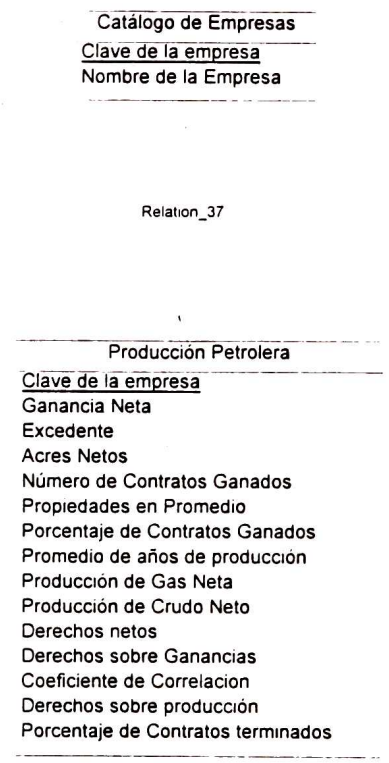


Figura 6.14 .- Modelo Lógico del ejemplo

Modelo Físico

Cat_emp			
EMP_CVE	<pk>	Integer	not null
EMP_NOMB		Text(15)	null

EMP_CVE = EMP_CVE

Prod_Petro			
EMP_PP_CVE	<pk>	Integer	not null
EMP_CVE	<fk>	Integer	null
PP_GAN		LongInteger	null
PP_EXCE		LongInteger	null
PP_ACRES		LongInteger	null
PP_CONTRAT		Integer	null
PP_PROP		Integer	null
PP_PORCONT		Integer	null
PP_PROMANIO		Integer	null
PP_PRODGAS		LongInteger	null
PP_PRODCRUD		LongInteger	null
PP_DERECHOS		LongInteger	null
PP_DERGAN		LongInteger	null
PP_COEF		LongInteger	null
PP_DERPROD		LongInteger	null
PP_CONTRTER		LongInteger	null

Figura 6.15.- Modelo Físico del ejemplo

Si se desea realizar la selección de todas las empresas petroleras de Estados Unidos. la consulta sería:

Q1 = **SELECT** * **FROM** PROD_PETRO **WHERE** PP_PAIS = "USA"

En LIDA, la ejecución de esta consulta se realizaría así:

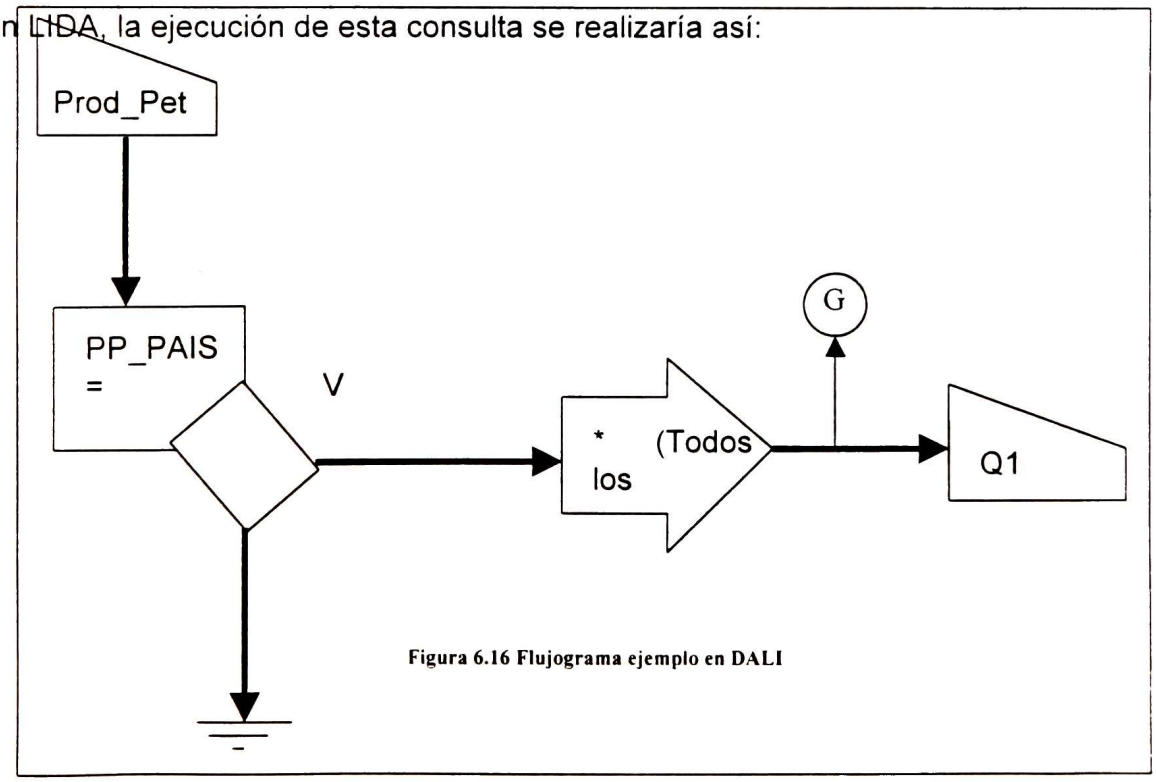


Figura 6.16 Flujograma ejemplo en DALI

El resultado Q1 es el conjunto de registros que cumpla con la condición del SELECT.

Si se desea observar el resultado de esta consulta de una manera gráfica se deberá invocar al sistema DALI, una vez que se haya realizado la operación SELECT. La gráfica aparecerá en el lugar señalado como **G** que se muestra en la figura 6.16.

Como se trata de una tabla con más de 5 campos referentes al campo llave, DALI generará una gráfica para datos multivariados como la que se muestra a continuación:

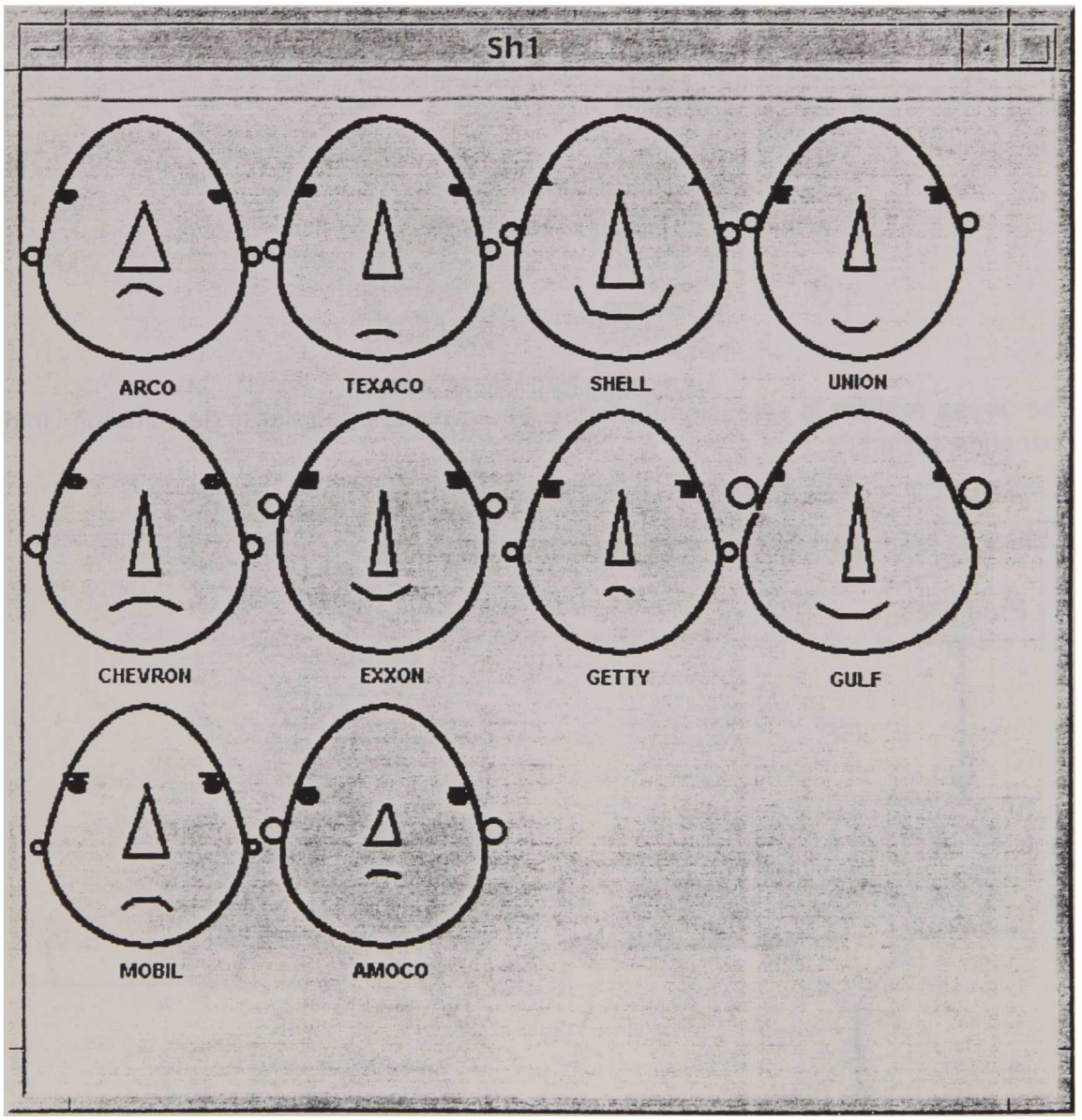


Figura 6.17.- Gráfica generada por DALI para datos multivariados.

Con el ejemplo anterior se puede captar de manera clara y directa la situación general de cada una de las empresas observando los rasgos característicos en las caras generadas. A simple vista se puede concluir que las compañías que se encuentran en mejores condiciones son: Shell, Exxon o Gulf y las que tienen algunos problemas son Texaco, Chevron o Mobil de acuerdo a los valores que presentan ciertas características como número de contratos ganados, producción de gas neta, propiedades en promedio, etc.

Para observar la asignación de valores de campos contra características faciales, observe la tabla 2.2 y 2.3 del capítulo 2.

6.3 OTROS EJEMPLOS

El siguiente ejemplo muestra una gráfica de pastel cuya información corresponde al porcentaje de recaudación de un determinado año fiscal. Los valores suman un total del 100% y debido a que son pocas características, esta es la gráfica más apropiada.

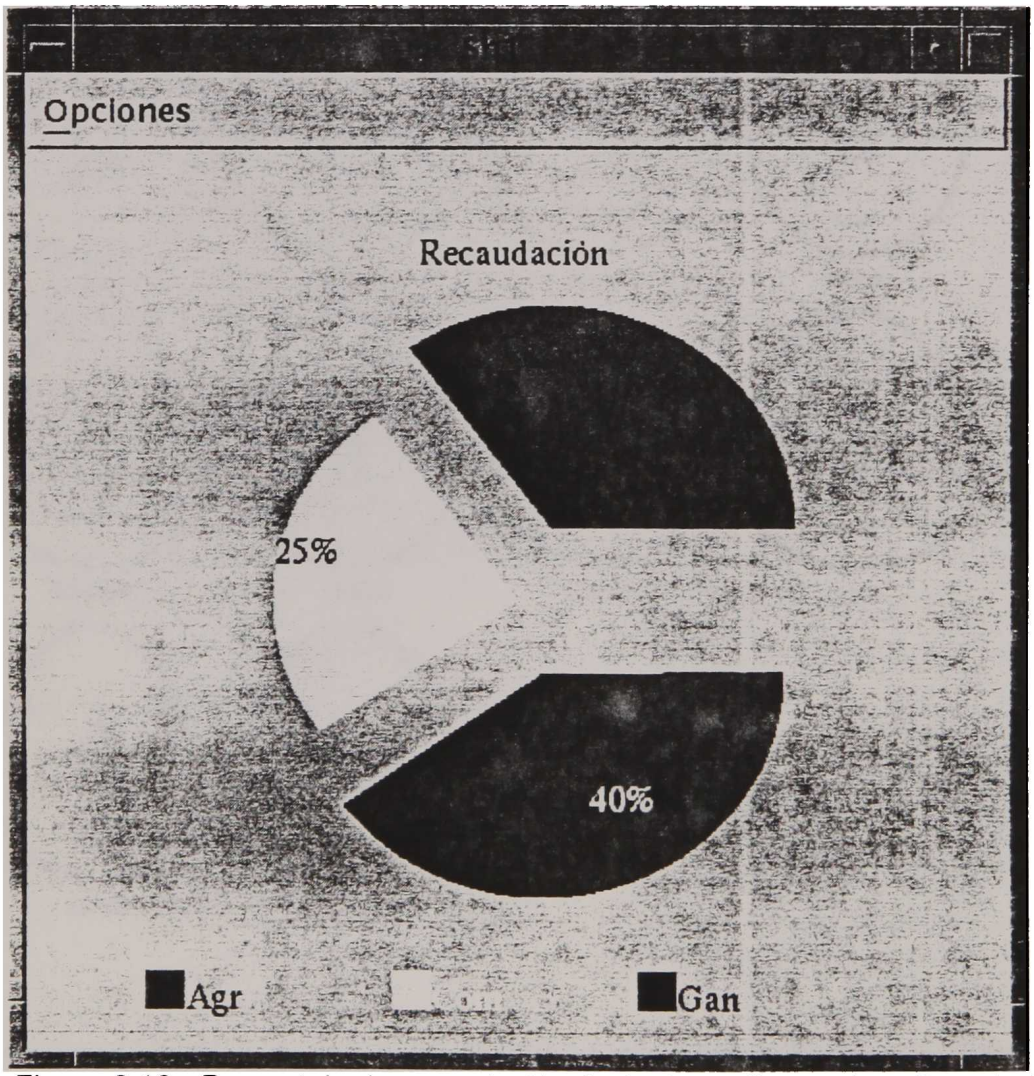


Figura 6.18.- Porcentaje de recaudación Agricultura, Comercio y Ganadería

La siguiente gráfica es una gráfica de matriz de permutaciones resultado de una tabla de datos con más de tres características la construcción de esta gráfica se describe en el capítulo 3.

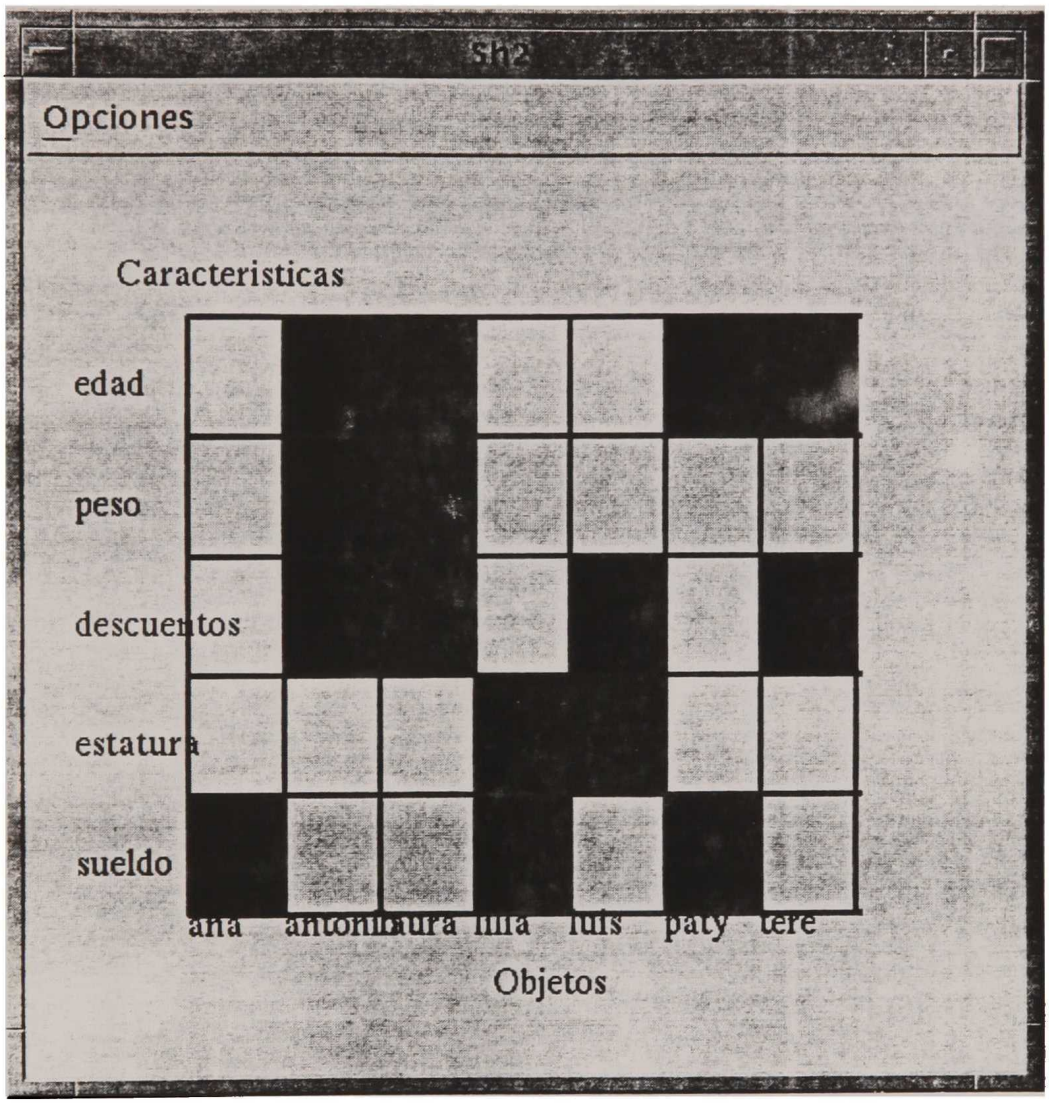


Figura 6.19.- Matriz de Permutaciones

Cada registro de la tabla cuenta con los siguientes campos:

Nombre del Empleado

Edad

Peso

Descuentos

Estatura

Sueldo

El *nombre del empleado* corresponde al elemento OBJETO de la tabla y el resto de los campos son sus características. Cada uno de estos valores se va agrupando de acuerdo a su similitud generando la matriz que se muestra en la parte superior.

La siguiente gráfica es un scatter plot que representa la información referente a un lote de venta de autos en donde se compara el número de millas recorridas por auto y el precio de cada uno.

En esta gráfica podemos observar que menor sea el número de millas recorrido por un automóvil, mayor es su precio de venta.

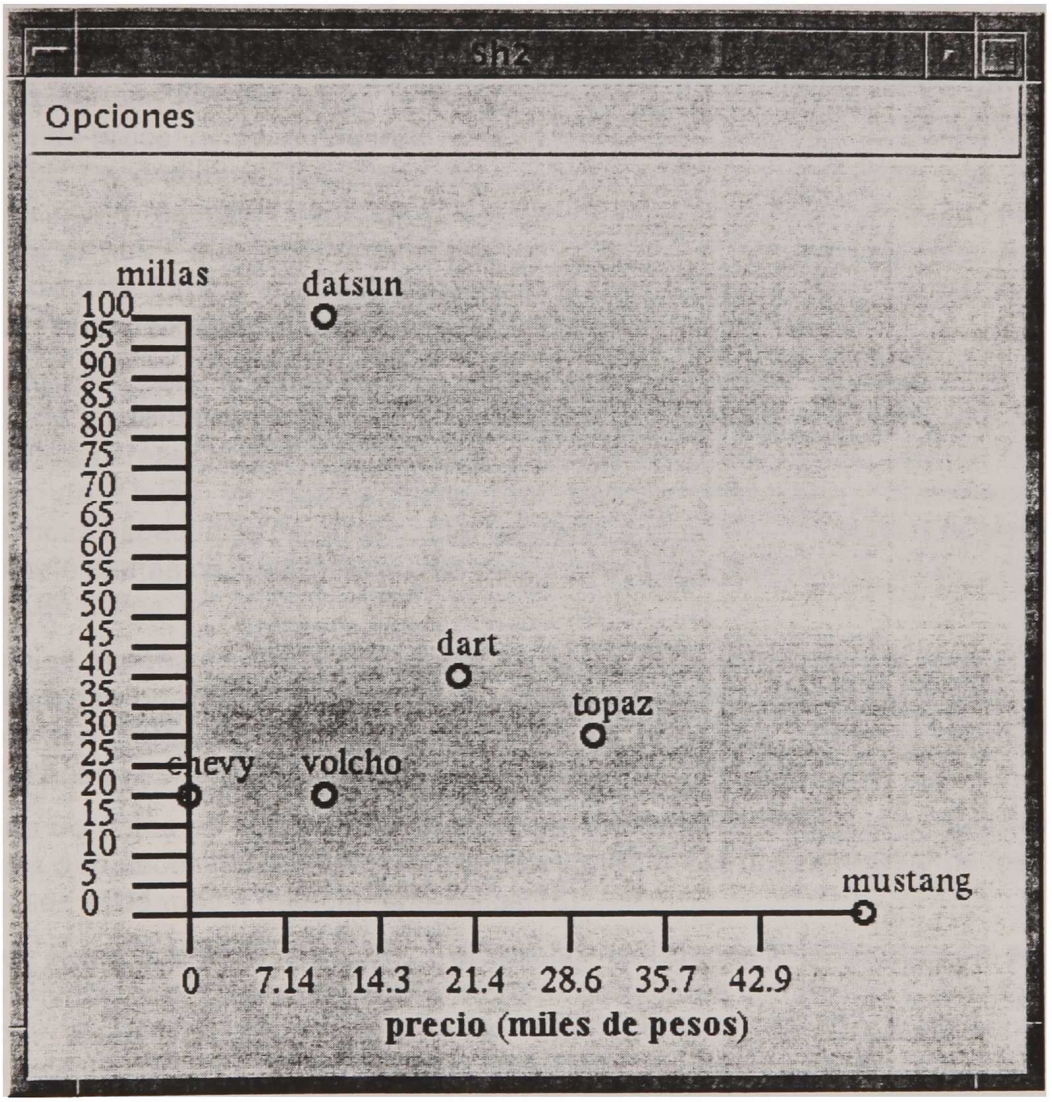


Figura 6.20.- Gráfica de Scatter Plot

6.3 CONCLUSIÓN DEL CAPÍTULO

A través de este capítulo se pudieron observar los diferentes tipos de visualización de datos que puede realizar el sistema DALI, así mismo se pudo observar la relación que existe entre el sistema LIDA y DALI. DALI complementa al sistema LIDA al representar en forma gráfica los resultados de las operaciones realizadas a través del flujograma, permitiendo al usuario una mejor visualización del comportamiento de los datos en el punto del flujograma que desee. Los resultados de las consultas u operaciones que el usuario realice a través de LIDA ya no sólo serán generados en forma de tabla sino que estarán representados gráficamente en el punto del flujograma que el usuario indique. Cada gráfica resultante puede servir en lo sucesivo para obtener reglas que puedan determinar el comportamiento de la información bajo ciertas condiciones u operaciones.

CONCLUSIONES

La relevancia de este trabajo radica en la metodología de graficación como un proceso automático de programación que genera gráficas a partir de datos.

La información de la base de datos puede ser vista a través de gráficos lo cual nos permite observar de una manera global la situación actual de la información y el comportamiento que presentan cuando se realiza alguna modificación sobre la misma.

Este es el principal objetivo de DALI: proporcionar una herramienta que permita visualizar la información a través de gráficas de acuerdo con las características de los datos

La interacción de DALI con el sistema LIDA puede concebirse como un sistema de simulación de procesos de información en donde se podrá observar el comportamiento de los datos dependiendo de los procesos que se ejecuten sobre ellos. Las gráficas generadas como resultado de estos procesos permitirán estudiar de una forma más sencilla los cambios que vaya presentando la información después de cada proceso, para que de esta forma, se puedan obtener patrones o reglas que permitan definir el comportamiento de los datos con base en un proceso específico.

La visualización de información es una herramienta valiosa para una mejor comprensión e interpretación de la información y es por ello que se puede utilizar como ayuda en otras áreas de investigación cuyo propósito es descubrir información implícita en Bases de Datos ya que a través de gráficas podemos ir observando las características específicas de un conjunto de datos o el comportamiento que sigue ese conjunto bajo ciertas circunstancias. Uno de los principales campos de investigación es la Minería de Datos.

Se denomina Minería de Datos al proceso de buscar y analizar grandes cantidades de datos. Haciendo una comparación con la minería tradicional, las grandes colecciones de datos son vetas potenciales de información disponible pero su búsqueda y extracción puede ser un proceso difícil y exhaustivo.

La minería de datos es como la exploración de una base de datos para identificar los fenómenos que necesitan ser observados, mostrando la estructura regular de los datos pero también ayudando a detectar anomalías[18].

La idea de Keim, Kriegel y Seidl [18] es utilizar las habilidades fenomenales del sistema de visión humano, el cual es capaz de analizar cantidades compactas o medianas de datos muy eficientemente. El sistema de visión humano, es capaz de reconocer inmediatamente patrones en imágenes lo cual sería muy difícil y en algunos casos casi imposible o consumiría mucho tiempo si se hace por computadora.

El reto es encontrar formas adecuadas de presentar datos multidimensionales para apoyar a los usuarios en el análisis e interpretación de datos.

Una de las metas es visualizar tantos conjuntos de datos como sea posible El único límite sería la resolución del dispositivo gráfico.

Explorando los datos de esta forma, el usuario puede aprender más acerca de los datos que realizando cientos de consultas a la base de datos.

La automatización de actividades en todas las áreas de negocios, ingeniería ciencia y gobierno, ha llegado a producir un creciente incremento en el flujo de datos así como también una gran cantidad de datos almacenados. Esto da la pauta para realizar sistemas que permitan descubrir información dentro de estas grandes bases de datos y poder realizar sobre ellos estudios sobre minería de datos determinando patrones de comportamiento de los datos y generando reglas.

El papel del sistema que incluye LIDA y DALI es integrar sus varias fuentes de datos y presentar visualizaciones que reflejen ciertos patrones, para que el investigador pueda observar distintos escenarios de la información y así determinar ciertas estrategias que resuelvan problemas planteados. Es posible descubrir información mediante la generación de reglas y la detección de anomalías basándose en la gran cantidad de datos. La simulación en LIDA es una poderosa estrategia de modelación para la planeación, predicción, ganancia de experiencia y la toma de decisiones.

En sistemas complejos, uno de los problemas es la gran cantidad de información. La visualización de datos es una forma de poder reducir la complejidad, reteniendo los elementos escenciales, encontrando relaciones escondidas y mejorando el entendimiento de los datos. De esta forma, la interpretación de la información y de los datos es un esfuerzo cooperativo entre el usuario/analista humano y la máquina. Utilizando la percepción visual se tiene un reconocimiento de patrones y análisis de tendencia pudiendo observar una gran cantidad de información en un período de tiempo corto.

BIBLIOGRAFÍA

[1] Kamran Parsaye., Chignel Mark, "Inteligent Database Tools & Applications" Willey & Sons, USA 1993.

[2] N.C.Shu, "Visual Programming" Van Nostrand Reinhold Company, New York 1988

[3] Sierra Romero Noe, "HeVICOP, Herramienta Visual para la Construcción de Programas", Tesis de Maestría, CINVESTAV IPN, 1995

[4] Chapa Vergara S., "Programación Automática a partir de Descriptores de Flujo de Información". Tesis doctoral. CINVESTAV –IPN, México D.F. 1991

[5] Schmid C., "Statistical Graphics: Design Principles and Practices", Willey, New York 1983

[6] Tukey John, "Methodology and the Statistician´s Response for Bolt Accuracy and Relevance" , Journal of Am.Statistics Assoc. 1974

[7] Lohse J., Rueter H., Biolsi K., Walker N. "Classifying Visual Knowlwdge Representations: a Foundation for Visualization Research" CH2913 /2 IEEE 1990

[8] Croxton F.E,. Cowden D.J., "Estadística General Aplicada", Fondo de Cultura Económica

[9] Wainer Howard, "On Multivariate Display", "Recent Advances in Statistics", Academic Press, 1983 Pag.487-499

[10] Bruckner Lawrence A. , "On Chernoff Faces", "Information Linkage Between Applied Mathematics and Industry" , Academic Press 1979 Pag. 19-35

[11] Bertin Jaques, "Graphics and Graphic Information Processing", Walter de Gruyter, Berlin- New York, 1981

[13] Shiguang Yu, "Visualización de una Base de Datos Espacial", Tesis doctoral CINVESTAV-IPN, México D.F 1997

[14] Alday Pedro, Tesis de Maestría CINVESTAV-IPN, México D.F. 1997

[15] Smith Milles J., "Database Abstractions:Aggregation and Generalization", ACM Trans. Database Syst. Vol. 1,No.3 Sept. 189-222

[16] Mackinlay Jock, "Automating the Design of Graphical Presentations of Relational Information", ACM Trans. On Graphics, Vol.5 No.2, April 1986 110-141

[17] Williams C., Rasure J., Hansen C., "The State of the Art of Visual Languajes for Visualizaton", IEEE, Computer Graphics and App. Vol2, 1992

[18] Keim Daniel A., Kriegel Hans_Peter, Seidl Thomas, "Supporting Data Mining of Large Databases by Visual Feedback Queries" IEEE, 1994 p.p.302-306

[19] Nye Adrian, "Xlib Programming Manual for version 11 of the Xwindow System" , O`Relly & Associates, Inc. 1990.

[20] Barkakati Nabajyoti, "Xwindow System Programming", SAMS, Macmillan Computer Publishing, USA 1991

[21] Brain Marshal, "MOTIF Programming: The Essentials ... and More" Digital Press, 1992

[22] Chapa Vergara Sergio V., Méndez Segundo Laura, "Un ambiente de visualización en bases de datos inteligentes", Memorias del Simposium Internacional de Computación: La computación en el proceso de globalización económica, Instituto Politécnico Nacional, Noviembre 1994

Los abajo firmantes, integrantes de jurado para el examen de grado que sustentará la **Lic. Laura Méndez Segundo** declaramos que hemos revisado la tesis titulada:

"DALI HERRAMIENTA PARA LA REPRESENTACION GRAFICA DE INFORMACION" consideramos que cumple con los requisitos para obtener el Grado de Maestro en Ciencias, con especialidad en Ingeniería Eléctrica.

Atentamente

Dr. Sergio V. Chapa Vergara _____

Dr. Pedro Mejía Alvarez _____

Dr. Manuel González Hernández _____