IMPLEMENTACION DEL SISTEMA DE MAPEO EN LA EMPRESA ELECTRICA DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ciencias y Humanidades

IMPLEMENTACION DEL SISTEMA DE MAPEO EN LA EMPRESA ELECTRICA DE GUATEMALA

MARCO TULIO ARANIVA

Modelo de trabajo profesional presentado para optar al título de Ingeniero en Ciencias de la computación en el grado de Licenciado.

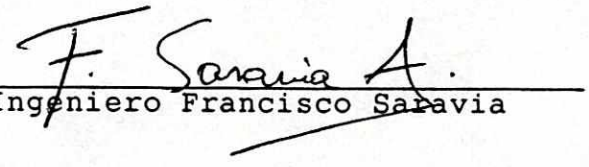
Guatemala

1990

Vo. Bo. :

(f) _____
Licenciado David Alvarez
Asesor

Tribunal :



(f) _____
Ingeniero Francisco Saravia

(f) _____
Licenciada Marta Julia Fernández

(f) _____
Licenciado David Alvarez

Fecha de aprobación : 5 de Octubre de 1990

A mis padres :

Marco Tulio Araniva
Ana Bersabé de Araniva

AGRADECIMIENTO

A los Ingenieros Eduardo Barrientos y Jorge Saravia, por la confianza depositada en mi persona para la elaboración de este proyecto. Al Licenciado David Alvarez, por su valiosa asesoría a lo largo de todo el proyecto.

RESUMEN

En este documento se presenta una descripción de cómo se ha implementado el sistema de mapeo en la Empresa Eléctrica de Guatemala. Primero se muestra lo que hace el sistema con sus diferentes opciones. Posteriormente se da una explicación de los diferentes componentes del sistema y cómo se relacionan entre sí.

Se tienen dos ambientes de trabajo. Uno lo constituye una minicomputadora marca "Data General", modelo MV-15000. El otro ambiente está dado por las estaciones de trabajo.

En la minicomputadora reside la base de datos técnica. Esta base (DG/SQL) utiliza el modelo relacional. Puede ser utilizada por dos programas de interfase elaborados en lenguaje "C". Un programa sirve para ejecutar "queries" dinámicos. El otro programa tiene compilados aquellos "queries" de uso más frecuente.

En las estaciones de trabajo residen los distintos planos del sistema. Cada estación de trabajo cuenta con una copia de ACAD. Se elaboraron programas en AUTOLISP con el objetivo de hacer la manipulación de los planos y de todos aquellos elementos que tengan una referencia geográfica. Se hicieron programas en "Turbo Pascal",

versión 5.5, para establecer comunicación entre los dos ambientes de trabajo. La comunicación se hace por medio de puertos seriales RS-232C.

Un usuario de este sistema de mapeo puede llegar a tener la impresión de que todo el trabajo se realiza solamente en ACAD, ya que es aquí donde se generan los distintos requerimientos de consulta o actualización a la base de datos, siendo el resto transparente. Es en ACAD donde se presentan los distintos resultados de los requerimientos.

Lo que presenta este trabajo es, basicamente, los procedimientos para hacer la interfase entre ambos ambientes.

CONTENIDO

		Páginas
I.	INTRODUCCION	1
II.	FUNCIONAMIENTO DEL SISTEMA	4
III.	VISTA GLOBAL DEL SISTEMA	17
IV.	AMBIENTE DATA GENERAL	21
	A. Organizacion de la base de datos	22
	B. Programas de interfase	25
V.	ESTACION DE TRABAJO	36
	A. Comunicaciones	36
	B. Gráficas	39
	C. Rutinas es Autolisp	51
VI.	CONCLUSIONES Y RECOMENDACIONES	62
VII.	BIBLIOGRAFIA	64
	APENDICES	
	A. Codificación de los postes	65
	B. Descripción de la base de datos	67
	C. Queries utilizados	71
	D. Comunicaciones	83
	E. Estaciones de trabajo	86
	F. Rutinas en Autolisp	88

LISTA DE FIGURAS

		Página
1	Primera pantalla	4
2	Detalle geográfico	5
3	Menú de los diferentes elementos del sistema	6
4	Ventana del sistema	7
5	Menú postes	8
6	Tipos de postes	9
7	Traslación del texto de los postes	9
8	Presentación de los postes en una ventana	10
9	Menú de anclas	11
10	Tipos de anclas	11
11	Anclas en una ventana	12
12	Menú de transformadores	13
13	Transformadores en una ventana	14
14	Menú de líneas primarias	14
15	Segmento de línea primaria en una ventana	15
16	Segmento de línea secundaria en una ventana	16
17	Vista global	17
18	Línea Acad	40

19 Línea Secundaria 1 49

20 Línea Secundaria 2 50

21 UTM 1 65

22 UTM 2 66

I. INTRODUCCION

En la Empresa Eléctrica de Guatemala surgió hace algún tiempo la necesidad de tener un inventario actualizado de los diferentes elementos del sistema de distribución, como también una localización exacta de éstos. Para conseguirlo se decidió levantar el mapeo de la zona de servicio.

Un sistema de mapeo puede ser definido como aquel que se compone de una parte geográfica y una parte técnica, visto como una sola unidad donde los cambios en un sector afectan al otro.

Los sistemas de mapeo son utilizados en todo el mundo, en áreas muy diversas como la militar, comunicaciones, eléctrica, salud, demográfica, etc., proponiéndose todos el mismo objetivo : tener una localización geográfica de la información para ser utilizada con diferentes fines.

En un principio se tenía la duda de si resultaría mejor comprar un paquete de mapeo, o bien elaborar uno propio. Al final, la decisión fue elaborar el paquete, siendo las principales razones el costo y el hecho de contar con los recursos necesarios, tanto humanos como materiales, para la realización de este proyecto. El proyecto, cuando se encuentre más avanzado y se tenga actualizado, tendrá una utilidad de amplias proporciones, ya que dará un estado de la existencia de los diferentes elementos y de la ubicación precisa de éstos.

Los elementos que se han considerado para este proyecto son los siguientes : postes, anclas, bancos de transformadores, líneas primarias y líneas secundarias. Con base en la información existente se podrá darla acerca de la carga de los transformadores que se encuentran en cierta zona o la extensión en kilómetros del recorrido de una línea primaria, sólo por mencionar algunas aplicaciones a manera de ejemplo. Se pretende que teniendo la base hecha, se pueda ir sumando nueva información a la base de datos, como por ejemplo equipo de protección, contadores, etc, y nuevas aplicaciones (estudios eléctricos e información del sistema) al uso de la información.

La implementación de este proyecto abarca una buena cantidad de recursos para realizar las diferentes actividades. Se tiene a personas para levantar la información del sistema, tanto técnica como geográfica. Posteriormente la información técnica es ingresada a la computadora por digitadores, mientras que la información geográfica (los mapas y diferentes elementos) se ingresa por medio de digitalizadores, los cuales son tabletas para el ingreso de mapas o dibujos a la computadora. Tanto la información técnica como la geográfica residen en máquinas o ambientes de trabajo distintos, razón por lo cual es necesaria su integración.

Este trabajo presenta los procedimientos y herramientas que se han utilizado para la implementación del proyecto hasta su

estado actual en el área de computación y haciendo mayor énfasis en la implementación de los programas de interfase entre información geográfica e información técnica. El proyecto se encuentra en los inicios del ingreso de la información, tanto gráfica como técnica. Se continúan elaborando nuevas rutinas para ingresar el resto de elementos no considerados hasta el momento. El ingreso de la información pretende finalizarse en dos años.

Se presenta una visión global de la operación del sistema y de las relaciones que existen entre las diferentes partes.

Lo que se tiene en este momento son los mecanismos para la actualización de la información y la presentación de ésta. Es decir lo que se ha hecho es la base del sistema de mapeo con la idea de irle añadiendo aplicaciones (estudios eléctricos e información del sistema) en el futuro.

II FUNCIONAMIENTO DEL SISTEMA

Toda sesión al sistema de mapeo se hace por medio de una estación de trabajo, la cual es una microcomputadora de gran velocidad de procesamiento de información como también de alta calidad en la presentación de ésta. La primera pantalla que se presenta en el sistema es la siguiente (fig 1).

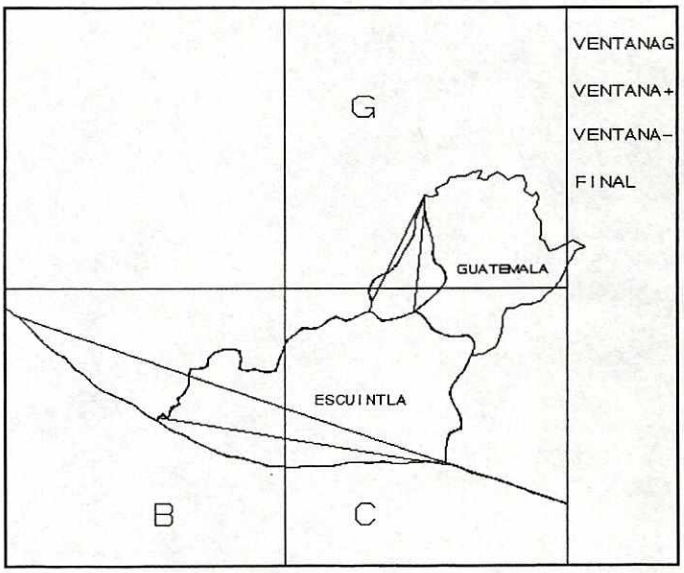


Figura 1

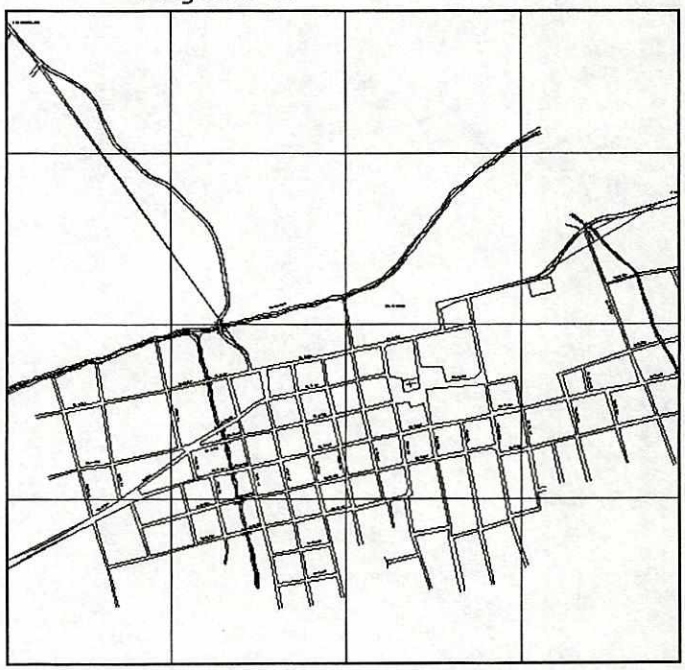
El mapa que se muestra corresponde a la zona de servicio de la Empresa Eléctrica, la cual incluye los departamentos de Guatemala, Escuintla y Sacatepéquez.

Con base en el menú que aparece a la derecha del mapa de la zona de servicio de la Empresa Eléctrica, Se puede hacer una ventana[1] sobre cualquier porción del mapa. La primera de las opciones sirve para escoger el detalle geográfico del área limitado por la ventana. Las opciones VENTANA+ y VENTANA- sirven respectivamente para hacer acercamientos o regresar a

[1]Una ventana es un área rectangular de igual o menor tamaño que la pantalla.

la ventana anterior. Por ejemplo, del dibujo que contiene el área de servicio de la Empresa Eléctrica se puede escoger, utilizando un "mouse", un digitalizador o el teclado para seleccionar los puntos de interes, la siguiente área geográfica (figura 2).

Figura 2



Este detalle geográfico se encuentra almacenado en dibujos elaborados utilizando un CAD(Computer Aid Design), cada dibujo contiene un área de 2 x 2 kilómetros.

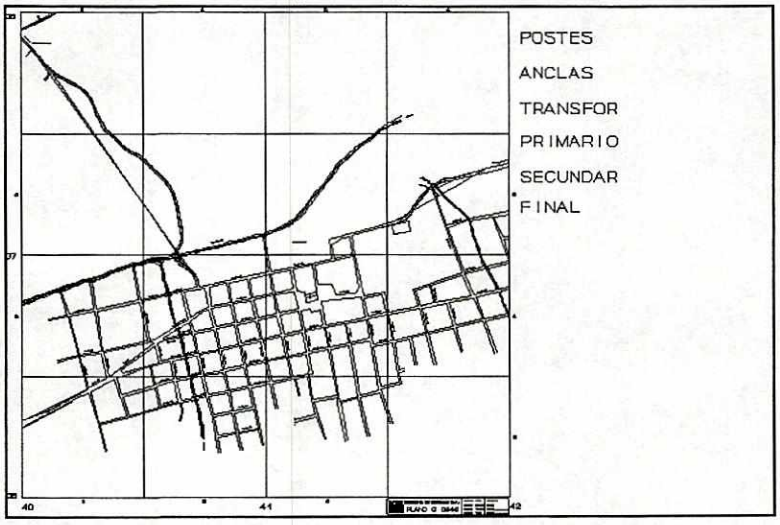
Las letras que se observaron en el plano principal ("B", "C" y "G" en la figura 1) identifican un área de 100 x 100 Kms. Estas letras sirven de punto inicial para la construcción de un código que utiliza el sistema de coordenadas UTM[2], las

[2]Ver apéndice A

cuales se emplean para la codificación de los postes (el elemento de mayor importancia del sistema de distribución para este proyecto), como también para la elaboración de los planos.

Despues de haber escogido la zona geográfica de interés, se muestra el siguiente menú (fig. 3).

Figura 3



Estas opciones sirven para actualizar o seleccionar los diferentes elementos del sistema de distribución considerados en este proyecto.

En los planos no existe más información que la geográfica, es decir las ciudades, carreteras, fincas, aldeas, etc. Toda la información del sistema, tanto técnica como de ubicación de los elementos, se encuentra almacenada en una base de datos residente en una minicomputadora[3]. Esto nos da dos ambientes de trabajo bien definidos :

[3]Categoría en la que cae la computadora utilizada que contiene la base de datos.

- Minicomputadora
- Estación de trabajo

Dichos ambientes se enlazan mediante programas implementados en ambas máquinas.

El único elemento que tiene una ubicación real en los planos es el poste, estando el resto de elementos ubicados en éste o en referencia a éste. La manera de ubicar al poste en el dibujo es mediante sus coordenadas reales en el plano. Para identificar cada poste se utilizaron las coordenadas UTM, más un número de poste dentro del área definida por dichas coordenadas.

Cualquier requerimiento que el usuario haga, tanto de consulta como de actualización, es transmitido a la minicomputadora, la cual devuelve los resultados a la estación de trabajo. De esta manera se pueden ingresar, borrar o seleccionar los diferentes elementos.

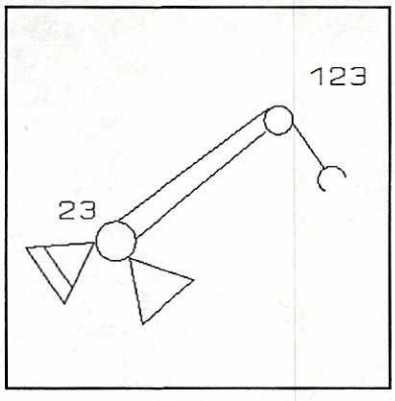


figura 4

En la figura 4 se muestra un segmento del sistema de distribución. Este segmento consiste de dos postes (número 23 y 123), los cuales se encuentran conectados por dos líneas. La línea superior se refiere a una línea del secundario y la línea de abajo se refiere al primario. Los

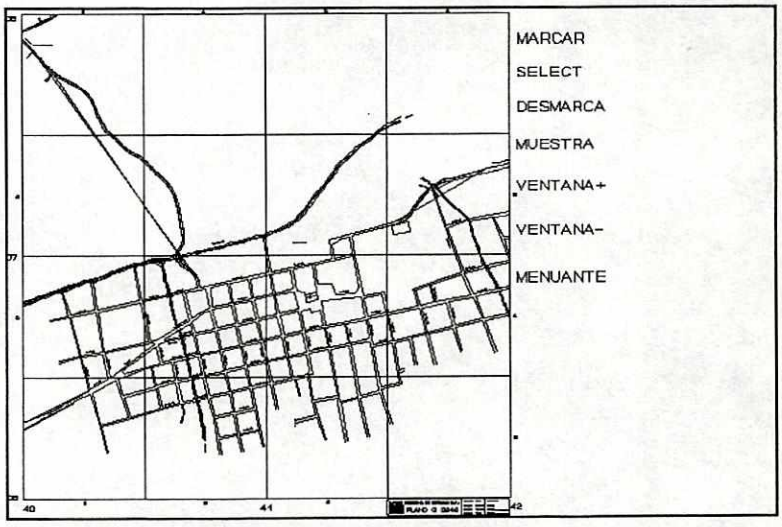
triángulos que se observan se refieren a los transformadores. El poste de la izquierda contiene a dos de éstos. El transformador de la izquierda es de fuerza motriz y el otro es convencional.

A continuación de describen las diferentes opciones existentes para cada uno de los elementos.

A. POSTES

Al escoger la opción de postes se muestra un menú, el cual se puede apreciar en la figura 5.

Figura 5

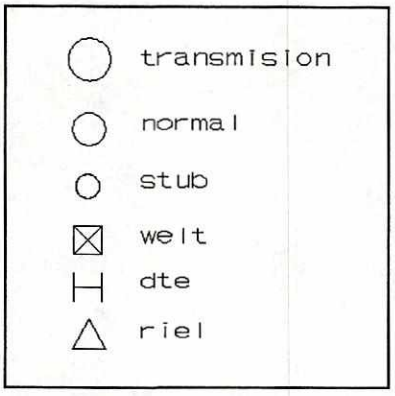


1. Marcar

Esta opción sirve para marcar un poste en el plano.

Existen seis tipos diferentes de postes (fig. 6). El poste[4] puede ubicarse en cualquier posición al igual que su número. Los postes WELT, DTE y riel pueden rotarse a partir del punto

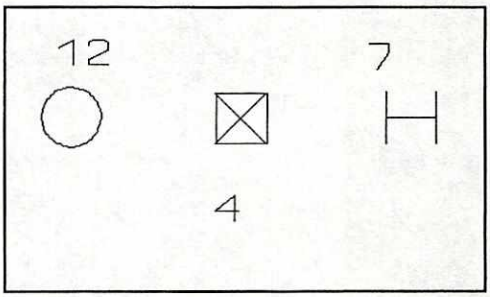
[4]La ubicación de todos los elementos puede hacerse tanto con un MOUSE como con el teclado.



de inserción.

Figura 6

Figura 7



En la figura 7 se presenta un ejemplo de la ubicación de los postes con su respectivo número.

Al marcar un poste o cualquier otro elemento se origina un llamado a la base de datos en la minicomputadora para actualizarla y ésta devuelve un resultado indicando si la operación fue exitosa o no.

2. Select

Se muestra la información técnica de un poste.

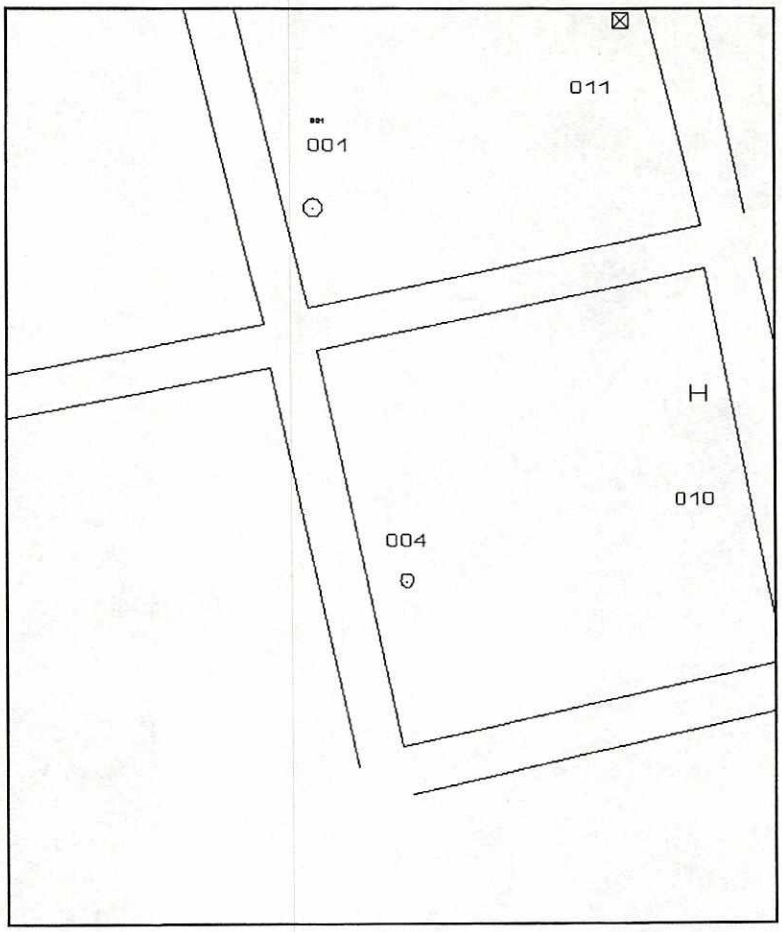
3. Desmarcar

Se selecciona un poste en el plano, poniendo en cero su ubicación en la base de datos para que éste pueda ser reubicado si así se desea.

4. Muestra

Se muestran todos los postes existentes en el área definida por una ventana. Un ejemplo de esto se aprecia en la figura 8.

Lo que se aprecia aquí es una porción del plano de una



ciudad, la cual contiene cuatro postes de distinto tipo.

Figura 8

B. ANCLAS

Al escoger la opción de anclas se muestra el siguiente menú (Fig. 9) .

Las anclas sirven para sujetar el poste cuando no existe un equilibrio de fuerzas, producido por las diferentes tensiones de los cables. Las opciones de este menú se explican a continuación.

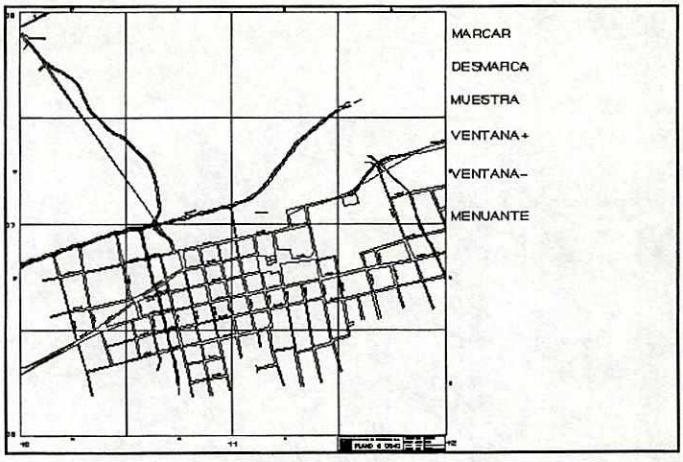


Figura 9

1. Marcar

Esta opción sirve para poner un ancla en un poste. Un poste puede tener más de un ancla, de las cuales hay ocho tipos (Ver figura 10).

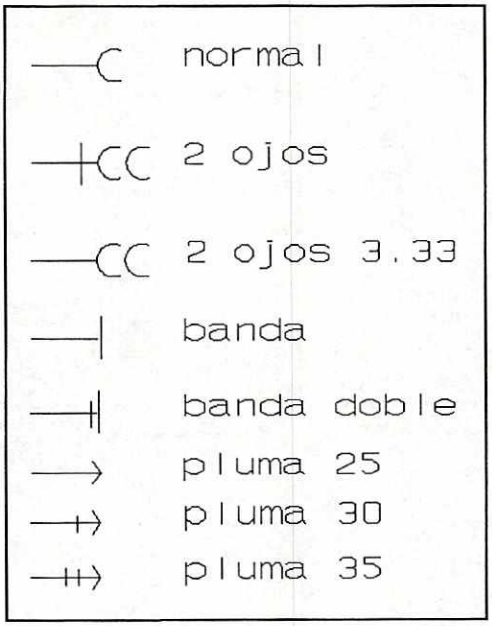


Figura 10

2 Desmarcar

Esta opción sirve para eliminar un ancla de un poste.

3 Muestra

Esta opción sirve para mostrar todas las anclas que están comprendidas en una ventana (Fig 11).

En esta porción del plano, el poste '001' tiene un ancla de 2 ojos. El poste '011' tiene un ancla normal y una pluma de 25 pies. Toda la información acerca del tipo de ancla y de su ángulo de rotación está almacenada en la base de datos técnica.

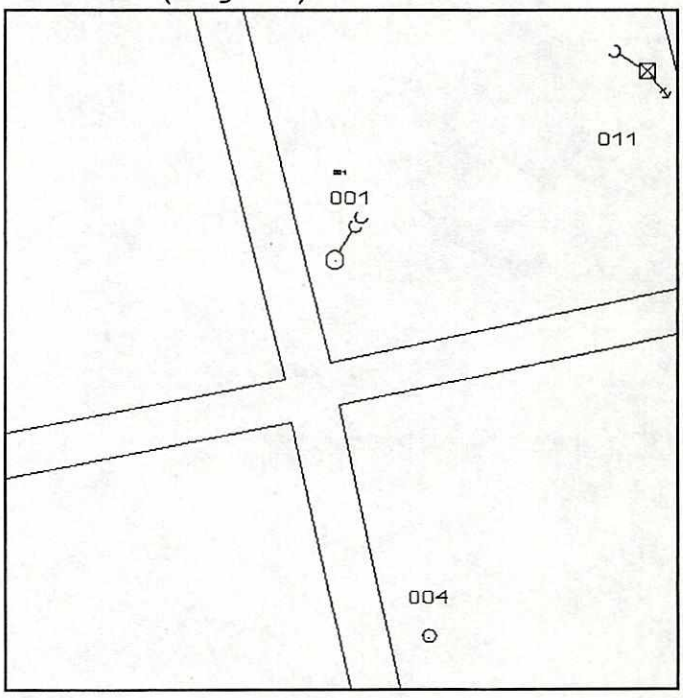


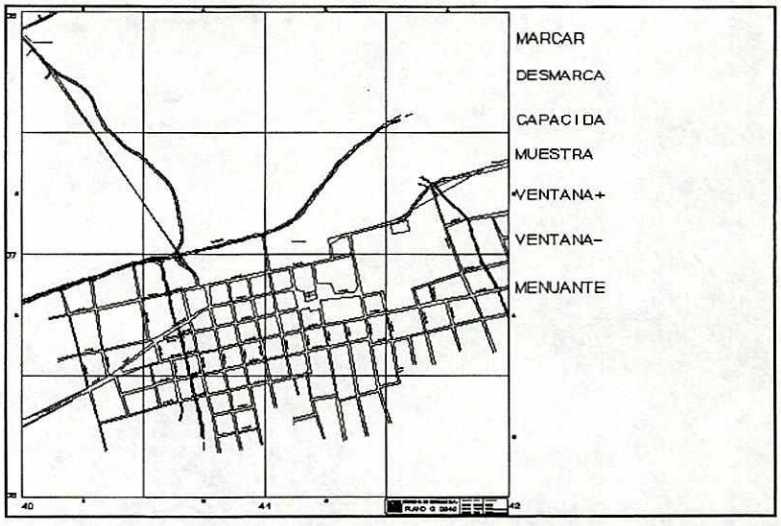
Figura 11

C. TRANSFORMADORES

Al escoger esta opción se presenta el siguiente menú (Fig 12).

1. Marcar

En un poste se pueden marcar de uno a tres transformadores. Estos forman lo que se conoce por banco de transformadores.



El sistema pregunta por el número de identificación del banco, el número y capacidad (medida en Kilovatios) de cada transformador.

2. Desmarca

Esta opción sirve para eliminar los transformadores de un poste(el banco completo), en lo referente a su localización, ya que la información técnica del transformador no se elimina de la base de datos.

3. Capacidad

Se suma la carga de cada transformador comprendido en una ventana y se presenta este resultado.

4. Muestra

Se muestran los bancos de transformadores comprendidos en una ventana. Ver ejemplo en la figura 13.

Los triángulos que se observan en el poste '010' forman un banco de transformadores. Al igual que las anclas, se guarda en la base de datos la información referente a la localización de los transformadores. Como se observa en el poste 10, hay dos transformadores (los cuales forman un banco).

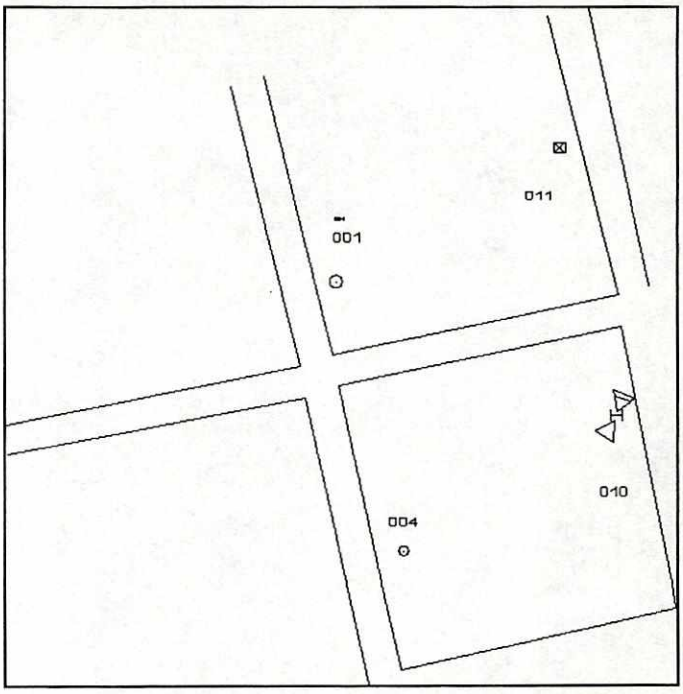


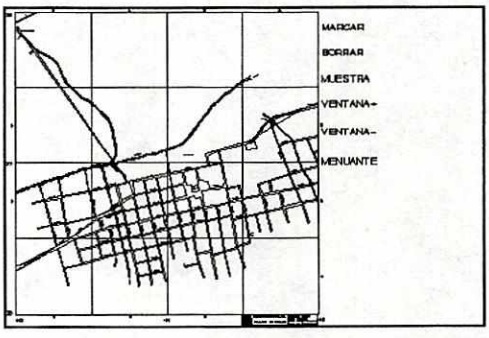
Figura 13

El transformador de la izquierda es convencional, en cambio el otro es de fuerza motriz.

D. Líneas Primarias

Al escoger esta opción se presenta el siguiente menú

Figura 14



1. Marcar

Se marca un segmento de línea entre dos postes

2. Borrar

Se elimina el segmento de línea escogido

3. Muestra

Se muestran todos los segmentos de línea primaria comprendidos en una ventana. A continuación se presenta un ejemplo. Ver figura 15.

Cada segmento de línea se dibuja del centro del poste origen al centro del poste destino.

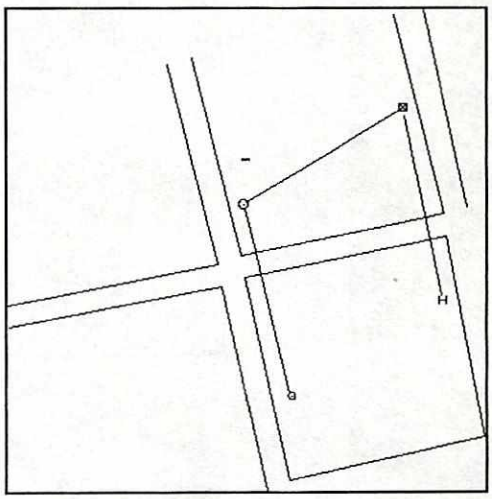


Figura 15

E. Líneas secundarias

El menú de líneas secundarias es igual al de líneas primarias. Las opciones se describen a continuación.

1. Marcar

Se dibuja un segmento de línea secundaria entre dos postes.

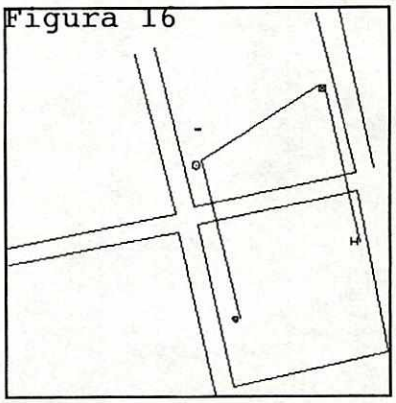
2. Borrar

Se borra un segmento de línea secundaria.

3. Muestra

Se muestran los segmentos de línea secundaria comprendidos en una ventana. Ejemplo en la figura 16.

Figura 16



A simple vista pudiera decirse que no hay diferencia entre la línea primaria y la línea secundaria. El problema es que a causa de lo limitado de la gráfica de la figura no se aprecia bien que la línea secundaria va a la parte superior de los postes que conecta. Además, la principal diferencia entre los dos tipos de línea es su presentación, pues se dibujan con diferentes tipos de línea (tipos de líneas de ACAD).

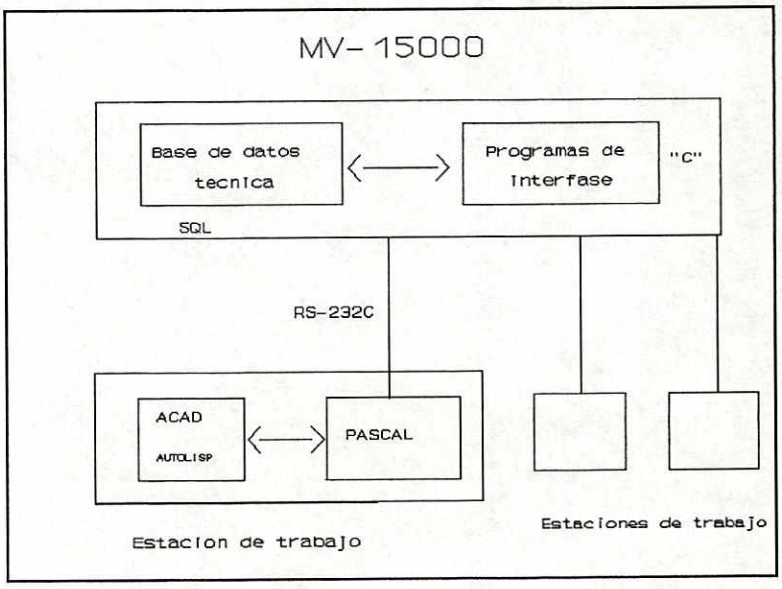
III VISTA GLOBAL DEL SISTEMA

Cuando un usuario selecciona alguna de las opciones mencionadas anteriormente, el sistema construye un "string" (cadena de caracteres) que se transmite de la estación de trabajo a la minicomputadora. Este "string" contiene un requerimiento que es procesado en la minicomputadora, la cual devuelve los resultados a la estación de trabajo. Cómo ya se mencionó anteriormente, existen dos ambientes de trabajo bien definidos.

- Minicomputadora
- Estaciones de trabajo

Una visión general del sistema puede apreciarse en la figura 17.

Figura 17



Cada estación de trabajo tiene instalado el ACAD y el programa de comunicaciones. El ACAD sirve para elaboración de dibujos y planos, así como para la presentación y manipulación de los mismos. Los planos se ingresan a la

computadora por medio de tabletas de dibujo (digitalizadores). Estos planos solamente pueden ser elaborados por personas con autorización para ello. Un usuario de mapeo nunca dibujará un plano, ni podrá modificarlo. Solamente puede editarlos : ingresar, consultar o borrar de los mapas los diferentes elementos del sistema. Los elementos se almacenan solamente en la base de datos técnica, y nunca pasan a formar parte definitiva de un plano. El lenguaje para manipular toda la información en ACAD es AUTOLISP [2], una versión de LISP [11], el cual proporciona un control sobre ACAD.

Todos los planos del sistema de distribución se encuentran en cada una de las estaciones de trabajo[5]. Cuando un usuario crea una ventana sobre el mapa de la zona de servicio (el primer mapa que el sistema muestra), una rutina elaborada en AUTOLISP carga los mapas que se encuentran en dicha ventana. Cuando el usuario hace algún requerimiento, éste se transmite a la computadora central (el otro ambiente de trabajo) por medio de un programa de comunicaciones elaborado en Pascal.

El "requerimiento" que se manda hacia la computadora central tiene el objetivo de extraer o de actualizar información en la base de datos técnica. Esta base de datos está diseñada para trabajar el modelo relacional y utiliza los

[5]Así se pensó originalmente, pero actualmente se está trabajando para tenerlos en un "server", en una red.

estándares de SQL[3][6] (Structured Query Language). Toda la información en la base de datos está organizada en tablas. Una tabla es una estructura que almacena información relacionada en forma de entidades, donde cada entidad se identifica en forma única. Por ejemplo, En la tabla de postes se tiene que cada entidad está identificada por el número de poste. Para cada número de poste la tabla contiene información sobre su ubicación, el tipo de poste, material, etc. En la computadora central existe un programa que tiene como función específica procesar estos requerimientos y devolver los resultados correspondientes.

Los resultados se regresan en un formato de listas, apropiado para ser manipulado por el interprete AUTOLISP. Una lista tiene la siguiente sintaxis

```
lista      --> "(" elemento ")"
elemento   --> dato [ elemento ]
dato       --> numero ¦ "string" ¦ nulo ¦ lista
```

Un ejemplo de una lista con el código de un poste, su tipo y sus coordenadas sería la siguiente :

```
( "B1248C007" ( 12070.88 48174.7) "B" ) )
```

AUTOLISP procesa esta lista y hace una presentación gráfica de los resultados, de acuerdo al símbolo definido por "B" y en la posición indicada por las coordenadas.

---

[6]Ver Apéndice B

En las próximas páginas se hace una descripción detallada de los ambientes que componen el sistema.

IV AMBIENTE DATA GENERAL

La computadora central, la cual es marca DATA GENERAL modelo MV-15000, contiene la base de datos técnica. Esta se puede utilizar de dos maneras, mediante interfases con varios lenguajes de programación.

- Utilizando el precompilador
- Utilizando el HLI (Host Language Interface)

El precompilador se utiliza para procesar aquellos "queries"[7] de los que se conoce de antemano su estructura y cuyas únicas variaciones están en los valores de las variables consideradas.

El otro tipo de interfase es el HLI, con el que se pueden ejecutar "queries" definidos en el momento de la corrida del programa.

Ambos tipos de interfase tienen sus ventajas. El uso del precompilador da como resultado mejores tiempos de respuesta que el HLI, pero tiene la limitación de poder ejecutar solamente lo que se ha definido con anterioridad.

En el sistema de mapeo se presentan estas dos situaciones mencionadas, por lo que se implementaron dos programa en el lenguaje "C" : el primero utiliza el precompilador (nombre del

---

[7]Un "query" es una instrución que sirve para hacer una consulta o una actualización en la base de datos.

programa: queries) y el segundo el HLI (nombre del programa: prog_hli).

Estos programas se describirán más adelante.

A. ORGANIZACION DE LA BASE DE DATOS

Se tienen las siguientes tablas[8] :

- Postes
- Transformadores
- Bancos
- Anclas
- Lineas_primarias
- Lineas_sec
- Trafos_posicion
- Detalle_lineas
- Detalle_lineassec

En el diseño de las tablas se aplicaron las tres primeras reglas de normalización de bases de datos relacionales [5], llevándolas a la forma normal Boyce-Codd para evitar las anomalías que se pueden producir al no hacerlo [5]. En el diseño se tuvo el cuidado de proveer el mayor grado de integridad en la base[10]. Se construyeron las referencias necesarias.

1. Postes

Esta tabla, contiene, entre otros campos, los siguientes: p_numero, p_coordexx, p_coordeyy. Estos tres campos son de suma importancia para el sistema de mapeo. El campo p_numero

[8]Ver Apéndice B

da el código del poste (UTM + número de poste) y los campos p_coordexx y p_coordeyy dan la ubicación del poste en el plano con un decimal de aproximación.

El número de poste está contenido en las otras tablas, para relacionar los otros elementos con su poste correspondiente, con el objeto de especificar la localización geográfica de dichos elementos.

2. Bancos

Esta tabla contiene información acerca de los diferentes bancos de transformadores del sistema. Por medio del campo b_poste, se relaciona el banco con la entidad de postes correspondiente. En esta tabla también se almacenan las coordenadas "X" y "Y" (en los campos b_coordexx y b_coordeyy, respectivamente) para la ubicación geográfica de la información del banco por representar en el plano.

3. Transformadores

En esta tabla se almacena la información técnica de los transformadores del sistema de distribución. Aquí no se tiene una ubicación exacta de los transformadores en el poste que se encuentran (la relación entre el transformador y el poste correspondiente se hace por medio de la tabla BANCOS. Cada transformador hace referencia a su banco correspondiente por medio del código de banco). Esto no es un problema, ya que en la tabla TRAFOS_POSICION se encuentra el tipo y el ángulo de

rotación de cada transformador. La tabla TRAFOS_POSICION está relacionada a la tabla BANCOS por medio del número de banco. No se conoce que número de transformador está ubicado en determinada posición, ya que esto carece de importancia.

4. Anclas

En esta tabla se almacenan las diferentes anclas existentes (hay un total de ocho tipos). Cada ancla se encuentra relacionada a un poste por medio del código de poste. Uno de los campos de esta tabla es el ángulo de rotación del ancla, ya que interesa saber la orientación de ésta.

5. Líneas_primarias

Aquí se tiene una relación entre un poste y otro, dada por la información de un segmento de línea primaria. El detalle de las diferentes fases de la línea se encuentra en la tabla DETALLE_LINEASPRI.

6. Líneas_secundarias

Al igual que en la tabla LINEAS_PRIMARIAS, en esta tabla se almacena la información de un segmento de línea secundaria. El detalle acerca de los diferentes conductores se encuentra en la tabla DETALLE_LINEASPRI.

B. PROGRAMAS DE INTERFASE

Estos programas se utilizan para recibir los requerimientos desde las estaciones de trabajo, procesar estos requerimientos, actualizar o consultar la base de datos y regresar resultados a la estación de trabajo.

1. Queries

Este programa recibe de la estación de trabajo un "string", el cual contiene el tipo de "query" por ejecutar y los parámetros (los cuáles varían de acuerdo al requerimiento) que se le pasan a éste.

El "string" tiene un formato así : OD ,
donde "O" es un "substring" que indica la rutina para ejecutar el "query" y "D" es un "substring" que trae los valores de los campos que serán actualizados o consultados en las tablas respectivas. El "substring" "O" puede tener 2 ó 3 caracteres. El primero sirve para construir el siguiente case.

```
CASE O[1]  BEGIN
"p", "P" : CALL postes(OD)
"a", "A" : CALL anclas(OD)
"t", "T" : CALL transformadores(OD)
"l", "L" : CALL lineas(OD)
END
```

El segundo carácter de "O" sirve para indicar el tipo de

"query" por ejecutar (select, update, insert). Con las líneas, sin embargo, el segundo carácter indica si es una línea primaria o secundaria y el tercero da el tipo de "query".

En las próximas páginas se hace una descripción de los procedimientos que se llaman en el CASE descrito anteriormente.

a. Postes

Este procedimiento tiene el siguiente case :

```
CASE O[2] BEGIN
   0 : CALL pmarca(D)
   2 : CALL pselect(D)
   4 : CALL pdesmarca(D)
   6 : CALL pobtiene(D)
 END CASE
```

Estos procedimientos se desciben a continuación :

a.1 Pmarca

El "string" "D" tiene el siguiente formato :

CCCCCCCCCXXXXXXXXYYYYYYYYWWWWWWWWZZZZZZZZT

Los nueve primeros caracteres contienen el código del poste. Las "X" y las "Y" dan el punto de inserción o coordenadas del poste en el plano de ACAD (son seis dígitos enteros, un decimal); "W" y "Z" dan el punto de inserción del código del poste en el mismo plano. La "T" da el tipo de poste

que se ingresó (hay seis diferentes tipos de poste).

Antes de ingresar el registro a la tabla "POSTES", se verifica si ya existe un registro con el mismo código de poste. Si existe y tiene los valores en cero, se hace una actualización del poste. En caso que el poste no exista, éste se grabará.

En caso que se inserte o se actualice el poste, se devolverá a la estación de trabajo el siguiente string : "(p_numero)", la cual es una lista no nula para AUTOLISP. En caso contrario se devolverá una lista nula ("()"), la cual indicará que la operación no fue exitosa, eliminándose del mapa el poste recién marcado.

a.2 Pselect

EL "string" "D" tiene el siguiente formato :

CCCCCCCCC

donde las C's dan el código del poste para así seleccionar todos los campos del registro correspondiente en la tabla postes. El resultado se regresa en el siguiente formato :
((p_numero CCCCCCCCC) (p_direccio "FFFFFFFFFFFFFFFF") ... (...) (...) ... (p_textox DDDDDDDD))
para ser procesado por AUTOLISP.

a.3 Pobtiene

El string "D" tiene el siguiente formato :

CCCCCCCCDDDDDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

donde las "C" dan el código UTM, definido por el punto inferior izquierdo, las "D" dan el punto superior derecho de la ventana definida por el usuario en ACAD; las "X" y "Y" dan el punto inferior izquierdo y las "Z" y "W" dan el punto superior derecho de la ventana seleccionada.

Sería cierto afirmar que las UTM, para este tipo de "query", están de más, ya que con los dos punto(punto inferior izquierdo y punto superior derecho de la ventana) bastaría para obtener los postes solicitados, pero el tiempo de respuesta del "query" no fuera el óptimo, ya que no se estaría tomando en cuenta al único índice definido para la tabla (el definido por la llave : p_numero).

a.4 Pdesmarca

EL "string" que se recibe es exactamente igual al de PSELECT. Este procedimiento tiene como función poner en cero los campos que dan la ubicación geográfica del poste y de su código. Así éste podrá ser colocado posteriormente en una nueva posición.

b. Anclas

Este procedimiento tiene el siguiente case :

```
CASE O[2] BEGIN
   0 : CALL amarca(D)
```

```
   4 : CALL adesmarca(D)
   6 : CALL aobtiene(D)
END CASE
```

Estos procedimientos se desciben a continuación :

b.1 Amarca

El "string" "D" tiene el siguiente formato :

CCCCCCCCCRRRT

Los nueve primeros caracteres contienen el código del poste, las "R" dan el ángulo de rotación del ancla en el poste y la "T" da el tipo de ancla ("A" - "I"). En la práctica, un poste tendrá a lo sumo tres anclas, pero en la tabla hay lugar para 360 anclas, ya que la llave de esta tabla está formada por el código del poste, más el ángulo.

b.2 Adesmarca

En este procedimiento se borra un ancla de la tabla ANCLAS. para hacer esto el "string" que se recibe tiene la siguiente forma :

CCCCCCCCCRRR

Los nueve primeros caracteres contienen el código del poste y las "R" dan el ángulo de rotación del ancla en el poste.

b.3 Aobtiene

Este procedimiento se ocupa para seleccionar todas las

anclas que están comprendidas en una ventana definida en ACAD. El "string" que se recibe tiene el siguiente formato :

CCCCCCCCCDDDDDDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

El cual es idéntico al del procedimiento pobtiene, pero se regresan resultados distintos.

c. Bancos y transformadores

Este procedimiento tiene el siguiente case :

```
CASE O[2] BEGIN
   0 : CALL tmarca(D)
   4 : CALL tdesmarca(D)
   5 : CALL t_trafos(D)
   6 : CALL tobtiene(D)
   7 : CALL tcapacidad(D)
END CASE
```

Estos procedimientos se describen a continuación :

c.1 Tmarca

Esta opción sirve para ingresar o actualizar un banco (según sea el caso) e ingresar las posiciones de los transformadores que contiene.

El "string" "D" tiene el siguiente formato :

CCCCCCCCCNNNNNNNNNNXXXXXXXXYYYYYYYYAAAT-AAAT-AAAT

Las "C" dan el código del poste al cual se halla atado el banco, las "N" definen el número de banco. Este número se

divide en dos, ya que viene de la siguiente forma :

R-B , donde "R" es el número de circuito y "B" es el código del banco para ese circuito. Tanto "R" como "B" son "strings" de tamaño variable.

Las "X" y las "Y" dan la ubicación en el plano del punto de inserción del bloque que contiene solamente información textual del banco, del cual se hablará en su oportunidad cuando se describan las funciones elaboradas en AUTOLISP.

Las "A" y la "T" dan el ángulo de rotación de los transformadores en el poste. Puede ser que el banco tenga uno, dos o tres transformadores.

Si los campos que tienen el punto de inserción del poste no se encuentran en cero en la tabla BANCOS, la transacción no se efectuará.

c.2 Tdesmarca

Este procedimiento recibe el número de banco, pone en cero los campos que dan la ubicación del texto del transformador. Se eliminan los registros correspondientes en la tabla TRAFOS_POSICION.

c.3 T_transf

Este procedimiento da como resultado los transformadores de cada poste que se encuentra en una ventana definida en

ACAD. El "string" que se procesa en este procedimiento es igual al que se procesa en aobtiene, pobtiene o tobtiene. Para la ubicación de los transformadores se ocupa el mismo criterio que las anclas, es decir su origen está en uno de los puntos del círculo definido por el radio del tipo de poste. La ubicación de los transformadores está en la tabla TRAFOS_POSICION, para hacer la selección de la información se tocan las siguientes tablas: POSTES, BANCOS y TRAFOS_POSICION.

c.4 Tobtiene

Este prodedimiento da como resultado una o más "listas" de LISP [11], en las cuales se incluyen los transformadores de un banco, junto con sus capacidades y números de compañía así como su ubicación. El "query" que selecciona la información utiliza tres tablas : TRANSFORMADORES, POSTES, BANCOS. Una de las condiciones para seleccionar información es que los campos que contienen el punto para ubicar el texto en el plano sean mayores que cero.

Realmente, las rutinas "tobtiene" y "t_transf" se llaman en ese mismo orden desde la opción de selección de bancos en ACAD. Pueden ser vistas como complementarias una de otra.

c.5 T_capacidad

El "string" que recibe esta rutina es igual al de "tobtiene", el resultado que manda es la suma de las

capacidades (carga de los transformadores) en una determinada área.

d. Líneas

Este procedimiento tiene el siguiente case :

```
CASE O[2] BEGIN
   'p', : CALL lineas_primarias(D)
   's' : CALL lineas_secundarias(D)
 END CASE
```

d.1 Líneas primarias

Este procedimiento tiene el siguiente case :

```
CASE O[2] BEGIN
   '0', : CALL lpri_marca(D)
   '4' :  CALL lpri_borra(D)
   '6' :  CALL lpri_muestra(D)
END CASE
```

d.1.1 Lpri_marca

Este procedimiento recibe un "string" que trae la información referente a un segmento de línea entre dos postes. El "string" tiene el siguiente formato :

```
AAAAAAAAABBBBBBBBBXXXXXXXXYYYYYYYYDDDDDDDRRRRQWTTTTTTTTTTxxxx
xxxxyyyyyyyyTTTTTTTTTTxxxxxxxxyyyyyyyyTTTTTTTTTTxxxxxxxxyyyyy
yyy
```

Las "A" contienen el código del poste origen. Las "B" dan el poste destino. las "X" con las "Y" dan el punto origen de la línea. El largo de la línea, el ángulo, el tipo de línea (abierta o cerrada), el tipo de fases (1 fase negra, 3 fases, 2 fase roja-verde, etc) están contenidos en "D", "R", "Q", "W" respectivamente. Las "T" dan el texto que describen a una, dos o tres fases[9]. La información se guarda en las tablas LINEAS_PRIMARIAS y DETALLE_LINEASPRI.

d.1.2 Lpri_borra

Esta opción elimina un segmento de línea primaria. El "string" que se recibe tiene el siguiente formato :

CCCCCCCCCDDDDDDDD

Se elimina todo lo relacionado con el segmento, incluyendo la información de las fases.

d.1.3 Lpri_muestra

Esta opción muestra los segmentos de línea primaria, con toda su información, que están comprendidos en una ventana. El "string" que se recibe tiene el siguiente formato :

CCCCCCCCDDDDDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

Las "C" con las "D" dan las coordenadas UTM inferior izquierda y superior derecha. "X", "Y", "Z", y "W" dan las

[9]El texto de las fases contiene el número de fases, el calibre y el tipo de conductor

coordenadas de esos puntos.

d.2 Líneas secundarias

Las líneas secundarias son similares con las líneas primarias en cuanto al procesamiento de la información en este programa. La diferencia principal es que las líneas secundarias carecen del campo tipo_fases, ya que todas se dibujan igual, aparte de que residen en tablas distintas a las líneas primarias.

B. PRU_HLI

Este programa sirve para procesar "queries" dinámicos. Es posible ingresar o seleccionar información en la base de datos, aunque para fines de este proyecto se usará solamente para consulta. Se puede extraer la información que se desee, de la o las tablas que se especifiquen, con las condiciones que se requieran. El programa está concluido, pero no se hace una descripción de este porque no están terminadas todavía las rutinas en AUTOLISP que lo utilicen, es decir la construcción de los "queries" dinámicos.

V AMBIENTE DE LA ESTACION DE TRABAJO

En este momento no se cuenta con las estaciones de trabajo definitivas, ya que se está en la fase de su adquisición. Sin embargo si se tiene claro cuáles tienen que ser sus características[10], entre las que destacan la rapidez de procesamiento de información y una alta calidad en la presentación de gráficas. Para la implementación del proyecto se ha contado con una microcomputadora AT-286, compatible con IBM.

En cada estación de trabajo se tiene una copia de ACAD con varios archivos utilitarios y programas, entre ellos uno que sirve de comunicaciones entre la estación de trabajo y la computadora central. Esto viene a dar dos áreas de trabajo bien definidas :

- Comunicaciones
- Gráficas

Estas áreas se describen a continuación.

A. COMUNICACIONES

La primera actividad que se desarrolló en el proyecto de mapeo fue la elaboración de un programa para comunicar los dos ambientes de trabajo. El lenguaje escogido fue Pascal (Turbo Pascal 5.5) por la facilidad que da para hacer este tipo de

[10]Ver apéndice E

programas, aunque bien se pudo haber utilizado otro lenguaje como "C" o "Assembler". El objetivo final de este programa era establecer comunicación a 9600 baudios (bits por segundo) por medio de los puertos seriales RS-232C. Se comenzó con la elaboración de una pequeña parte de la emulación[11] de una terminal Data General. Realmente existen dos programas hechos en "Turbo Pascal"

- Manda
- Comunifi

los cuales se describen a continuación.

1. manda

Los objetivos de este programa son establecer comunicación entre la estación de trabajo y la computadora central e iniciar una sesión con esta última. Para iniciar una sesión, hay que contestar dos preguntas : el nombre de usuario y la clave secreta o "password". Lo que hace el programa es mandar desde la estación de trabajo las dos respuestas. Cada respuesta es un "string", terminado con el código ASCII 10 "Carriage Return". De esta manera se realiza el proceso normal que efectúa un usuario al entrar a una sesión. Una vez que se mandaron los dos "strings", se manda un tercero, el cual contiene lo siguiente :

[11]Ver apéndice D

"x queries".

Esto lo que hace es ejecutar el programa "queries" descrito en las páginas anteriores. Al cargarse este programa ya pueden realizarse los diferentes "queries" originados en ACAD.

2. comunifi

El objetivo de este programa es el de mandar a la computadora central, más específicamente al programa "queries", todos los requerimientos originados por los usuarios en la estación de trabajo. Asimismo, este programa se encarga de recibir todos los resultados y guardarlos en un archivo el cual lee posteriormente AUTOLISP.

La manera como ACAD (AUTOLISP) se comunica con el programa "queries" es con la escritura del requerimiento en un archivo, el cual se almacena en el disco virtual "E". La información se traslada de AUTOLISP a Pascal y viceversa por medio de este archivo. Posteriormente se ejecuta el programa "comunifi". Este, a su vez, cuando recibe los resultados los guarda en un archivo (en el disco virtual "E") y termina la ejecución del programa. Posteriormente, el programa elaborado en AUTOLISP recupera el control de la ejecución y lee los resultados.

B. GRAFICAS

ACAD es el paquete utilizado para el manejo de gráficas y planos. Es el único ambiente de trabajo visible para el usuario. Las diferentes opciones implementadas han sido elaboradas en AUTOLISP. AUTOLISP es el lenguaje de programación de ACAD, siendo capaz de ejecutar todos los comandos existentes en este paquete. Es esta integración entre un lenguaje y los diferentes comandos de manipulación de gráficas, lo que hace ideal a ACAD como un medio para desarrollar el proyecto de mapeo.

Por ejemplo en ACAD existe el comando "line", el cual dibuja una línea entre dos puntos especificados por el usuario. Con AUTOLISP es posible dar la siguiente instrucción:

```
(Command "line" pause pause "")
```

Donde "pause" significa que el usuario puede seleccionar cualquier punto, y las comillas finales significan un "ENTER" para indicar que la instrución terminó. En lugar que el usuario seleccione los puntos, estos pueden estar ya definidos. Si a la variable "punto1" se le asigna la siguiente lista : (2 2) y a la variable "punto2" la lista (4 4), la siguiente instrucción dibujará una línea de 45 grados entre ambos puntos (ver figura 18) :

```
(command "line" punto1 punto2 "")
```

Es posible dibujar muchas otras figuras o símbolos de

diferentes tamaños, o dibujar algo más complejo como el plano de una ciudad.

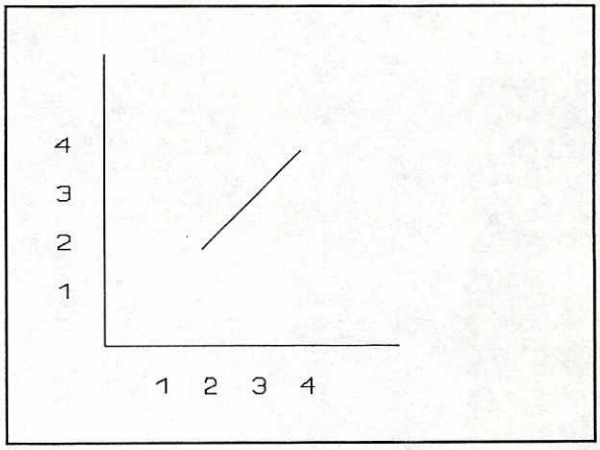


figura 18

ACAD provee un código único para cada entidad en un dibujo. Una entidad es cada símbolo, figura, texto o bloque (agrupación de una o más entidades) en un dibujo.

La línea del ejemplo anterior es una entidad, la cual tiene asociado dos puntos para indicar: el principio y fín, un tipo de línea (hay varios estilos), un color y otras características.

Para construir el símbolo de un poste normal, se dibujó primero un círculo con un radio determinado. Posteriormente se definió un atributo de texto para almacenar el código del poste, el cual es un "string". Teniendo estas dos entidades se definió un bloque llamado "normal" que las agrupa.

Mediante el manejo de las diferentes entidades, es posible la implementación del sistema de mapeo ya que se puede ingresar, seleccionar, mostrar o borrar aquellas entidades que interesen.

Hay que observar que los diferentes símbolos o bloques que

corresponden a elementos del sistema de distribución (postes, anclas, transformadores, líneas) no forman parte de los planos en si. Los elementos residen en la base de datos en la computadora central y sólo se muestran en los planos cuando el usuario lo solicite. Al terminar la sesión de mapeo, estas entidades no son guardadas en los planos. En caso que se ingrese un nuevo elemento, por decir un transformador, la información de éste quedará almacenada en la base de datos técnica.

Los diferentes elementos del sistema de mapeo se clasifican en "layers". Un "layer" es una clasificación de entidades. Todas las líneas, figuras o bloques que se dibujen bajo el "layer" activo pertenecerán a éste. Cada "layer" tiene un color y tipo de línea asignado, pero cualquier entidad puede pertenecer a cualquier "layer" [1]. Los postes se dibujan bajo el "layer" postes, las anclas bajo el "layer" anclas, y así los diferentes elementos en su "layer" correspondiente. A continuación se hace una descripción de las rutinas en AUTOLISP necesarias para la implementación del sistema de mapeo.

Selección de los mapas

Se tiene un número aproximado de 500 planos de 2 x 2 kilómetros, lo cual da un tamaño del orden de los 100 Megabytes. Es un problema tener todos los planos cargados al mismo tiempo, por limitaciones en memoria y degradación del

sistema. A causa de esto, la ventana máxima que se puede hacer es de 64 kilómetros cuadrados. Para escoger la ventana, es necesario seleccionar dos puntos que definan un área rectangular. Cada vez que se escoge un punto (mediante la tecla "ENTER" o el botón "pick" del mouse), éste queda registrado en una variable de ACAD llamada "LASTPOINT". Cada vez que hay algún desplazamiento del cursor, se dibuja un área rectangular comprendida entre el primer punto escogido para la ventana y la posición actual del cursor. Cada vez que hay un desplazamiento, se muestra en la parte superior izquierda de la pantalla el código de coordenada UTM correspondiente a ese punto. De esta manera, el usuario puede darse una idea del área que seleccionó.

Cuando se escoge el segundo punto, la ventana queda formada y se hace un "zoom" o acercamiento de esa región, Posteriormente se cargan todos los planos que están comprendidos en ella. Esta opción se llama "ventanag" y es la primera opción del primer menú. Las otras dos opciones de este menú, ventana+ y ventana-, sirven para hacer acercamientos y distanciamientos, pero sin cargar planos.

Una vez escogida el área de interés se presenta el menú que contiene los diferentes elementos (figura 3). Estas opciones se describen a continuación.

a. POSTES

En el menú de postes (figura 5) se presentan las siguientes opciones : marcar, select, desmarcar y muestra.

a.1 Marcar

Cuando se marca un poste se toma el punto de inserción del bloque de poste correspondiente(hay un bloque diferente por cada tipo de poste). De este punto se obtienen las coordenadas UTM. Posteriormente se pregunta el número del poste[12]. Este número se puede cambiar de lugar en el plano, por lo que también se toma el punto de inserción del número de poste. Ya que se conoce el tipo de poste, se forma el siguiente "string":

P0CCCCCCCCCCXXXXXXXXXYYYYYYYYYZZZZZZZZZWWWWWWWWWT

Las letras tienen el mismo significado que se explicó en la opción "Pmarcar" del programa "queries". El primer carácter indica que se va a trabajar con postes. El "0" indica que se va a grabar un poste (en la base de datos).

Este "string" se guarda en un archivo. A continuación, se ejecuta la instrucción :

(command "shell" "comunifi")

la cual ejecuta el programa de comunicaciones para que lo transmita a la computadora central. Una vez terminado el programa "comunifi", se lee la respuesta que mandó la

[12]El código del poste se forma asi : UTM + número de poste

computadora central (en el mismo archivo que se mandó el "string"). Si la respuesta que se recibió es una lista nula (nula = "()"), la transacción no se realizó correctamente, por lo que el poste se borra del plano.

El proceso de guardar el requerimiento en un archivo, llamar el programa de comunicaciones y leer del archivo la respuesta que manda la computadora central se repite para todas las opciones.

a.2 Desmarca

Se escoge un poste y se forma el siguiente "string":

P4CCCCCCCCC

donde las "C" dan el código del poste. Esta opción sirve para poner en cero los campos que aportan la ubicación de este poste.

a.3 Muestra

Se toman los dos puntos de la ventana que se hizo para generar la pantalla actual. El "string" que se manda a procesar es el siguiente:

P6CCCCCCCCCDDDDDDDDDXXXXXXXXXYYYYYYYYYZZZZZZZZZWWWWWWWWW

La lista que se recibe tiene el siguiente formato:

((numero_poste coordenada_poste_x coordenada_poste_y coordenada_texto_x coordenada_texto_y tipo_Poste) (...) (...) ...

)

El número de sublistas que regresa el programa "queries" depende del tamaño de la ventana y de la densidad de postes en ella. Cada sublista trae el número de poste, el tipo, las coordenadas de ubicación del poste y de su código. Cada sublista involucra un "insert" del bloque que define el tipo de poste.

a.4 Select

Se selecciona un poste. Se muestra la información del mismo almacenada en la base de datos, en una ventana de ACAD cuyo objetivo es presentar la información relacionada con un bloque.

b. ANCLAS

En el menú de anclas (figura 4) se presentan las siguientes opciones : Marcar, desmarcar, muestra.

b.1 Marcar

Primero se escoge un poste. Posteriormente se presenta un menú con ocho tipos de anclas por escoger. Al escoger un ancla, ésta se puede rotar en cualquier ángulo. Una vez seleccionado el ángulo de rotación se construye el siguiente "string" para que se actualice la base de datos :

A0CCCCCCCCCRRRT

Las letras ("C", "R" y "T") tienen el mismo significado que en la opción del mismo nombre en el programa "queries". La "A"

con el "0" indican que se va a ingresar un ancla en la base. Si la lista que se recibe del computador es nula, el ancla se borra inmediatamente del plano.

b.2 Desmarcar

Se manda el siguiente "string" :

A4CCCCCCCCCRRR

con el objetivo de eliminar el ancla con un ángulo de rotación igual a "RRR" en el poste "CCCCCCCCC" de la tabla anclas.

b.3 Muestra

Se muestran todas las anclas comprendidas en una ventana. El "string" que se manda es similar al de la opción muestra de "postes". La lista que se recibe tiene el siguiente formato :

((coordenada_x coordenada_y angulo tipo) (...) ...)

Cada sublista trae las coordenadas de inserción del bloque definido por el tipo de ancla. También se recibe el ángulo de rotación del ancla.

c. BANCOS DE TRANSFORMADORES

Con los bancos de transformadores se puede hacer lo siguiente : marcar un banco, eliminar un banco, mostrar los bancos de transformadores comprendidos en una ventana, sumar la carga de los transformadores comprendidos en una ventana. Estas opciones se explican a continuación.

c.1 Marcar

Con esta opción se puede definir un banco en un poste. Un banco de transformadores puede tener un máximo de tres transformadores (no tienen que ser necesariamente del mismo tipo). Los transformadores se pueden rotar al igual que las anclas.

Una vez marcados los transformadores, se pregunta el número de banco, los números de compañía y las capacidades de los transformadores. Solamente el número de compañía es el que interesa, ya que los números de compañía con sus capacidades se pueden averiguar en la base de datos técnica. Estos se preguntan para visualizar la posición del bloque de información en el plano y evitar el traslape con otros elementos. El "string" que se manda a la computadora central está explicado en detalle en el programa "queries".

c.2 Desmarcar

Con esta opción se desmarca un poste y se elimina toda relación con el banco de transformadores que contiene. Lo único que se elimina es la relación con el banco y las posiciones de los transformadores. La información técnica del banco y de los transformadores no se altera.

c.3 Muestra

Muestra todos los bancos comprendidos en una ventana

c.4 Capacidad

Suma las capacidades de todos los transformadores comprendidos en una ventana.

d. LINEAS PRIMARIAS

d.1 Marcar

Se marca un segmento de línea entre dos postes. Primero se escogen dos postes y se traza una linea dirigida al centro de ambos bloques. Esta línea nunca llega al centro, ya que se corta en el punto de intersección de ésta con el símbolo del poste. Posteriormente se preguntan los tipos de conductores para las fases. Puede haber un máximo de tres fases en un segmento de línea. Cada fase se identifica de acuerdo a su posición por el nombre de un color (roja,verde,negra). Existen ocho combinaciones de fases que son necesarias identificar. Para cada combinación de éstas se asignó un color diferente, aunque todas las fases pertenecen al mismo "layer" (primario) . Se pregunta el texto de las fases el cual se puede cambiar de lugar. Una línea primaria puede ser abierta o normal. De acuerdo a su tipo, se dibuja un estilo de línea diferente.

d.2 Borrar

Sólo con marcar una línea, ésta se elimina de la tabla de líneas primarias.

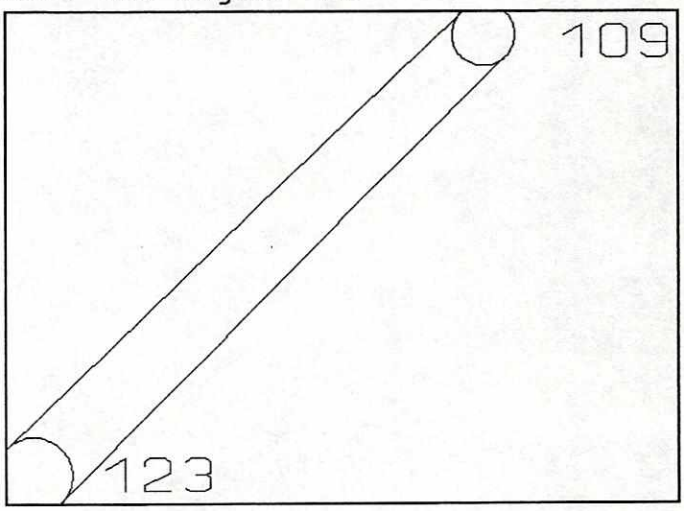
d.3 Muestra

Se muestran todos los segmentos de línea primaria comprendidos en una ventana.

e. LINEAS SECUNDARIAS

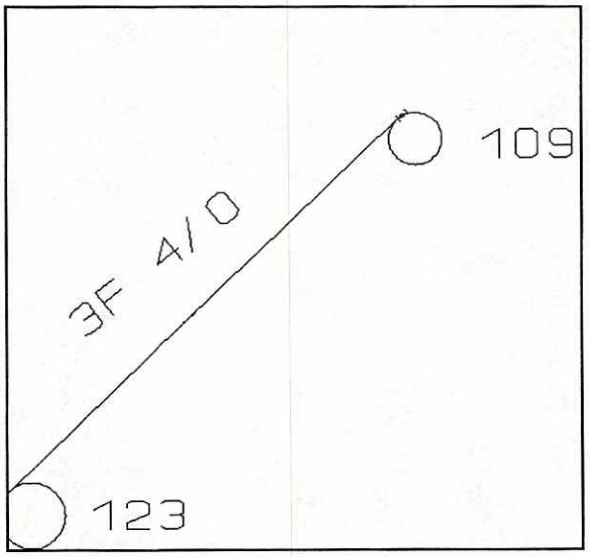
e.1 Marcar

Para marcar una línea secundaria, se da al usuario la alternativa de marcar la línea en la parte superior de los postes o en la parte inferior. Ver figura 19.



Si se seleccionaron los postes "109" y "123" para trazar entre ellos un segmento de línea secundaria, el programa en AUTOLISP traza dos líneas tangentes para que el usuario seleccione una de ellas. Una vez seleccionada la línea, se pregunta el texto de las fases, el cual se puede cambiar de lugar y rotarse. Ver figura 20.

Cuando los postes tienen una forma circular no hay ningún problema para trazar las tangentes. En el sistema de mapeo se presentan postes que tienen forma cuadrada o triangular. Lo que



se hizo para resolver este problema fué trazar círculos concéntricos a estos símbolos.

e.2 Desmarcar

Se selecciona un segmento de línea y éste se elimina de la tabla de líneas secundarias en la base de datos.

e.3 Muestra

Se muestran los segmentos de línea secundaria comprendidos en una ventana.

C. RUTINAS EN AUTOLISP

Las rutinas mas utilizadas, de las elaboradas en AUTOLISP, son obtiene y ventana.

La rutina "ventana" sirve para hacer un acercamiento o "zoom" en la pantalla. Primero se escoge un punto, el cual es el origen de la ventana. Esto se hace así :

- se lee la lista, la cual se genera cada vez que hay un cambio en la posicion del cursor o un "ENTER".
- Se asigna el primer elemento de la lista a la variable "fuente".
- El segundo elemento se asigna a la variable punto2.

"fuente" tiene el código del tipo de "input" [2], "punto2" contiene un entero o una lista. Dependiendo del valor de fuente y de punto2 se puede saber si se seleccionó algún punto. La lista contenida en "punto2" (si acaso es una lista) se asigna a la variable "pos". La ventana queda hecha cuando se selecciona un nuevo punto, asignándose éste a la variable "ultimo" para hacer el siguiente "zoom" :

```
(command "zoom" "w" pos ultimo)
```

En el caso de "ventanag", primero se llama a la rutina "ventana", para luego utilizando los valores de "pos" y "ultimo", cargar los planos comprendidos en la ventana escogida. Los planos llevan como nombre la coordenada UTM donde se encuentra el punto inferior izquierdo del plano, ejemplo :

B1248, G0340. Para hacer la conversión de punto a coordenada UTM se elaboró la rutina "obtiene". Esta rutina convierte el punto contenido en "LASTPOINT" a un formato UTM, ejemplo : Se tiene el siguiente punto : (123678.75 36177.89) en la variable "LASTPOINT". Al invocar la rutina "obtiene", ésta da como resultado en la variable "UTM" el siguiente valor : C3623B.

A continuación se describen el resto de rutinas hechas para implementar las diferentes opciones. Solamente se describen aquellas partes consideradas como más importantes[13].

rutina pmarcar

- Se ingresa un poste en una posición cualquiera. El tipo del poste está contenido en la variable "p_tipo".

- Se llama a la rutina "obtiene". Esta rutina da los primeros 6 caracteres del código UTM del poste. Lo otros tres están dados por el número del poste.

- Se obtienen las coordenadas que dan el punto de inserción del poste. Este punto está contenido en la variable de ACAD "lastpoint". Las coordenadas "X" y "Y" están dadas respectivamente por las variables "xmrealp" y "ymrealp".

- Se edita el atributo del poste que contiene su número y se cambia de lugar.

---

[13]El código se muestra en el apéndice F.

- Se obtienen las coordenadas del punto de inserción del atributo del poste. Los puntos "X" y "Y" se asignan a las variables "xmrealt" y "ymrealt" respectivamente.

- Se obtiene el número de poste, el cual está contenido en el atributo mencionado anteriormente. Este número se asigna a la variable "p_numero".

- Con la información obtenida anteriormente, se construye el siguiente "string":

```
manda = CONCATENE "P0" utm p_numero xmrealp "-" ymrealp
        "-"  xmrealt "-" ymrealt tipo_poste
```

el cual se escribe en el archivo "e:\archivo" y posteriormente se llama al programa "comunifi".

rutina pdesmarca

- Se selecciona un poste

- Se llama a la rutina obtiene

- Se extrae de este poste su número.

- Se construye el siguiente "string":

```
manda = CONCATENA "P2" utm numero.
```

Se escribe manda en el archivo "e:\archivo"

- Se escribe este "string" a "e:\archivo" y se llama al programa "comunifi".

rutina pselect

- Se seleciona un poste.

- Se llama a la rutina obtiene.

- Se extrae del poste su número.

- Se construye el siguiente "string":

manda = CONCATENA "P4" utm numero.

Se escribe manda en el archivo "e:\archivo"

- Se escribe este "string" a "e:\archivo" y se llama al programa "comunifi".

Se obtiene el registro del poste selecionado. Esta información se guarda en un bloque con el objetivo de ser editado con la instrucción "attdia" para hacer la presentación en una ventana.

rutina pmuestra

"pos" y "ultimo" son listas, las cuales contienen los puntos inferior izquierdo y superior derecho de la ventana. Estos se asignan a las variables xp, yp, xt. yt. Las coordenada UTM que generan estos puntos se asignan a poste1 y poste2, respectivamente. Estos valores se concatenan y se mandan a la computadora central.

manda = CONCATENA "P6" poste1 "A000" poste2 "D999" xp "-" yp "-" xt "-" yt

Se escribe manda en el archivo "e:\archivo"

- Se llama al programa "comunifi".

- Se lee el archivo "e:\archivo", el cual tiene la respuesta, la que se guarda en la lista "lista".
Cada sublista trae la información de un poste, con la información necesaria para su ubicación. El siguiente "While" ilustra esta operación.

- While lista <> NULA DO BEGIN
    - lista1 = primer elemento de lista.
    - lista = lista - primer elemento de lista.
    - En base a la información contenida en lista1, es posible dibujar los postes con su respectivo número o código.
    - El código del poste, que es un atributo del poste, se cambia de lugar hacia el punto contenido en lista1.
    - end

rutina amarcar

- Se selecciona un poste.

- Se llama a la rutine obtiene. Se obtienen los primeros seis caracteres de la coordenada utm del poste.
- Se tiene como centro de rotación el centro del poste. Es

posible colocar el tipo de ancla seleccionada en cualquier ángulo.

- Se arma al siguiente "string" :

    manda = CONCATENA "A0" utm p_numero angulo tipo_ancla

- Se llama al programa "comunifi".

rutina adesmarca

- Se selecciona un ancla. Un atributo de texto del ancla es el número de poste.

- Del ancla se extrae información acerca del poste al cual se relaciona. Se extrae también el ángulo de rotación.

- Se arma el siguiente "string" :

    manda = CONCATENA "A4" poste angulo.

- Se llama al programa "comunifi".

rutina amuestra

Se manda un string idéntico al que se generó en la rutina pmuestra, con la única diferencia que los dos primeros caracteres sirven para indicar que se trata de obtener las anclas de una ventana.

- Se llama al programa "comunifi".

- Los resultados se asignan a "lista". Cada sublista contiene

información acerca del tipo y posición de cada ancla. Estas sublistas se procesan de una manera similar al "WHILE" descrito en la rutina p_muestra.

rutina tmuestra

Esta rutina se ocupa para extraer los bancos de transformadores de una ventana junto con la información técnica. Se contruyen dos "strings" : manda y manda2, los cuales se ocupan para hacer distintos "queries". El primer "query" se utiliza para extraer de la base de datos, los transformadores que están en cada poste (se extraen las coordenadas, tipo y ángulo de rotación de cada transformador).

- Se contruyen los siguientes "strings":

```
manda = CONCATENA "T6" poste1 "A000" poste2 "D999"
                  ix "-" iy "-" dx "-" dy
manda2 = CONCATENA "T5" poste1 "A000" poste2 "D999"
                  ix "-" iy "-" dx "-" dy
```

"ix", "iy", "dx", "dy" definen respectivamente los puntos inferior izquierdo y superior derecho de la ventana.

- El "string" manda se guarda en "E:\archivo". Se llama al programa "comunifi".

La respuesta se guarda en "lista", Esta lista se recorre para dibujar los transformadores en sus respectivas posiciones.

- WHILE lista <> NULA DO BEGIN

- lista1 = primer elemento de lista1.
- lista = lista - primer elemento de lista
- "INSERT" del transformador en base con la información de "lista1".

END.

- El "string" manda2 se guarda en "E:\archivo". Se llama al programa "comunifi".

La respuesta se guarda en "lista". Esta lista se recorre para dibujar la información de los bancos de transformadores en sus respectivas posiciones. El segundo "query" trae como resultado información de cada banco (número de banco, números de compañía, capacidades en KVA).

- Se hace otro "WHILE" similar al explicado anteriormente

rutina trafo_bloque

Esta rutina se ocupa para mandar a la computadora central los transformadores ingresados en la rutina "trafo_poste"

- Se extrae la información de los atributos del bloque "trafo_bloque".

- Se arma el siguiente "string":

manda = CONCATENA "T0" p_numero banco coordexx coordeyy angulo1 tipo1 angulo2 tipo2 angulo3 tipo3

- Se manda el número de banco junto con los transformadores a la computadora central por medio del programa "comunifi".

rutina t_capacidad

- Se manda un "string" similar al que se mandó en tmuestra

- El resultado es la suma de las capacidades de los transformadores en la ventana seleccionada.

rutina lineas_primar

Esta rutina se ocupa para ingresar un segmento de línea primaria entre dos postes. Se pregunta el texto de las fases, el cual se puede trasladar y rotar.

- Se selecciona primero el poste 1.

- Se selecciona posteriormente el poste 2.

- Se hace un círculo concéntrico en ambos postes con un radio igual o menor al poste que los contiene. El centro de cada círculo es el mismo que el del símbolo donde se encuentran. Esto se hace para trazar una línea que va del centro de un poste al centro del otro, para después aplicar en ambos extremos un "trim" de la línea contra el círculo dibujado. Esto se hace para que la línea toque solamente los límites de cada poste. La razón de dibujar el círculo es que el comando "trim" no puede usarse con bloques.

- Se pregunta la información de las fases, la cual se guarda en el bloque "bloque_lineas".

- El texto de las fases puede cambiarse de lugar y/o rotarse.

- Una línea primaria puede ser normal o abierta. Cuando es abierta se dibuja el siguiente símbolo al final del segmento : "¦". Este símbolo es considerado como un atributo del bloque "lineas". A causa de que la escala en "x" de este bloque se ajusta a la distancia entre los dos postes, este atributo crece en igual proporción lo cual no es deseable. Para hacer esto se cambia la entidad que identifica el atributo, la sublista identificada por el numero 41, la cual contiene la escala en "x". Esta escala se pone en 1.

- Las atributos de "lineas", que contienen información acerca del texto de las fases, se substituyen por los atributos del bloque "bloque_lineas".

- Después de haber cambiado las listas de información de las entidades, éstas se actualizan.

- Se transmite la información a la computadora central. Se utiliza el programa "comunifi" y se almacena la información en el archivo "e:\archivo".

rutina lineas_secund

- Se marca primero el poste 1.

- Se marca el poste 2.

- angulo_radianes es el ángulo entre el punto de inserción del poste1 con el punto de inserción del punto2.

- Se tiran dos líneas tangentes entre los postes, de las cúales se selecciona una de ellas.

- Una vez escogida la línea, la forma de ingresar el texto de las fases y su movimiento es similar al proceso que se hace en la rutina "lineas_primarias". Los tipos de líneas se trabajan igual que en "lineas_primarias".

VI. CONCLUSIONES Y RECOMENDACIONES

La principal razón, para desarrollar el sistema de mapeo en lugar de comprar uno, fue el elevado precio de éstos.

En este momento se cuenta con las herramientas básicas para el ingreso de los planos, elementos del sistema eléctrico, y la información técnica de estos últimos. La siguiente fase en el desarrollo del proyecto es la implantación de utilitarios que analicen la red eléctrica. Para tener tiempos de respuesta aceptables cuando se utilicen estos utilitarios, será necesario contar con estaciones de trabajo de alta velocidad, ya que las computadoras personales existentes en el mercado no se ajustan a los requerimientos de velocidad de procesamiento requeridos en estos estudios.

Muchas personas pueden argumentar que todo este trabajo es innecesario, ya que en el mercado existe una buena cantidad de productos que cumplen a cabalidad con las necesidades de información geográfica de una institución como la Empresa Eléctrica. Después de haber hecho algunos estudios se concluyó que el principal problema para una correcta implantación del sistema no era el "software" que iba a manejar la información, sino la base geográfica con la cual se iban a elaborar los planos. En este momento no existe en el país ninguna empresa que proporcione planos actualizados del área de servicio.

Un sistema de mapeo debe verse en forma integrada : Los planos y las metodologías de elaboración y actualización de éstos, junto con el "software" que los manipula. Cualquier falla, tanto en el "software" como en la elaboración de los planos, llevará el sistema al fracaso.

Este sistema no debe de verse como algo estático que será utilizado por mucho tiempo sin cambios de ningún tipo. Se debe utilizar toda la tecnología que esté al alcance para mejorarlo.

VII. BIBLIOGRAFIA

1. AutoCAd Reference Manual. Autodesk, Inc., 1989. 455 pp.

2. AutoLisp Programmer's Reference. AutoDesk, Inc., 1989. 210 pp.

3. Designing and Mantaining a SQL Database. Data General Corporation, 1988.

4. Jourdain, R. Programmer's Problem Solver. Adison-Wesley, Readin, Mass., 1986. 430 pp.

5. Ozkarahan, E. Database Machines and Database Management. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1986. 636 pp.

6. Grogono, P. Programación en PASCAL. Addison-Wesley Iberoamericana, Wilmingtong, Delaware, 1986. 371 pp.

7. Programming for DG/SQL Database Access. Data General Corporation, 1988.

8. Schatt. S. A fondo : Redes de área local. Ediciones Anaya Multimedia, S.A, Madrid., 1989. 294 pp.

9 . The "C" language reference and runtime Manual. Data General Corporation, 1988.

10. Wiedelhol, G. Diseño de Base de Datos. MC-Graw-HIll, México, 1985. 921 pp.

11. Winston P. y P. Berthold Klaus. LISP. Adison-Wesley, Reading, Mass., 1984. 434 pp.

APENDICE A

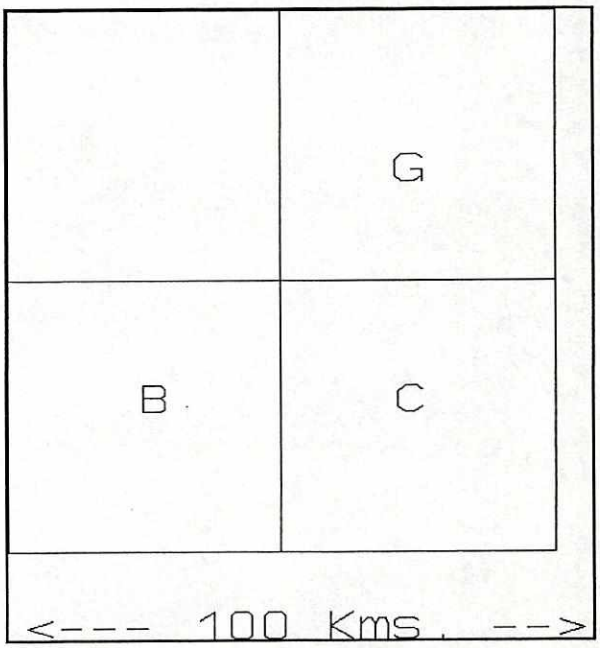
CODIFICACION DE LOS POSTES

Para la codificación de los postes se escogieron las coordenadas UTM y un número de poste. Estas coordenadas UTM solamente dan una idea aproximada de la ubicación del poste en un área de 500 x 500 metros. Un código de poste tiene el siguiente formato :

AyyxxBddd

donde "A" da un cuadrante de 100 x 100 Kilómetros (fig. 21).

Figura 21

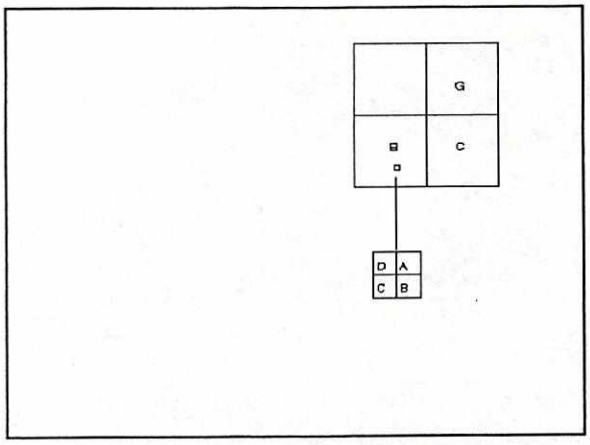


La empresa solamente necesita de 3 de estos cuadrantes para cubrir su área de servicio. "yy" da el desplazamiento en el eje "Y" a partir del origen del cuadrante "A", al igual que "xx" da el desplazamiento en "X". Estos desplazamientos son medidos en números enteros (rango 00-99) y cada unidad representa un kilómetro. "B" da un subcuadrante de 500 x 500 mts (ver figura 22). Dentro del área definida por

"Ayyxx", se tienen cuatro subcuadrantes. "ddd" da el número del poste dentro de esta última área.

De esta manera se tiene un código de nueve dígitos alfanuméricos que identifican de manera única a un poste, pero no se tiene una ubicación exacta del poste (ejemplo de código de poste : B1234C123). A causa de esto en la tabla de postes se agregaron dos campos, para indicar la ubicación exacta del poste.

Figura 22



APENDICE B

DESCRIPCION DE LA BASE DE DATOS

A continuación se presenta una descripción de las tablas utilizadas de la base de datos de mapeo para este proyecto.

TABLA postes

```
p_numero      : CHAR(9)          { código del poste }
p_direccio    : CHAR(50)         { dirección del poste }
p_circuito    : SMALLINT         { circuito al que esta
                                   conectado }
p_numebanc    : CHAR(4)          { banco al que se
                                   conecta }
p_tipo        : CHAR(1)          { tipo de poste }
p_material    : CHAR(1)          { tipo de material }
p_altura      : SMALLINT         { altura del poste en
                                   pies }
p_construc    : CHAR(1)          { tipo de construcción }
p_coordexx    : DECIMAL(7,2)     { punto de inserción en
                                   el plano
p_coordeyy    : DECIMAL(7,2)       de ACAD del poste }
p_proveedo    : CHAR(1)          { código del proveedor }
p_anofabri    : SMALLINT         { año de fabricación }
p_precinst    : DECIMAL(7,2)     { costo al instalarse }
p_estadofi    : SMALLINT         { código de estado
                                   físico }
p_fechamap    : INTEGER          { fecha que se levantó
                                   la información}
p_valormap    : DECIMAL(7,2)     { valor al levantarse la
                                   tarjeta
                                                       }
p_valoact     : DECIMAL(7,2)     { valor actual }
p_textox      : DECIMAL(7,1)     { ubicación en el plano
                                   de ACAD
p_textoy      : DECIMAL(7,1)       del texto del código
                                   del poste }
```

Llave : (p_numero)

TABLA anclas

```
a_poste      : CHAR(9)          { Poste donde se
                                  encuentra }
a_angulo     : SMALLINT         { ángulo de rotación
                                  del ancla }
a_tipo       : CHAR(1)          { tipo de ancla }
```

Llave : (a_poste,a_angulo)

TABLA bancos

```
b_circuito   : SMALLINT         { circuito al que está
                                  conectado }
b_numero     : CHAR(4)          { número de banco }
b_conexion   : CHAR(2)          { tipo de conexión }
b_voltajep   : CHAR(10)         { voltajes
b_volatejes  : CHAR(7)                        }
b_cargatip   : CHAR(1)          { carga típica }
b_poste      : CHAR(9)          { poste donde se
                                  encuentra}
b_coordexx   : DECIMAL(7,1)     { punto donde se inserta
                                  el bloque
b_coordeyy   : DECIMAL(7,1)       con información del
                                  banco en el plano de
                                  ACAD }
```

Llave : (b_circuito,b_numero)

TABLA trafos_posicion

```
tp_circuito  : SMALLINT         { banco al que hace
                                  referencia
tp_banco     : CHAR(4)                  }
tp_angulo    : SMALLINT         { ángulo de rotación }
tp_tipo      : CHAR(1)          { tipo de transformador
                                }
```

Llave : (tp_circuito,tp_banco,tp_angulo}

```
TABLA transformadores

   t_compania       : INTEGER        { número de compañía }
   t_numserie       : CHAR(10)       { número de serie    }
   t_fabrican       : CHAR(2)        { código del fabricante
                                       }
   t_capacida       : SMALLINT       { capacidad del
                                       transformador }
   t_estaus         : CHAR(1)        { estatus del
                                       transformador }
   t_voltajep       : CHAR(12)       { voltajes
   t_voltajes       : CHAR(7)                          }
   t_fasecone       : CHAR(1)        { fase a la que se
                                       conecta }
   t_derivaci       : CHAR(1)        { derivaciones }
   t_tipo           : CHAR(1)        { tipo del transformador
                                                       }
   t_polarida       : CHAR(1)        { polaridad }
   t_circuito       : SMALLINT       { número de circuito }
   t_banco          : CHAR(6)        { código del banco al
                                       que se
                                       conecta }

   Llave : (t_compania)

TABLA lineas_primarias

   lp_posteor          : CHAR(9)     { poste origen del
                                       segmento }
   lp_postede          : CHAR(9)     { poste destino }
   lp_desplazamiento   : DECIMAL(6,2)
   lp_angulo           : DECIMAL(4,1)
   lp_tipo_linea       : CHAR
   lp_tipo_fases       : CHAR


   Llave : (lp_posteor,lp_postede)


TABLA detalle_lineaspri

   dp_posteor          : CHAR(9)
   dp_postede          : CHAR(9)
   dp_texto            : CHAR(10)
   dp_coordenadax      : DECIMAL(7,1)
```

```
   dp_coordenaday    : DECIMAL(7,1)

   LLave : (dp_posteor,dp_postede,dp_texto)

TABLA lineas_secundarias

   ls_posteor          : CHAR(9)        { poste origen del
                                          segmento }
   ls_postede          : CHAR(9)        { poste destino }
   ls_desplazamiento : DECIMAL(6,2)
   ls_angulo           : DECIMAL(4,1)
   ls_tipo_linea       : CHAR
   ls_tipo_fases       : CHAR

   Llave : (ls_posteor,ls_postede)

TABLA detalle_lineassec

   ds_posteor          : CHAR(9)
   ds_postede          : CHAR(9)
   ds_texto            : CHAR(10)
   ds_coordenadax      : DECIMAL(7,1)
   ds_coordenaday      : DECIMAL(7,1)

   LLave : (ds_posteor,ds_postede,ds_texto)
```

Existen más tablas que las aquí mencionadas pero para fines de este proyecto se listaron las más importantes. Tampoco se describen las referencias entre las tablas utilizadas.

APENDICE C

QUERIES UTILIZADOS

A continuación se hace una descripcion de los "queries" utilizados en las diferentes rutinas pertenecientes a las opciones descritas en el programa "queries". El código que se muestra entre el símbolo de dólares($) y el punto y coma(;) corresponde a SQL.

pmarca

Esta es la rutina llamada para ingresar un nuevo poste a la tabla "postes". Lo primero que hace es el siguiente "query":

```
$ SELECT p_coordexx,p_coordeyy INTO
  :p_coordexx:indice,:p_coordeyy FROM postes WHERE
  p_numero = :p_numero;
```

Se seleccionan los campos de ubicación del poste para ver si éste ya tiene una ubicación en el plano. ":p_numero" contiene el código del poste. Si el poste ya existe en la tabla (SQLCODE = 0) y los campos de ubicación son iguales a cero, la tabla de POSTES es actualizada para el registro identificado por "p_numero". Si el cógigo del poste no existe, éste se graba en la tabla. Caso contrario se omite cualquier acción.

```
if p_coordexx = 0 and  p_coordeyy = 0 and SQLCODE = 0
```

```
 then begin
   $  UPDATE postes
      SET p_coordexx = :px,
          p_coordeyy = :py,
          p_textox = :p_textox,
          p_textoy = :p_textoy,
          p_tipo = :p_tipo
   WHERE p_numero = :p_numero;
  end
 else
if SQLCODE <> 0 then begin
  $ INSERT INTO postes
        SET p_numero = :p_numero,
            p_coordexx = :px,
            p_circuito = 0,
            p_tipo = :p_tipo,
            p_material = " ",
            p_coordeyy = :py,
            p_direccio = " ",
            p_numebanc = 0,
            p_ctrlbanc = " ",
            p_altura = 0,
            p_construc = " ",
```

```
            p_proveedo = " ",
            p_anofabri = 0,
            p_precinst = 0,
            p_estadofi = 0,
            p_fechamap = 0,
            p_valormap = 0,
            p_textox = :p_textox,
            p_textoy = :p_textoy,
            p_valoract = 0;
        end
```

pdesmarca

```
$ SELECT p_numero INTO :p_numero:inpdes FROM postes
 WHERE p_numero = :p_numero;
```

Si el poste existe (SQLCODE igual a cero), se actualizan las coordenadas de ubicación del poste(se ponen en cero)

```
 if SQLCODE = 0 then begin
      $ UPDATE postes
         s p_coordexx = 0,
             p_coordeyy = 0,
             p_textox = 0,
             p_textoy = 0
```

```
WHERE p_numero = :p_numero;
end
```

selecpob

```
$ DECLARE curpob CURSOR FOR SELECT
p_coordexx,p_coordeyy,p_numero,p_tipo,p_textox,p_textoy
FROM postes WHERE p_numero <= :pder and p_numero >= :pizq and
p_coordexx  >= :lizqx  and p_coordeyy >= :lizqy and
p_coordexx <= :lderx and p_coordeyy <= :ldery;
```

Lo que se muestra aquí es la declaración del "cursor" para poder seleccionar los postes comprendidos en una ventana. Las últimas cuatro comparaciones son necesarias para limitar el espacio de la ventana ya que si solamente se utilizan las dos primeras se puede obtener más información que la realmente solicitada.

pselect

A continuación se muestra el "select" para obtener los campos de la tabla "postes":

```
$ SELECT p_direccio,p_circuito,p_numebanc,p_ctrlbanc,
p_tipo,p_material,p_altura,p_construc,p_coordexx,p_coordeyy,
p_proveedo,p_anofabri,p_precinst,p_estadofi,p_fechamap,
```

```
p_valormap,p_valoract
INTO
:p_direccio:indisel,:p_circuito,:p_numebanc,:p_ctrlbanc,
:p_tipo,:p_material,:p_altura,:p_construc,:p_coordexx,
:p_coordeyy,:p_proveedo,:p_anofabri,:p_precinst,:p_estadofi,
:p_fechamap,:p_valormap,:p_valoract FROM postes
WHERE p_numero = :p_numero;

amarca

$ SELECT a_poste INTO :a_poste FROM anclas in da WHERE
a_poste  = :a_poste and a_angulo = :a_angulo;
```

Si ya existe un ancla en el poste con el mismo ángulo de rotación, la transacción no puede realizarse. En caso contrario (SQLCODE no igual a cero) se ingresa un nuevo registro a la tabla ANCLAS.

```
if  SQLCODE <> 0 then begin
     $ INSERT INTO anclas
       SET a_poste = :a_poste,
           a_tipo = :a_tipo,
           a_angulo = :a_angulo;
   end
```

adesmarca

Se elimina una ancla para este poste (a_poste), escogida por el usuario en ACAD.

```
$ DELETE FROM anclas in da WHERE a_poste  = :a_poste and
a_angulo = :a_angulo;
```

ancla_select

Esta es la rutina que sirve para seleccionar todos las anclas comprendidas en una ventana. El "cursor" se hace sobre la tabla postes para poder hacer la selección de los postes y sus anclas según la ventana definida por el usuario, ya que solamente en la tabla "postes" existe la información referente a una ubicación geográfica.

```
$ DECLARE curanc CURSOR FOR SELECT
a_poste,a_angulo,a_tipo,p_coordexx,p_coordeyy,p_tipo
FROM postes,anclas
WHERE
  a_poste <= :pder and a_poste >= :pizq and
  p_numero = a_poste and p_coordexx >= :lizqx and
  p_coordeyy >= :lizqy and p_coordexx <= :lderx and
  p_coordeyy <= :ldery;
```

tmarca

```
$ SELECT b_coordexx,b_coordeyy INTO :b_coordexx,:b_coordeyy
FROM bancos in da WHERE b_numero = :b_numero and b_circuito =
:b_circuito;
```

Primero se pregunta si para este número de banco y circuito ya existe una ubicación geográfica del texto del banco. Esta ubicación (campos b_coordexx, b_coordeyy) se utiliza para ubicar el punto de inserción del bloque de información del banco en el plano de ACAD, el cual contiene el número de banco, los números de compañía con sus capacidades.

```
if b_coordexx = 0 and  b_coordeyy = 0 and SQLCODE = 0
   then begin
      $ UPDATE bancos
         SET b_coordexx = :bx,
             b_coordeyy = :by,
             b_poste = :b_poste
      WHERE b_numero = :b_numero and
            b_circuito = :b_circuito;
```

En el "string" que se recibe pueden venir de uno a tres transformadores, cada uno con su tipo y ángulo de rotación. Por cada transformador se hace un "**INSERT**" en la tabla

TRAFOS_POSICION, la cual se relaciona con la tabla BANCOS mediante el número de banco y de circuito.

```
while (existan transformadores) do
 $ INSERT INTO trafos_posicion
       SET tp_circuito = :b_circuito,
           tp_banco = :b_numero,
           tp_angulo = :tp_angulo,
           tp_tipo = :tp_tipo;
  end  /* fin del while */
 end /* fin del then */
 else
   if (SQLCODE != 0)  then begin
```

Si el banco no existe todavía, éste se ingresa a la tabla BANCOS. Solamente se graban los campos referentes al número de banco, número de circuito y los campos de ubicación.

```
         $ INSERT INTO bancos
            SET b_circuito =:b_circuito,
                b_numero = :b_numero,
                b_conexion = " ",
                b_voltajep = " ",
                b_voltajes = " ",
                b_cargatip = " ",
                b_coordexx = :bx,
```

```
                b_coordeyy = :by,
                b_poste  = :b_poste;
while (existen transformadores ) {
   Se graban los diferentes transformadores
$ INSERT INTO trafos_posicion
      SET  tp_circuito = :b_circuito,
           tp_banco = :b_numero,
           tp_angulo = :tp_angulo,
           tp_tipo = :tp_tipo;
end /* fin del while */
end /* fin del else */
```

Se aclara que en la tabla TRAFOS_POSICION solamente existe el tipo y ángulo de rotación de cada transformador pertenenciente a un banco.

No existe una relación entra la información de la tabla TRAFOS_POSICION con la tabla TRANSFORMADORES así que es imposible saber que número de compañía esta en tal posición en caso de que los transformadores sean más de uno y de igual tipo.

tdesmarca

Se ponen en cero los campos de ubicación del banco y se eliminan los registros correspondientes de la tabla

TRAFOS_POSICION. Esto se hace para poder redefinir el banco.

```
 $ SELECT b_numero INTO :b_numero FROM bancos in da WHERE
b_numero = :b_numero and b_circuito = :b_circuito;
 if ( SQLCODE == 0) then begin
      $ UPDATE bancos
         SET b_coordexx = 0,
             b_coordeyy = 0,
             b_poste = ""
      WHERE b_numero = :b_numero and
            b_circuito = :b_circuito;

      $ DELETE FROM trafos_posicion
      WHERE tp_circuito = :b_circuito and
            tp_banco = :b_numero;
 end
```

trans_capacidad

Se suman las capacidades de los transformadores comprendidos en una ventana.

```
  $ SELECT SUM(t_capacida) INTO :capacidad
  FROM bancos,transformadores,postes
  WHERE
  t_circuito = b_circuito and t_banco = b_numero and
```

```
b_poste <= :pder and b_poste >= :pizq and p_numero =
:b_poste and p_coordexx >= :lizqx and p_coordexx <= :lderx
and p_coordeyy >= :lizqy and p_coordeyy <= :ldery;
```

trans_texto_select

A continuación se presenta la declaración del "cursor" para hacer la selección de los campos que serán presentados en "ACAD" como el bloque de información del banco.

```
$ DECLARE curtra1 CURSOR FOR SELECT
b_circuito,b_numero,b_coordexx,b_coordeyy,t_compania,
t_capacida FROM bancos,transformadores,postes
WHERE
t_circuito = b_circuito and t_banco = b_numero and
b_poste <= :pder and b_poste >= :pizq and p_numero = b_poste
and p_coordexx >= :lizqx and p_coordexx <= :lderx and
p_coordeyy >= :lizqy and p_coordeyy <= :ldery;
```

trans_select

A continuación se presenta la declaración necesaria para construir el "cursor" que extrae la información de los transformadores (posición, ángulo) de cada banco que está comprendido en una ventana.

```
$ DECLARE curtra CURSOR FOR SELECT
tp_angulo,tp_tipo,p_coordexx,p_coordeyy,p_numero,p_tipo
FROM postes,bancos,trafos_posicion
WHERE
tp_circuito = b_circuito and tp_banco = b_numero and
b_poste <= :pder and b_poste >= :pizq and
p_numero = b_poste and p_coordexx >= :lizqx and
p_coordexx <= :lderx and p_coordeyy >= :lizqy and
p_coordeyy <= :ldery;
```

En el menú de bancos en ACAD, al escoger la opción "muestra", AUTOLISP primero manda a ajecutar la rutina "trans_select", para dibujar los transformadores. Luego se llama a la rutina "trans_texto_select", para mostrar la información de cada banco.

Se ocuparon dos rutinas en lugar de una para esta opción por lo complejo del proceso.

APENDICE D

COMUNICACIONES

Al principio del programa "comunifi" se definen las siguientes constantes :

```
LSR = $2FD
LCR = $2FB
TX  = $2F8
RX  = $2F8
```

LSR contiene la dirección del registro de "status" del puerto serial. LCR da la dirección del registro de control. TX y RX direccionan el registro que contiene la informacion (un "byte"), a mandar o recibir desde la computadora central.
Las siguientes instrucciones configuran el puerto serial para transmitir a 9600 baudios, con un "stop bit" sin paridad.

```
Port[lcr] := 128  { se pone un 1 en el bit 7 de LCR }
Port[tx]  := 12   { se aplica un "divisor latch" de 12
                    para poner
Port[lcr] := 3      la comunicacion a 9600 baudios. El 3
                    en LCR configura el puerto para
                    mandar un "stop bit"
                  sin paridad }
```

El siguiente bloque de código se utiliza para mandar un "string" a la computadora central. La variable "largo" contiene

el largo del "string" mandado desde ACAD. Para mandar el "string" , éste se manda "byte" por "byte" con un atraso de ocho milisegundos entre cada carácter porque la computadora "MV-15000" pierde caracteres en la trasmisión.

```
REPEAT
   Delay(8);
   if largo = ind then manda := 10
                  else manda := byte(query[ind]);

   ind := ind + 1;
   ultimo := manda;
   codifica;
UNTIL ind > largo;
```

En la rutina "codifica" se manda cada caracter a la computadora central. En esta rutina antes de manda un carácter se ejecuta la siguiente instruccion :

```
Repeat until (port [lsr] and 64 ) <> 0
```

La cual quiere decir que no se transmita ningún carácter hasta que el registro TX esté desocupado.

El siguiente bloque de código se ocupa para recibir información del programa "queries". Para detectar si se recibió un carácter se verifica si el registro LSR tiene un 1 en el bit 0. Cada

carácter que se recibe se asigna al vector "reg1".

```
  repeat
   if (port[lsr] and 1) = 1
        then begin
             recibe := port[rx];
             cuantos := cuantos + 1;
             reg1[cuantos].info := recibe;
        if reg1[cuantos].info = 38 then  begin
                                        signo := signo + 1;
                                     if signo = 2 then
                                          termino := true;
                                    end;
        end;
  until termino;
```

APENDICE E

ESTACIONES DE TRABAJO

La microcomputadora con la que se cuenta en este momento para la implementación del proyecto de mapeo tiene las siguientes características :

- 640 Kbytes de RAM con 1.5 Mbytes de memoria extendida.
- Una tarjeta de gráficas y un monitor EGA
- 40 Mbytes en disco duro
- Procesador 80286 de 8 Mhz.
- Coprocesador matematico 80287

Los paquetes de "software" con los que se cuenta son Turbo Pascal version 5.5 y ACAD version 10. El sistema operativo que se utiliza es el DOS 3.3 de IBM.

Es necesario tener la memoria extendida porque en caso de faltar ésta, el tamaño máximo de un programa en AUTOLISP es de 45 Kbytes, contando código y datos. La mayor parte de esta memoria se utiliza para la construcción de un RAM DISK, donde reside código en lenguaje de máquina para ACAD (archivos identificados por la terminación OVL) y varios archivos utilitarios. La existencia del coprocesador matemático es necesaria para que pueda funcionar esta versión de ACAD (también para la versión 9).

Esta máquina no es ni por cerca lo óptimo para ocuparse

como estación de trabajo, dada su poca velocidad en el procesamiento de gráficas y una resolución no adecuada para la presentación de éstas. Como herramienta de implementación es medianamente aceptable. Cuando el proyecto ya esté en plena operación, se tendrán verdaderas estaciones de trabajo con una velocidad mínima de 20 mips (millones de instrucciones por segundo) y una tarja de gráficas con una resolución el doble de la actual (la actual es de 640 X 350 pixels).

APENDICE F

RUTINAS EN AUTOLISP

El código que se muestra a continuación corresponde a las partes más importantes descritas en la sección C del capítulo V.

```
(defun pmarcar()
  (command "insert" simbolo pause "" "" ""  )
  (obtiene)
  (setq xmrealp (rtos x 2 1))
  (setq ymrealp (rtos y 2 1))
  (obtiene)
  (setq xmrealt (rtos x 2 1))
  (setq ymrealt (rtos y 2 1))
  (setq entpos (entnext (entlast)))
  (Setq entidad (entget entpos))
  (setq p_numero (cdr (assoc 1 entidad) ) )
  (setq manda  (strcat "P0" utm p_numero xmrealp "-" ymrealp
    "-"  xmrealt "-" ymrealt tipo_poste))
  ( command  "shell" "comunifi")
 )
```

```
(defun pdesmarca()
  (setq poste (entget (car (entsel) ) ) )
  (setq atributo (entget (entnext poste) ) )
  (setq numero (cdr (assoc 1 atributo) ) )
  (obtiene)
  (setq st (strcat "p2" utm numero) )
  (command "shell" "comunifi")
 )

(defun pselect()
  (setq poste (entget (car (entsel) ) ) )
  (setq atributo (entget (entnext poste) ) )
  (setq numero (cdr (assoc 1 atributo) ) )
  (obtiene)
  (setq st (strcat "p2" utm numero) )
  (command "shell" "comunifi")
  )

(defun pmuestra()
  (setq xp (rtos (car pos) 2 1))
  (setq yp (rtos (cadr pos) 2 1))
  (convierte pos)
  (setq poste1 utm)
```

```
(setq xt (rtos (car ultimo) 2 1))
(setq yt (rtos (cadr ultimo) 2 1))
(convierte ultimo)
(setq poste2 utm)
(setq manda (strcat "P6" poste1 "A000" poste2 "D999"  xp
"-" yp "-"  xt "-" yt ))
(command "shell" "comunifi")
(while (and (listp (setq lista1 (car lista))) (not (null
 lista1)))
(setq lista  (cdr lista))
(setq poste  (car lista1))
(setq pl (cadr lista1))
(setq texto (caddr lista1))
(setq tipo_poste (cadddr lista1))
(command "insert" simbolo pl "" "" ""  poste) )
(command "point" (getvar "lastpoint"))
(command "attedit" "" "" "p_numero" "" "c"
(mapcar '- (getvar "lastpoint") (list 4 4))
(mapcar '+ (list 7.0 7.0) (getvar "lastpoint")) "p" texto
 "")
)
)   ; fin de la rutina
```

```
(defun amarcar()
  (setq poste (entget (car (entsel) ) ) )
  (setq atributo (entget (entnext poste) ) )
  (setq p_numero (cdr (assoc 1 atributo) ) )
  (obtiene)
  (command "insert" ancla ( getvar "lastpoint") "" "" pause
  p_numero)
  (setq manda (strcat "A0" utm p_numero angulo tipo_ancla))
  )

(defun adesmarca()
  (command "attext" "e" pause "" "s" "e:\ancla" "e:\archivo")
  (setq f (open "e:\archivo.txt" "r"))
  (setq st (read-line f))
  (close f)
  (setq p_numero (substr st 28 9))
  (setq angulo (substr st 17 3))
   (command "shell" "comunifi") ) )
  )

(defun amuestra()
   (setq ix (rtos (car pos) 2 1) )
   (setq iy (rtos (cadr pos) 2 1 ))
```

```
  (convierte pos)
  (setq poste1 utm)
  (setq dx (rtos (car ultimo) 2 1))
  (setq dy (rtos (cadr ultimo) 2 1))
  (convierte ultimo)
  (setq poste2 utm)
  (setq manda (strcat "A6" poste1 "A000" poste2 "D999"
                      ix "-" iy "-" dx "-" dy          ))
  (command "shell" "comunifi")
  (while (and (listp (setq lista1 (car lista))) (not (null
   lista1)))
   (command "insert" ANCLA pl "" "" angulo "" )
   )
   )   ; fin de la rutina

(defun tmuestra()
  (setq ix (rtos (car pos) 2 1))
  (setq iy (rtos (cadr pos) 2 1))
  (convierte pos)
  (setq poste1 utm)
  (setq dx (rtos (car ultimo) 2 1))
  (setq dy (rtos (cadr ultimo) 2 1))
  (convierte ultimo)
```

```
(setq poste2 utm)
(setq manda (strcat "T6" poste1 "A000" poste2 "D999"
                    ix "-" iy "-" dx "-" dy          ))
(setq manda2 (strcat "T5" poste1 "A000" poste2 "D999"
                    ix "-" iy "-" dx "-" dy          ))
(command "shell" "comunifi")
(while (and (listp (setq lista1 (car lista))) (not (null
 lista1)))
 (setq lista  (cdr lista))
 (setq tipo_transf (car lista1))
 (cond ( (=  tipo_transf "A") (setq trafo "transf") )
       ( (= tipo_transf "B") (setq trafo "fm") )
       ( (= tipo_transf "C") (setq trafo "ap") )

       ( t t) ) ; find de cond
(command "insert" trafo (cddr lista1) "" "" (cadr lista1)
"")
   ) ; fin del while
(setq f (open "e:\archivo" "w"))
 (write-line manda2 f)
 (close f)
 (command "shell" "comunifi")
 (while (and (listp (setq lista1 (car lista))) (not (null
  lista1)))
```

```
(setq lista  (cdr lista))
(command "insert" "trafo_bloque" (cdddr lista1) "" "" "" (caddr
lista1) (cadr lista1) (car lista1) )
      )   fin del while
    )     fin de la rutina

(defun trafo_bloque()
(command "setvar" "attdia" "1")
(command "insert" "trafo_bloque" (getvar "lastpoint") "" "" "")
(command "setvar" "attdia" "0")
(command "attext" "e" (getvar "lastpoint") "" "s" "E:\TRAFOB"
"e:\archivo")
(setq banco (substr st 1 10))
(SETq coordexx (substr st 11 8))
(setq coordeyy (substr st 19 8))
(setq manda (strcat "T0" p_numero banco  coordexx))
(setq manda (strcat manda  coordeyy  angulo1 tipo1 ))
(setq manda (strcat manda angulo2 tipo2  angulo3 tipo3))
(command "shell" "comunifi")
)

(defun t_capacidad()
  (setq manda (strcat "T7" poste1 "A000" poste2 "D999"
```

```
                        ix "-" iy "-" dx "-" dy          ))
  (command "shell" "comunifi")
   )

(defun lineas_primar()
 (setq poste1 (car (entsel)))
 (setq poste1 (entget poste1))

 (setq poste2 (car (entsel)))
 (setq poste2 (entget poste2))
 (command "circle" centro_poste1 radio1)
 (command "circle" centro_poste2 radio2)
 (command "line" centro_poste1 centro_poste2 "")
 (command "trim" circulo1 "" centro_poste1 "")
 (command "trim" circulo2 "" centro_poste2 "")
 (command "layer" "off" "referencia" "")
 (command "insert" "bloque_linea" (getvar "lastpoint") "" ""
  "")
 ( if (> (strlen  texto_fase)  0) (progn
        (command "attedit" "" "" "fase" "" punto_fase1 ""
         "p" pause "")
        (command "attedit" "" "" "fas1" "" (getvar
      "lastpoint") ""
```

```
        "a" pause "")
         ) ); end del progn y del if

(command "color" color_linea )
(cond ((= tipo_linea "A") (setq string_linea "¦o¦") )
      ((= tipo_linea "N") (setq string_linea "") )
      (t t)            ) ;fin del cond
(command "setvar" "attdia" "0")
(command "insert" "lineas" punto1 distancia "1" angulo
         string_linea cod_poste1 cod_poste2 texto_fase1
         texto_fase2 texto_fase3 texto_neutral tipo_linea )
 (setq e1 (entlast))
(if (= tipo_linea "A") (progn
            (setq e2 (entnext e1))
            (setq ed (entget e2))
            (setq ed
                  (subst (cons 41 1)
                         (assoc 41 ed) ed ) )
            (entmod ed)
            (entupd e1) ) ) fin del progn y del if
 ) ; de la rutina
```

```
(defun lineas_secund()
    Se marca primero el poste 1
 (setq poste1 (car (entsel)))
 (setq poste1 (entget poste1))
 (setq layer_actual (cdr (assoc 8 poste1))
;   Ahora se extrae información del poste2
 (setq poste2 (car (entsel)))
 (setq poste2 (entget poste2))
 (command "circle" centro_poste1 radio1)
 (setq circulo1 (entlast))
 (command "circle" centro_poste2 radio2)
 (setq circulo2 (entlast))
 (if (and (> (sin angulo_radianes) -0.731353)
     (< (sin angulo_radianes) 0.731353) )  ; comienza el then
       (progn
        (setq opcion1p1 (mapcar '+ '(0 0.5) centro_poste1) )
        (setq opcion1p2 (mapcar '+ '(0 0.5) centro_poste2) )
        (setq opcion2p1 (mapcar '- centro_poste1 '(0 0.5) ) )
        (setq opcion2p2 (mapcar '- centro_poste2 '(0 0.5) ) )
       ) fin del progn y a continuación comienza el else
       (progn

        (setq opcion1p1 (mapcar '+ '(0.5 0) centro_poste1) )
        (setq opcion1p2 (mapcar '+ '(0.5 0) centro_poste2) )
```

```
        (setq opcion2p1 (mapcar '- centro_poste1 '(0.5 0) ) )
        (setq opcion2p2 (mapcar '- centro_poste2 '(0.5 0) ) )
      ) fin del progn
  ) fin del if
 (command "osnap" "tan")
 (command "line" opcion1p1 opcion1p2 "")
 (setq opcion1 (entlast) )
 (command "line" opcion2p1 opcion2p2 "")
 (setq opcion2 (entlast) )
)
```