UNIVERSIDAD DE CARABOBO

FACULTAD DE INGENIERIA

DEPARTAMENTO DE POTENCIA

DISEÑO Y CODIFICACIÓN DE UN PROGRAMA PARA EL CÁLCULO DE SECTORES DE BAJA TENSIÓN

TRABAJO ESPECIAL DE GRADO PRESENTADO ANTE LA ILUSTRE UNIVERSIDAD DE CARABOBO PARA OPTAR AL TITULO DE INGENIERO ELECTRICISTA

CORDERO G. MARVIN D.

PRIETO F. JESÚS A.

Valencia Abril del 2000.

UNIVERSIDAD DE CARABOBO

FACULTAD DE INGENIERIA

DEPARTAMENTO DE POTENCIA

CERTIFICADO DE APROBACIÓN

Los abajo firmantes, miembros del jurado designado para estudiar el Trabajo Especial de Grado titulado "**DISEÑO Y CODIFICACIÓN DE UN PROGRAMA PARA EL CÁLCULO DE SECTORES DE BAJA TENSIÓN**", realizado por los bachilleres: Cordero Marvin y Jesús Prieto hacemos constar que hemos revisado y aprobado dicho trabajo.

Prof. José Raga

Presidente

Prof. Cesar Ruiz

Prof. Francisco Naveira

DEDICATORIA

Dedico este trabajo a DIOS nuestro señor cuya grandeza me ha permitido lograr este paso tan importante en mi vida.

Dedico este trabajo a mi Madre y mi Abuela personas que me han guiado por todos los caminos y metas que me he trazado y estoy seguro que lo seguirán haciendo.

Dedico este trabajo a mis tres hermanos Carlos, Alejandro y Roberto apoyos con los que cuento incondicionalmente en todo momento.

Dedico este trabajo a mis tíos y tías, primos y primas personas que forman parte inseparable en mi vida.

Dedico este trabajo a todas mis amistades especialmente a los de mi escuela y todas aquellas personas que de una u otra forma ayudaron a la realización de este trabajo, y que ahora son y siempre serán parte de mi familia.

Marvin Cordero.

DEDICATORIA

Dedico por sobre todas las cosas este trabajo a Dios todo poderoso, quien me ha dado la energía para mantenerme en pie a pesar de las vicisitudes y cuando he caído me ha dado la fuerza necesaria para seguir adelante.

Dedico este trabajo a toda mi familia y en especial a la memoria de mi madre quien trabajó he hizo muchos sacrificios para que yo me superara, quien siempre me inspiró, me alentó a luchar y me enseñó todo lo bueno que sé con el mejor de los métodos "El Ejemplo", a ti mi vieja con lágrimas en mi corazón por no tenerte a mi lado y no poder compartir contigo este momento que toda madre espera, dedico mi trabajo que es poco para todo lo que me diste y allá donde estas ahora se que estarás feliz por este momento que mas que mio es tuyo.

También dedico este trabajo a todos aquellos que luchan por superarse en la vida, que a pesar de los problemas que se le presentan se esfuerzan por aprender, por crecer como personas, como seres humanos y como profesionales, por y para aquellas personas que saben que la vida es una escuela y el mundo un mar de oportunidades.

Jesús Prieto.

AGRADECIMIENTOS

Agradecemos a DIOS todo poderoso por guiarnos durante este camino y permitirnos lograr culminarlo con todo éxito, anhelando su presencia y apoyo en nuestras vidas durante todo momento.

Agradecemos al profesor Cesar Rodolfo Ruiz por su guía y cooperación prestada para la elaboración de nuestro trabajo de grado.

De igual forma agradecemos a nuestro tutor profesor José Raga por su ayuda y apoyo para con nosotros.

Agradecemos al maestro "Avila" por su guía durante toda nuestra carrera.

Agradecemos a nuestro compañero y amigo Juan Carlos Mota por la ayuda prestada en las distintas etapas de nuestra carrera.

Y por último agradecemos a nuestros amigos y hoy colegas Ing. Héctor Rivas, Ing. Victor Carrera por su ayuda prestada en este trabajo.

INDICE GENERAL

INDICE DE FIGURAS.

INTRODUCCIÓN

El cálculo de caida de tensión en las redes de distribución, tanto para fines de diseño como para evaluación, es un proceso que por métodos manuales resulta engorroso ya que requiere una gran cantidad de trabajo para resolver parcialmente el problema, por ejemplo, al diseñar un sector de baja tensión de una zona residencial lo que se hace tradicionalmente es obtener las distancias de cada tramo a partir del plano urbanístico de la zona y con estas calcular las impedancias que representan a estos tramos para un conductor dado, luego se plantean y resuelven las ecuaciones para calcular las caidas de tensión en la red y si los resultados no son satisfactorios se repite el proceso para otro conductor u otra disposición lo cual representa obtener otras longitudes distintas del plano. Hasta hace poco tiempo era la forma de realizarlo, gracias al surgimiento de nuevas herramientas de trabajo en el área de la informática surge la posibilidad de desarrollar soluciones innovadoras a problemas complicados.

Por otra parte la demanda de energia varia significativamente con el pasar del tiempo produciendo cambios en los parámetros eléctricos de dichas redes en donde el termino caida de tensión juega un papel de suprema relevancia en el deterioro de la calidad del servicio eléctrico. Para solucionar este problema se hace necesario el rediseño de las redes de distribución.

El software VBCAD permite el diseño y el rediseño de redes de distribución aéreas de baja tensión sobre un plano urbanístico hecho en AutoCAD 14 o superior, reduciendo de manera significativa el tiempo empleado para diseñar y evaluar las

los métodos manuales tradicionales. Este programa es de fácil manejo por lo cual también se puede utilizar como herramienta de apoyo didáctico para materias de la carrera de Ingeniería Eléctrica tales como: Transmisión de Energía I, Sistemas de Distribución y Sistemas Industriales I y II, entre otras.

Debido a la gran cantidad de herramientas y ventajas disponibles, unida a la facilidad de manejo, se escogió el lenguaje de programación Microsoft Visual Basic para desarrollar VBCAD.

Este trabajo de grado está dispuesto de la siguiente manera: el primer capitulo contiene los fundamentos teóricos de la Matriz Impedancia y Admitancia y su relación con el término caída de tensión, el segundo capitulo comprende el código fuente del programa junto con un diagrama de flujo general del mismo y una corrida del programa, y el tercer capitulo es dedicado al manual de usuario del programa.

CAPÍTULO 1

MATRIZ IMPEDANCIA Y MATRIZ ADMITANCIA

1.1 METODO NODAL.

Se llaman *nodos* a las uniones formadas cuando dos o más elementos de circuito (R, L, C o una fuente ideal de voltaje o corriente) se conectan en sus terminales. La formulación sistemática de ecuaciones, determinada en los nodos de un circuito al aplicar la ley de corrientes de Kirchhoff, es la base de algunas excelentes soluciones computacionales de los problemas de sistemas de potencia en los distintos niveles de tensión.

Con el fin de examinar algunos aspectos de las ecuaciones de nodos se estudiara el circuito simple de la Figura 1.1.

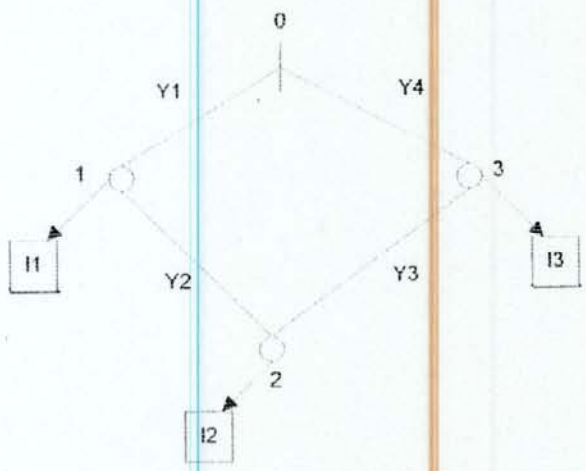


Figura_1.1. Circuito Simple.

Al aplicar la ley de corriente de Kirchhoff , con la suma de corrientes saliendo igual a la suma de corrientes que entran se obtiene:

$Y1 * (V0 - V1) - (Y2 * (V1 - V2)) = I1$

$Y2 * (V1 - V2) \quad (Y3 * (V2 - V3) = I2$

$Y3 * (V2 - V3) - (Y4 * (V3 - V0) = I3.$

Expresando Matricialmente las ecuaciones se obtiene:

$$\begin{bmatrix} I1 \\ I2 \\ I3 \end{bmatrix} = V0 * \begin{bmatrix} Y1 \\ 0 \\ Y4 \end{bmatrix} - \begin{bmatrix} (Y1 + Y2) & -Y2 & 0 \\ -Y2 & (Y2 + Y3) & -Y3 \\ 0 & -Y3 & (Y3 + Y4) \end{bmatrix} * \begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix}$$

Por lo que:

$$[I] = V0 * [Y_{ref}] - [Y_{BARRA}] * [V] \quad (a)$$

En donde:

V : Representa el vector columna de los voltajes de los nodos.

I : Representa el vector columna de las corrientes que salen de los nodos.

Y_{ref} : Representa la admitancia de los elementos de la red que están conectados al nodo de referencia.

V0 : Representa la tensión del nodo de referencia.

Y_{BARRA} : Matriz Admitancia de la red.

Al invertir Y_{BARRA} se obtiene una matriz importante denominada Matriz Impedancia o Z_{BARRA}.

Es decir: $Z_{BARRA} = Y_{BARRA}^{-1}$

Por lo tanto la ecuación (a) se transforma en:

$$[V] = [Z_{barra}] * V0 * [Y_{ref}] - [Z_{barra}] * [I]$$

Convirtiéndose la ecuación (a) en una valiosa herramienta para el calculo de las tensiones en los nodos de cualquier red independientemente del tamaño de la misma.

Las matrices Y_{BARRA} y Z_{BARRA} serán explicadas explícitamente en los siguientes apartados.

1.2 MATRIZ ADMITANCIA

La Matriz Admitancia Nodal o Matriz Y_{BARRA} como también suele denominarse es un modelo matemático que da el comportamiento en estado estable de todas las componentes que actúan juntas para formar el sistema y se basa en el análisis nodal de las ecuaciones de la red, la Matriz Admitancias presenta las características siguientes:

- En una matriz cuadrada de orden n (n = numero de nodos).
- Es simétrica con respecto a la diagonal principal ($Yij = Yji$).
- Es una matriz poco densa (*tiene muchos ceros*).
- Todos los elementos de la diagonal principal son negativos y los elementos fuera de la diagonal principal son positivos.

Por lo que los elementos de la Matriz Admitancia se pueden expresar en dos términos:

- $Yii = \sum yij$ (Sumatoria de todas las admitancias j que están en el nodo yii).
- $Yij = -yij \quad (i \neq j)$.

Tomando en cuenta que el factor más significativo de la Matriz Admitancia Nodal es que con ella se puede encontrar la Matriz Impedancia apoyándose en la siguiente igualdad ($Z = Y^{-1}$) , no obstante esto no resulta ser siempre lo más conveniente.

La construcción de la Matriz Admitancia es bastante sistemática, lo cual permite la preparación de un algoritmo de cálculo de sus elementos.

Este es un algoritmo genérico, contempla el acoplamiento magnético que existe entre las ramas, éste fenómeno no se presenta en las redes de distribución debido a los bajos niveles de corriente con que se trabaja.

En fin el algoritmo consiste en ir conectando tramo a tramo los elementos de la red que se irán convirtiendo en filas y columnas de una matriz que al final resultará ser la Matriz Admitancia, en definitiva la Matriz Admitancia es la matriz de coeficientes de tensión en el método nodal de resolución de redes.

1.3 MATRIZ IMPEDANCIA

La Matriz Impedancia o Matriz Z_{BARRA} como también suele llamarse, presenta características similares a la Matriz Admitancia, hecho que no debe impresionar debido a la procedencia de la misma ($Z = Y^{-1}$). Sin embargo la Matriz Z_{BARRA} se puede encontrar directamente, apoyandose en un algoritmo genérico.

Este algoritmo presenta cuatro casos que son los siguientes:

- Elemento entre referencia y un nodo nuevo.
- Elemento entre un nodo existente y un nodo nuevo.
- Elemento entre dos nodos existentes.
- Elemento entre un nodo existente y referencia.

Los cuales serán explicados en el siguiente apartado (1.4), y a diferencia del caso de la Matriz Admitancia irán conectando tramo a tramo los elementos de la red según contemple el caso en particular, dando como resultado la Matriz Impedancia.

También se debe tomar en cuenta el hecho de que las redes de distribución son redes generalmente grandes y con éste algoritmo directo se facilitará el estudio de las redes y se

volverá más viable y menos engorroso el trabajo, que el calcular la Matriz Y_{BARRA} y luego su inversa.

1.4 CASOS DE LA MATRIZ IMPEDANCIA

1.4.1 Caso 1: Añadir la Zb de un nodo nuevo (p) al nodo de referencia:

La adición de un nodo nuevo (p) al nodo de referencia a través de su impedancia Zb y la inyección de una corriente Ip al nuevo nodo no alteran la tensión Vp determinada como Vp = Ip * Zb.

Se agrega una fila y una columna nueva a la Matriz original.

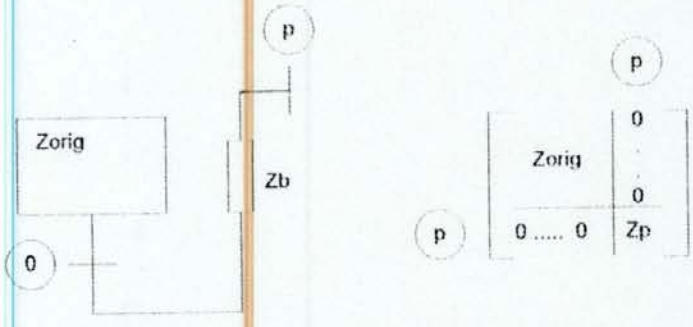


Figura_1.2. Adición de Zb entre un nodo nuevo y referencia.

1.4.2 Caso 2: Añadir la Zb de un nodo nuevo (p) a un nodo existente (k):

La adición de un nodo nuevo (p) a un nodo existente (k) a través de su impedancia Zb junto con la corriente Ip conectada al nuevo nodo provocará un incremento de tensión en el

nodo existente (k) dado como Vk= Vk° + IpZkk por lo que Vp será mayor que la nueva Vk por una cantidad dada de IpZb resultando Vp= Vk° + IpZkk + IpZb.

Se agrega una fila y una columna nueva, donde los primeros N elementos de la nueva fila son los elementos de la K-ésima fila de Zorig y que los N primeros elementos de la nueva columna son los elementos de la K-ésima columna de Zorig.

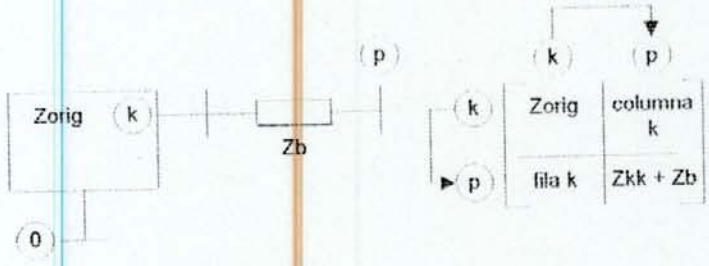


Figura_1.3. Adición de Zb entre un nodo nuevo y un nodo existente.

1.4.3 Caso 3: Añadir Zb entre dos nodos existentes (j) y (k):

La adición de una impedancia Zb entre dos nodos ya establecidos en la matriz Zorig produce la aparición de una columna y una fila provisional, en donde la nueva columna es la j menos la columna k de la matriz Zorig con Zbb (Zbb= Zjj + Zkk –2Zjk +Zb) en la fila (N + 1). La nueva fila es la transpuesta de la nueva columna.El carácter de provisional que toma esta columna y fila nueva se debe a la reducción de Kron, la cual se aplica a cada elemento de la matriz existente, y se observa que cada elemento Zhi(nueva) en la nueva matriz es:

$$Zhi(nueva) = ((Zhi) - (Zh(N+1) Z(N+1)i) / (Zjj + Zkk - 2Zjk + Zb)).$$

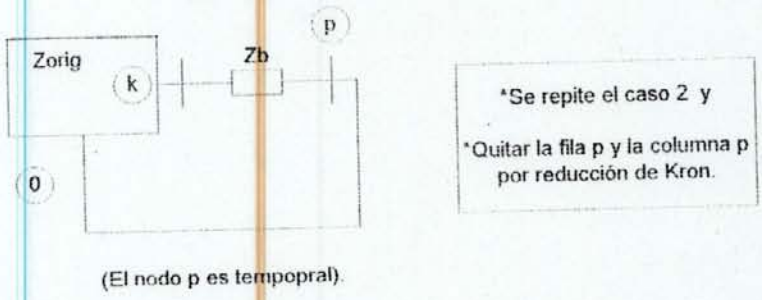


Figura_1.4. Adición de Zb entre dos nodos existentes.

1.4.4 Caso4: Añadir Zb desde un nodo existente (k) al nodo de referencia.

En este caso se aplica el mismo proceso del caso 2 y luego la reducción de Kron, es decir se adiciona un nodo (p) con su impedancia Zb conectado al nodo k, luego se cortocircuita el nodo (p) al nodo de referencia logrando que Vp = 0 y creando una fila y columna nueva que será reducida por Kron.

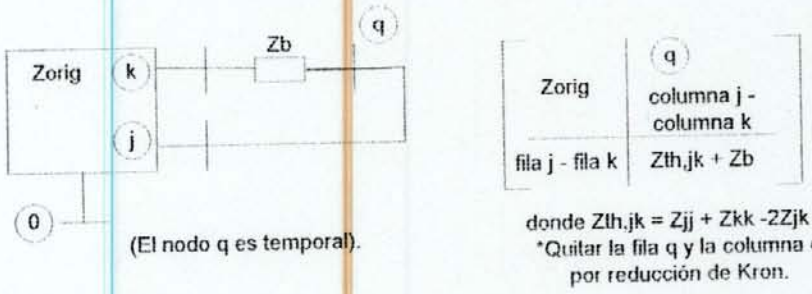
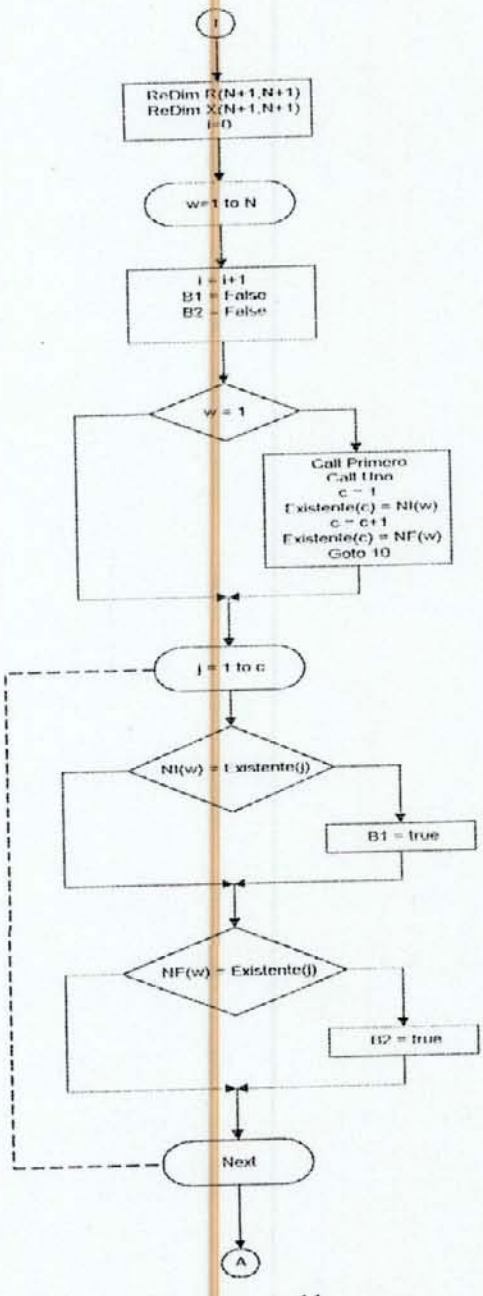


Figura_1.5. Adición de Zb entre un nodo existente y referencia.

1.5 DIAGRAMAS DE FLUJO DE LAS MATRICES IMPEDANCIA Y ADMITANCIA.

1.5.1 Matriz Z_{BARRA}:

En la Fig.1.6. se muestra el diagrama de flujo del programa VBCAD para el cálculo de la Matriz Z_{BARRA}:



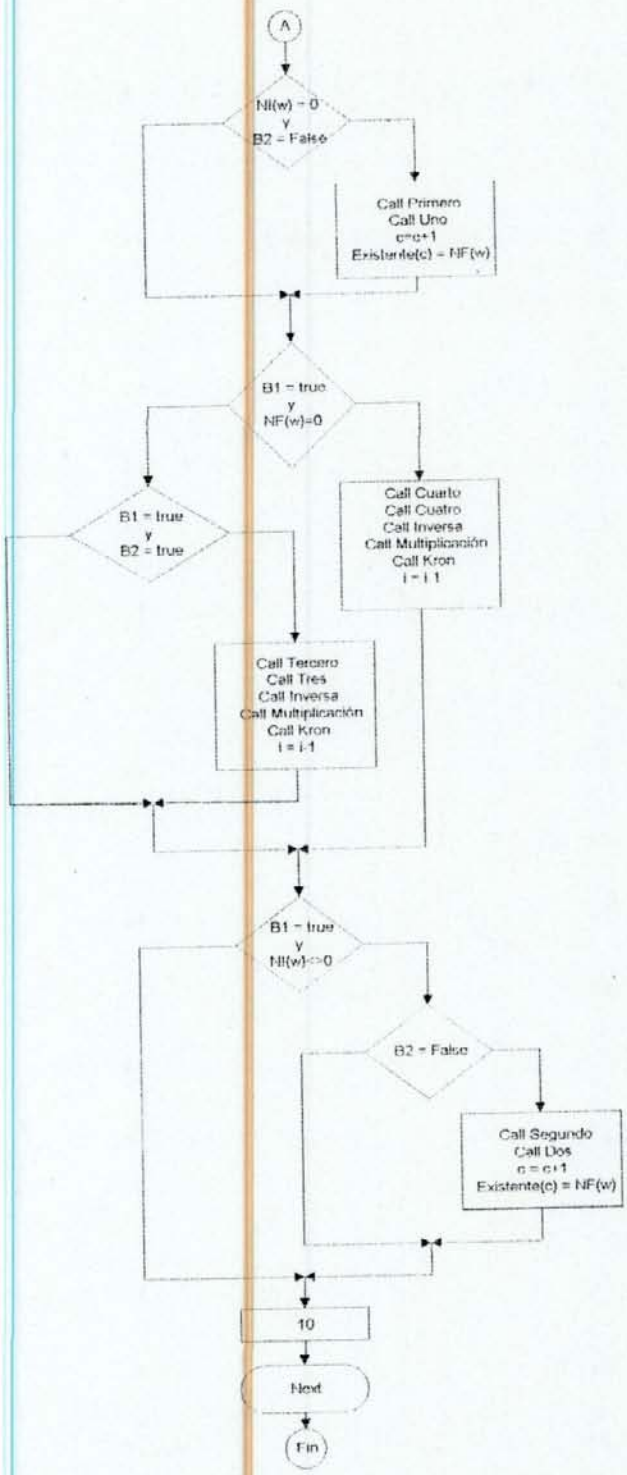


Figura_1.6. Diagrama de Flujo de la Matriz Z_{BARRA}.

Como se observa en el diagrama de flujo de la Figura_1.6. en este programa y en general en todos los realizados en este trabajo se utilizaron subprogramas que realizan funciones especificas, por lo que los mismos son pequeños programas dentro del programa completo que se hicieron de esta manera para simplificar la programación. En las explicaciones de los diagramas de flujo solo se mencionara la función que realiza cada subprograma dentro de cada programa principal.

El programa de cálculo de la Matriz Z_{BARRA} se basa en el algoritmo conocido del libro de Análisis de Sistemas de Potencia de Grainger y Stevenson con algunas modificaciones. A continuación se presentará de manera secuencial una explicación de este programa.

1. Lo primero que se hace en el programa es dimensionar las matrices que contienen la parte real e imaginaria de la Matriz Z_{BARRA}, como Visual Basic nos permite dimensionar las matrices de una manera dinámica, o dicho de otra manera, cambiando el tamaño reservado en la memoria de las mismas según la necesidad de cada problema, es posible utilizar la cantidad de memoria estrictamente necesaria para cada caso. Además se inicializa el contador "i" en cero.

2. Se inicializa un bucle "For" con la variable "w" (hasta el valor de "N", el cual representa el número de tramos) como contador en el cual cada vez que esta variable incrementa su valor el contador "i" también lo hace y además se le asignan el valor falso a las banderas B1 y B2. Dentro de este mismo bucle se pregunta si el valor de "w" es igual a uno, de ser afirmativa la respuesta se hace una llamada al procedimiento "Primero", el cual representa el primer caso, de la parte imaginaria, para la obtención de la Matriz de Impedancia

(Z_{BARRA}). La llamada al procedimiento "uno" realiza la misma operación que "Primero" pero para la parte real. Además sé inicializa el contador "c" en el valor de uno y se asignan al vector Existente los nodos inicial y final del tramo inicial, también se produce un salto de línea para salir del de la condición "If".

3. Dentro del bucle "For " de "w" sé inserta otro bucle "For" con "j" como contador, cuya función es buscar si algún nodo del tramo en estudio está guardado dentro del vector "Existente".

4. Se pregunta si se está en el primer caso, de ser afirmativa la respuesta se hace un llamado a los procedimientos "Primero" y "uno" y se asigna al arreglo vectorial "Existente" el nodo final del tramo.

5. Luego se verifica si se está en el cuarto caso, de ser cierta la respuesta se llama a los procedimientos "Cuatro" y "Cuarto" los cuales encuentran la parte real e imaginaria, respectivamente, de la Matriz Z_{BARRA}, además, se llaman a los procedimientos "Kron", "Inversa" y "Multiplicación", que tienen como función realizar la reducción de Kron necesaria para este caso. Después se reduce en uno el contador "i". De ser negativa la pregunta de, si se está en el cuarto caso, entonces se pregunta si se está en el tercero, si la respuesta es afirmativa se realiza la llamada a todos los procedimientos para realizar este caso, vale decir, "Tercero", "Tres", "Inversa", "Multiplicación" y "Kron".

6. Por último se pregunta si se está en el segundo caso y si la respuesta es afirmativa se llama a los procedimientos "Segundo" y "Dos, se incrementa en uno el contador "c" y se asigna al arreglo vectorial "Existente" el nodo final del tramo en estudio.

1.5.2 Matriz Y_{BARRA}:

A continuación se muestra el diagrama de flujo de la Fig.1.7. del programa VBCAD para el cálculo de la Matriz Y_{BARRA}.

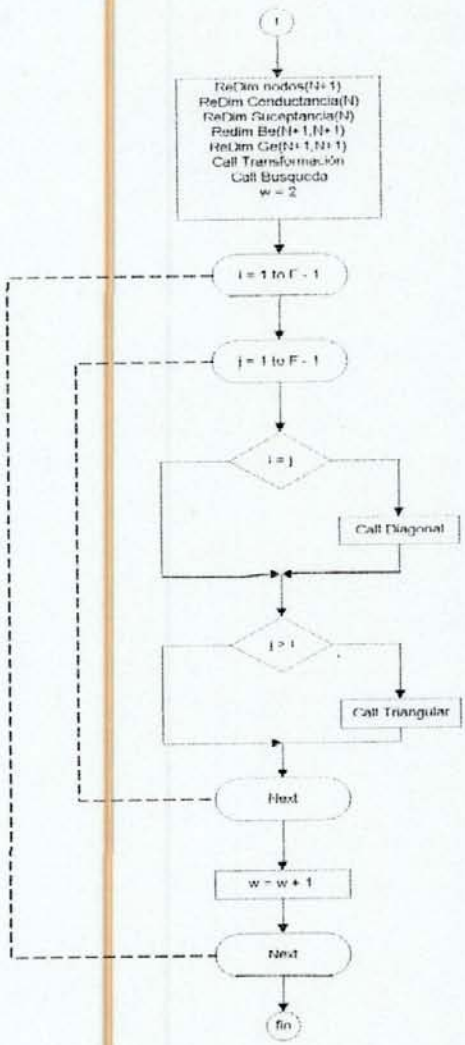


Figura_1.7. Diagrama de Flujo de la Matriz Y_{BARRA}.

1. Lo primero que se hace en este programa es dimensionar los vectores "nodos", "Conductancia", "Suceptancia" y las matrices "Be" y "Ge", las cuales contendrán la parte real e imaginaria de la Matriz Y_{BARRA}, también se hace una llamada a los procedimientos "Transformación" y "Busqueda", cuyas funciones son convertir las impedancias de la red en admitancias y buscar el número de nodos de esta respectivamente. Además se inicializa el contador "w" en dos.

2. Se inicializa un bucle "For" con "i" como contador y hasta el valor de "F-1" siendo "F" el número de nodos de la red.

3. Dentro del Bucle anterior se inserta otro bucle con las mismas características que el anterior pero con "j" como contador. Aqui se pregunta si se está en la diagonal principal de la Matriz Y_{BARRA}, si la respuesta es afirmativa se hace una llamada al procedimiento "Diagonal" el cual calcula los elementos de la Matriz Y_{BARRA} para este caso.

4. Luego se pregunta si se esta en la parte triangular superior de la Matriz Y_{BARRA} (por encima de la diagonal principal), de ser afirmativa la respuesta se procede a llamar al procedimiento "Triangular" el cual calcula los elementos de la matriz fuera de la diagonal principal.

5. Después de cerrar el "For" cuyo contador es "j" se incrementa el contador "w" en uno y se cierra el otro bucle.

1.6 COMENTARIOS

La Matriz Impedancia presenta mayor información que la Matriz Admitancia, debido a la forma de impedancia de thevenin que tiene del sistema, cabe destacar que los elementos

presentes en la diagonal principal de la Matriz Z_{BARRA} representan los equivalentes thevenin de los nodos, ubicados en la fila y columna correspondiente dentro de la Matriz Z_{BARRA} es decir Zth = Zkk en donde "k" representa el nodo de la red.

La Matriz Impedancia representa una gran herramienta en el estudio de análisis de fallas. Sin embargo como se pudo observar en los diagramas de flujos, el algoritmo para la construcción de la Matriz Admitancia es más sencillo que el de la Matriz Impedancia por lo que facilita su obtención.

1.7 CAIDA DE TENSIÓN

El termino Caída de Tensión representa la diferencia en magnitudes absolutas, y no fasorialmente, del voltaje entre los dos terminales de la red. Apoyándose en un análisis nodal aplicado a una red sencilla de 3 nodos y 4 tramos, como se muestra en la Fig.1.8:

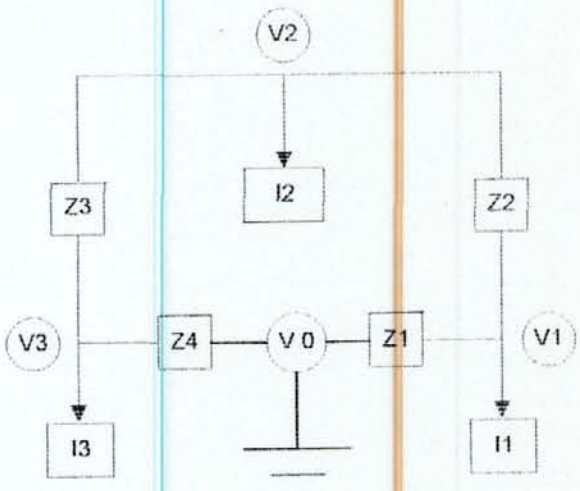


Figura_1.8. Aplicación a la red del Análisis Nodal.

Con ésta simple red se puede demostrar la expresión genérica de la caída de tensión, la cual es utilizada por el programa (VBCAD).

Planteando las siguientes ecuaciones:

$$(V0 - V1) / Z1 = I1 + (V1 - V2) / Z2$$

$$(V1 - V2) / Z2 = I2 + (V2 - V3) / Z3$$

$$(V2 - V3) / Z3 = I3 + (V3 - V0) / Z4$$

Teniendo en cuenta que:

$$Y = V*I$$

Se obtiene:

$$I1 = Y1 (V0 - V1) - Y2 (V1 - V2)$$

$$I2 = Y2 (V1 - V2) - Y3 (V2 - V3)$$

$$I3 = Y3 (V2 - V3) - Y4 (V3 - V0)$$

Luego:

$$I1 = ((V0 * Y1) + (V1 * (-Y1 - Y2)) + (V2 * Y2))$$

$$I2 = ((V1 * Y2) + (V2 * (-Y2 - Y3)) + (V3 * Y3))$$

$$I3 = ((V0 * Y4) + (V2 * Y3) + (V3 * (-Y3 - Y4)))$$

Expresándolo en forma Matricial:

$$[I] = V0 * [Y_{ref}] - Y_{barra} * [V]$$

Conociendo que:

$$Z_{barra} = Y_{barra} \wedge -1$$

Se obtiene:

$$[Z_{barra}] * [I] = [Z_{barra}] * V0 * [Y_{ref}] - [V]$$

Por ultimo:

$$[V] = [Z_{barra}] * V0 * [Y_{ref}] - [Z_{barra}] * [I]$$

En donde:

V : Representa el vector columna de los voltajes de los nodos.

I : Representa el vector columna de las corrientes que salen de los nodos.

Yref : Representa la admitancia de los elementos de la red que están conectados al nodo de referencia.

V0 : Representa la tensión del nodo de referencia cuyo valor es 120 volts monofásico.

Z_{BARRA} : Matriz Impedancia de la red.

Y_{BARRA} : Matriz Admitancia de la red.

Cabe destacar que la expresión de la caída de tensión en forma porcentual se expresa de la siguiente manera:

$$\%Vi = ((Vn - Vi) / Vn) * 100$$

En donde:

Vi : Representa la tensión en un nodo determinado.

Vn : Representa la tensión nominal.

Conociendo las cargas en cada nodo, y conociendo el factor de potencia él cual se asume igual para todas las cargas por ser muy similar el valor de las mismas, se procede a encontrar las corrientes como:

$$I = (KVA / (\sqrt{3} * 0{,}208)) \quad o \quad I = (KW / (\sqrt{3} * 0{,}208 * fp))$$

En donde:

KVA : Potencia aparente de la carga.

KW : Potencia útil de la carga.

Fp : Factor de potencia de la carga.

I : Corriente de la carga.

Conociendo la Matriz Impedancia de la red, se procede a encontrar la tensión de cada nodo (Vi).

$$[Vi] = [Z_{barra}] * V0 * [Y_{ref}] - [Z_{barra}] * [I]$$

Cabe destacar que la expresión anterior de [Vi] representa el equivalente monofásico de la red por lo que:

$$[Vi] = \sqrt{3} * \{ [Z_{barra}] * V0 * [Y_{ref}] - [Z_{barra}] * [I] \}$$

Y luego para cada nodo se tiene:

% Vi = ((208 – Vi) / 208) * 100.

CAPÍTULO 2

CODIFICACIÓN DEL PROGRAMA

2.1 DIAGRAMA DE BLOQUES DEL PROGRAMA VBCAD.

VBCAD es el nombre identificador que se decidió darle al software de Cálculo de Caída de Tensión en sectores de baja tensión, el nombre se debe al enlace o interfaz que existe entre los programas Visual Basic y AutoCAD.

A continuación se presenta el diagrama de bloques (Figura_2.1) que representa en forma bastante simplificada la operación del programa VBCAD.

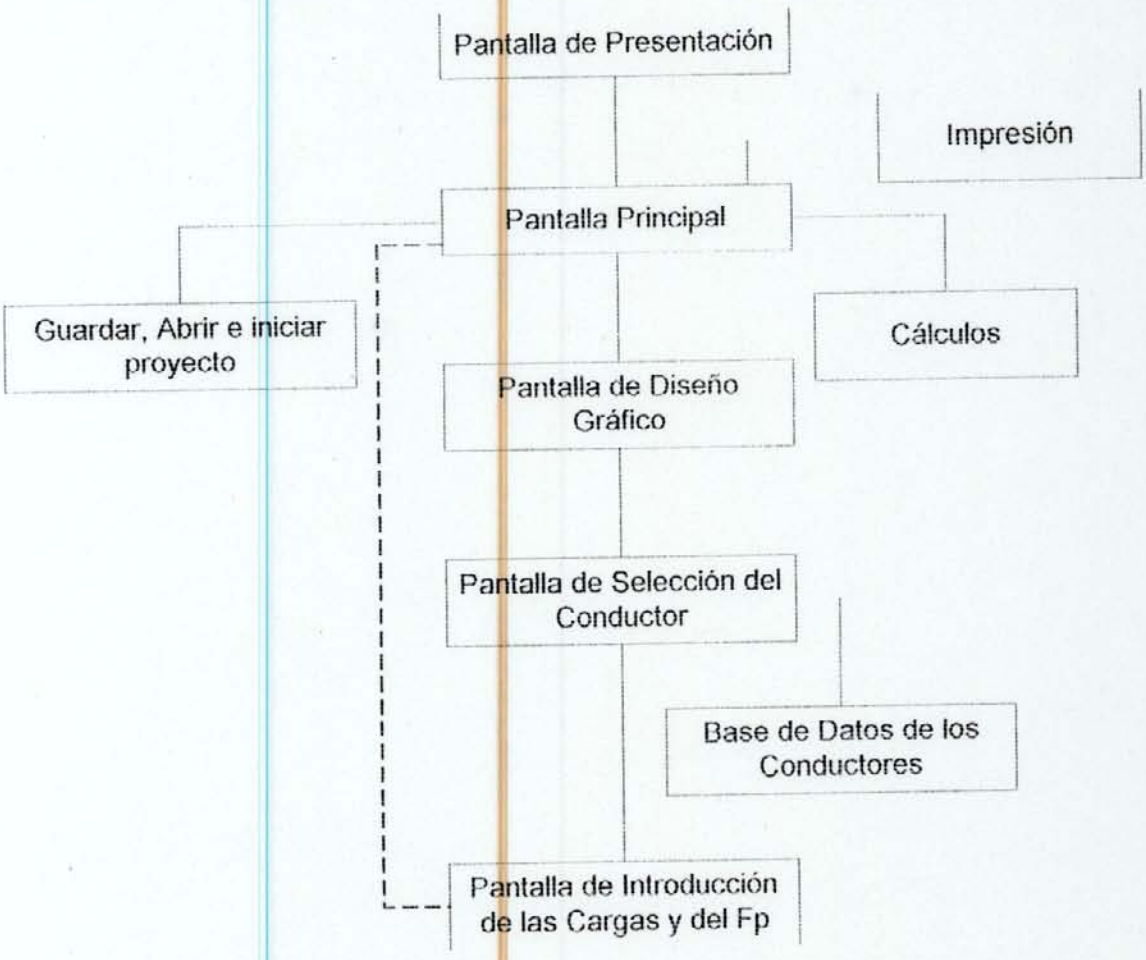


Figura_2.1. Diagrama de Bloques del Programa VBCAD.

2.2 CODIGO DE VBCAD.

```
Dim Color As String, CambiaProyB As Boolean, B1A As Boolean
Dim X1 As Single, Y1 As Single, X2 As Single, Y2 As Single
Private Sub Abrir_Click()

   DibujaProB = False
   Call Invisible
   Form1.Picture1.Visible = True
   Call Abrir_mzc

End Sub

Private Sub AbrirProyec_Click()

   Call ASuper_Proyec
   CambiaProyB = False: Command3.Enabled = True: Guardar_CamB = False

End Sub



Private Sub CaidadetensiónG_Click()

   If (N <> 0) Then
      VoltajeB = False
      Form1.MousePointer = vbHourglass
      Form1.Picture1.Visible = True
      Call Invisible
      'BAND es igual a 2 para poder realizar este procedimiento
      BAND = 2
      If AbrirlosB = False Then
         Call Delta_V
      End If
      Form1.Picture1.Cls
      Call Dibujar_dxf(Me.Picture1)
      Call Dibujo_esp(Me.Picture1)
      Call Numerar_nodos(Me.Picture1)
      BAND = 0
      Form1.MousePointer = vbDefault
   End If

End Sub
```

```
Private Sub Command1_Click()

   If BAND <> 0 Then
      BAND = 0
   End If
   Call Nuevos

End Sub
Private Sub Command2_Click()

  Call ASuper_Proyec
  DibujaProB = True
  CambiaProyB = False: Command3.Enabled = True: Form1.Command4.Enabled =
True

End Sub
Private Sub Command3_Click()

   If VoltajeB = True Then
      i = MsgBox("Para Guardar la Matriz Actual se Deben Calcular Tanto las Caidas
de Tensión Como las Corrientes de los Tramos", vbOKOnly, "")
      Exit Sub
   End If
   If ElementoC < 2 Then
      Call Crea_Carpeta
      Call GSuper_Proyec
      Call Guardarlos
   Else
      Call Guardarlos
   End If
   CambiaProyB = True: Guardar_CamB = False

End Sub



Private Sub Command4_Click()

If DiseñarB = True Then
   i = MsgBox("Para Imprimir la Última red Debe Guardar el Proyecto", vbOKOnly,
"")
   Exit Sub
```

```
End If
If Guardar_CamB = False Then
   If CambiaProyB = True Then
      Call Reabrir_Proy
      CambiaProyB = False
   End If
   Call Gestiona_Tamaño
   Form5.Visible = True: ImprimirB = True
   'Form1.Visible = False
Else
   i = MsgBox("Para Imprimir los Últimos Cambios Debe Guardar el Proyecto",
vbOKOnly, "")
End If

End Sub

Private Sub Command5_Click()

   If EncuadrarB = False Then
      EncuadrarB = True
   Else
      EncuadrarB = False
   End If
   BAND = 1
   B1A = False

End Sub

Private Sub Command6_Click()

     If Mantener_EncB = True Then
       MousePointer = vbHourglass
       Mantener_EncB = False
       Call Cambia_escala(Form1.Picture1)
       'Call Param_Dib_transf(Form2.Picture1)
       Form1.Picture1.Cls
       Call Dibuja_Pro(Form1.Picture1)
       Call Dibujar_dxf(Form1.Picture1)
       MousePointer = vbDefault
   End If

End Sub
```

```
Private Sub Discmzc_Click()

   Call Borrar
   BAND = 1: DiseñarB = True: VoltajeB = True: CorrB = True:
Form1.Label7.Visible = False
   EncuadrarB = False: Form2.Text1.Visible = False: c = 0
   Mantener_EncB = False
   AbrirlosB = False
   DibujaProB = False
   'ElementoC = ElementoC + 1
   Call Invisible
   Form1.Visible = False
   Form2.Visible = True

End Sub

Private Sub EliminarmProy_Click()

   Call Elimina_Sec

End Sub

Private Sub Estadopro_Click()

If DiseñarB = True Then
   i = MsgBox("Para ver la Última red Debe Guardar el Proyecto", vbOKOnly, "")
   Exit Sub
End If
If Guardar_CamB = False Then
   BAND = 2
   Form1.MousePointer = vbHourglass
   Call Invisible
   Form1.Picture1.Visible = True
   If CambiaProyB = True Then
      Call Reabrir_Proy
      CambiaProyB = False
   Else
      If DibujaProB = False Then
         Form1.MousePointer = vbHourglass
         Mantener_EncB = False
         Call Invisible
         Form1.Picture1.Visible = True
         Form1.Picture1.Cls
```

```
        Call Cambia_escala(Form1.Picture1)
        Call Dibujar_dxf(Form1.Picture1)
        DibujaProB = True
        Form1.MousePointer = vbDefault
      End If
   End If
   Call Dibuja_Pro(Form1.Picture1)
   Form1.MousePointer = vbDefault
 Else
   i = MsgBox("Para Ver Los Cambios de la Red Actual Junto Con Todas Las Demás
es Necesario Que Salve Los Cambios", vbOKOnly, "")
   Exit Sub
End If

End Sub

Private Sub Form_Activate()

   Elemento_Dimensiones Form1, 100, 100, 0, 0
   If ImprimirB = True Then
      ImprimirB = False
   End If
   If N = 0 And Contador = 0 Then
      Form1.Picture1.Visible = False: Call Invisible
      Call Centrar_Pantalla(Form1.Picture1): Form1.Label6.Visible = False:
Form1.Label7.Visible = False
      Form1.Picture1.Visible = True
   End If

End Sub

Private Sub GuadarProyec_Click()

If VoltajeB = True Or CorrB = True Then
   i = MsgBox("Para Guardar la Matriz Actual se Deben Calcular Tanto las Caidas de
Tensión Como las Corrientes de los Tramos", vbOKOnly, "")
   Exit Sub
End If
   If ElementoC < 2 Then
      Call Crea_Carpeta
      Call GSuper_Proyec
      Call Guardarlos
   Else
```

```
      Call Guardarlos
   End If
   CambiaProyB = True: Guardar_CamB = False

End Sub
Private Sub CaidadetensiónT_Click()

   If (N <> 0) Then
      VoltajeB = False
      Form1.MousePointer = vbHourglass
      Call Invisible
      Form1.Picture1.Visible = False
      Form1.Label1.Visible = True
      Form1.MSFlexGrid1.Visible = True
      Form1.Label4.Visible = True
      Form1.Text2.Visible = True
      If AbrirlosB = False Then
         Call Delta_V
      End If
      ReDim CorrR(Aux)
      ReDim CorrI(Aux)
      ReDim Corr(Aux)
      If Corriente(1) <> 0 Then
         Call Corriente_tramos
      End If
      Call BusquedaMayor
      Mayor = CLng(100 * Mayor) / 100
      Form1.Text2.Text = "Voltaje(" & nd & ")=" & Mayor
      Form1.MSFlexGrid1.SetFocus
      Call Manejo_flexgrip
      Form1.Text1.Visible = True
      Form1.Text1.SetFocus
      Form1.Text1.Visible = False
      Form1.MousePointer = vbDefault
   End If

End Sub


Private Sub Imprimir_Click()

If DiseñarB = True Then
```

```
   i = MsgBox("Para Imprimir la Última red Debe Guardar el Proyecto",
vbOKCancel, "")
   Exit Sub
End If
If Guardar_CamB = False Then
   If CambiaProyB = True Then
      Call Reabrir_Proy
      CambiaProyB = False
   End If
   Call Gestiona_Tamaño
   Form5.Visible = True: ImprimirB = True
   'Form1.Visible = False
Else
   i = MsgBox("Para Imprimir los Últimos Cambios Debe Guardar el Proyecto",
vbOKOnly, "")
End If

End Sub
Private Sub ModificarC_Click()

   If N <> 0 Then
      BAND = 6
      Form1.Visible = False: VoltajeB = True: CorrB = True
      Form4.Visible = True
      AbrirlosB = False
   End If

End Sub

Private Sub ModificarCa_Click()

   If N <> 0 Then
      ModificarCB = True: VoltajeB = True: CorrB = True
      Erase Carga
      ReDim Carga(F)
      Form1.Visible = False
      Form3.Visible = True
      AbrirlosB = False
   End If

End Sub

Private Sub ModificarT_Click()
```

```
  If N <> 0 Then
     ModificarTB = True: VoltajeB = True: CorrB = True
     c = N
     fp = 0
     AbrirlosB = False
     Erase NI
     Erase NF
     Erase Longitud
     Erase Tramo
     Erase Resistencia
     Erase Reactancia
     Erase Existente
     Erase Carga
     Erase X
     Erase r
     N = 0
     F = 0
     COND = ""
     BAND = 1
     MousePointer = vbDefault
     Form1.Visible = False
     Form2.Visible = True
   End If

End Sub



Private Sub Opciones_Click()

  If N <> 0 Then

    If Resistencia(N) <> 0 Then
       Form1.CaidadetensiónG.Enabled = True: Form1.Modificar.Enabled = True
       Form1.CaidadetensiónT.Enabled = True: Form1.PorcLim.Enabled = True:
Form1.CaidadetensiónT.Enabled = True
       Form1.Modificar.Enabled = True: Verpro.Enabled = True: Estadopro.Enabled =
True: Form1.CaidadetensiónG.Enabled = True

    End If
  Else
```

```
      Verpro.Enabled = False: Estadopro.Enabled = False: Form1.PorcLim.Enabled =
False: Form1.CaidadetensiónT.Enabled = False: Form1.CaidadetensiónG.Enabled =
False: Form1.Modificar.Enabled = False
  End If
  If DibujaProB = True Then
     Form1.CaidadetensiónG.Enabled = True: Form1.CaidadetensiónG.Enabled =
True
     Form1.CaidadetensiónT.Enabled = True: Form1.PorcLim.Enabled = True:
Form1.CaidadetensiónT.Enabled = True
     Form1.Modificar.Enabled = True: Verpro.Enabled = True: Estadopro.Enabled =
True
  End If

End Sub

Private Sub Archivo_Click()

    If N <> 0 Then
      If Resistencia(N) <> 0 Then
         Imprimir.Enabled = True: GuadarProyec.Enabled = True:
GuadarProyec.Enabled = True: EliminarmProy.Enabled = True: Abrir.Enabled =
True: Discmzc.Enabled = True
      Else
         Imprimir.Enabled = False: GuadarProyec.Enabled = False:
GuadarProyec.Enabled = False: EliminarmProy.Enabled = True: Abrir.Enabled =
True: Discmzc.Enabled = True
      End If
    Else
      Imprimir.Enabled = False: GuadarProyec.Enabled = False:
GuadarProyec.Enabled = False: EliminarmProy.Enabled = False: Abrir.Enabled =
False: Discmzc.Enabled = False
    End If
    If DibujaProB = True Then
      Discmzc.Enabled = True
      If ElementoC >= 2 Then
         Abrir.Enabled = True: Imprimir.Enabled = True
      End If
    End If
    If Contador <> 0 Then
      Abrir.Enabled = True: Imprimir.Enabled = True: Discmzc.Enabled = True:
 EliminarmProy.Enabled = True
    End If
```

```
End Sub

Private Sub Nuevo_Click()

  Call Nuevos
  Numero_Matriz = 0: Guardar_CamB = False

End Sub
Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y
As Single)

   Dim d1 As Double, d2 As Double, d3 As Double
   Select Case BAND
      Case 1
         If Button = 1 Then
            If B1A = True Then
               If EncuadrarB = True Then
                  X2 = X
                  Y2 = Y
                  Call Encuadrar(CDbl(X1), CDbl(Y1), CDbl(X2), CDbl(Y2))
                  Call Nueva_Escala(X1, Y1, X2, Y2)
                  Form1.Picture1.Scale (X1, Y1)-(X2, Y2)
                  Call Param_Dib_transf(Form1.Picture1)
                  EncuadrarB = False
                  Mantener_EncB = True
                  Form1.Picture1.Cls
                  Call Dibuja_Pro(Form1.Picture1)
                  Call Dibujar_dxf(Form1.Picture1)
               End If
            Else:
               B1A = True
               If EncuadrarB = True Then
                  X1 = X
                  Y1 = Y
               End If
            End If
         End If
   End Select

End Sub

Private Sub PorcLim_Click()
```

```
    On Error GoTo quehacer
    Dim Variable1 As String, Variable2 As String
    Variable1 = InputBox("Porcentaje Límite Tope de Bueno= " & Porcentaje1, "")
    Variable2 = InputBox("Porcentaje Límite Tope de Regular= " & Porcentaje2, "")
    Variable1 = Replace(Variable1, ".", ","): Variable2 = Replace(Variable2, ".", ",")
    Porcentaje1 = CSng(Variable1): Porcentaje2 = CSng(Variable2)
quehacer:
    Exit Sub

End Sub

Private Sub Salir_Click()

    End

End Sub

Private Sub Verpro_Click()

    If DiseñarB = True Then
        i = MsgBox("Para ver la Última red Debe Guardar el Proyecto", vbOKOnly, "")
        Exit Sub
    End If
    If Guardar_CamB = False Then
        If CambiaProyB = True Then
            Call Reabrir_Proy
            CambiaProyB = False
        Else
            Form1.MousePointer = vbHourglass
            Mantener_EncB = False
            Call Invisible
            Form1.Picture1.Visible = True
            Form1.Picture1.Cls
            Call Cambia_escala(Form1.Picture1)
            Call Dibujar_dxf(Form1.Picture1)
        End If
        Call Invisible
        Form1.Picture1.Visible = True
        Call Dibuja_Pro(Form1.Picture1)
        DibujaProB = True
        Form1.MousePointer = vbDefault
    Else
```

```
    i = MsgBox("Para Ver Los Cambios de la Red Actual Junto Con Todas Las
Demás es Necesario Que Salve Los Cambios", vbOKOnly, "")
      Exit Sub
   End If

End Sub
```

```
Dim fDibujarOK, B1A As Boolean
Dim Char, Cadena As String, X1 As Single, Y1 As Single, X2 As Single, Y2 As
Single
Private Sub Command1_Click()

   'regresa al menú principal
   Form2.Visible = False
   Form1.Visible = True

End Sub
Private Sub Command2_Click()

      'borra todo o los tramos seleccionados
      Select Case BAND
         Case 1
            Form2.Picture1.Cls
            Form2.Cls
            Erase XPosi
            Erase YPosi
            Erase XPosf
            Erase YPosf
            c = 0
            B1A = False
            xx = -(0.01 * Form2.Picture1.ScaleWidth)
            yy = 0
            xx1 = -(0.025 * Form2.Picture1.ScaleWidth)
            yy1 = 0
         Case Else

            If B1A = True Then
               XPosi(c + 1) = ""
               YPosi(c + 1) = ""
            End If
            i = 0
            While i <= c
               i = i + 1
               If Atriv(i) = True Then
                  If i < c Then
                     For j = i To c - 1
                        XPosi(j) = XPosi(j + 1)
                        YPosi(j) = YPosi(j + 1)
                        XPosf(j) = XPosf(j + 1)
                        YPosf(j) = YPosf(j + 1)
```

```
                    Atriv(j) = Atriv(j + 1)
                Next
              End If
                c = c - 1
            End If

          Wend
           ReDim Preserve XPosi(c)
           ReDim Preserve YPosi(c)
           ReDim Preserve XPosf(c)
           ReDim Preserve YPosf(c)
           ReDim Preserve Atriv(c)
           B1A = False
           BAND = 1
        End Select
        Form2.Picture1.Cls
        Call dibujar
        Call Dibujar_dxf(Form2.Picture1)
        Form2.Picture1.SetFocus

End Sub
Private Sub Command3_Click()
   'selecciona las lineas
   BAND = 2
   Call dibujar
   Call Dibujar_dxf(Form2.Picture1)
   B1A = False
   d3 = (Form2.Picture1.ScaleHeight + Form2.Picture1.ScaleWidth) * 0.5

End Sub
Private Sub Command4_Click()

   'mueve el transformador
   B2A = True
   BAND = 3
   B1A = False
   B5 = True

End Sub
Private Sub Command5_Click()

   'digitalización de la red
```

```
    N = c
    ReDim Tramo(N)
    ReDim NI(N + 1)
    ReDim NF(N + 1)
    ReDim Resistencia(N + 1)
    ReDim Reactancia(N + 1)
    ReDim nodos(N + 1)
    ReDim Longitud(N)
    ReDim Existente(N + 1)
    i = 0
    w = 1
    Call digitalizar
    'Call long_tramo
    If B5 = True Then
       Call Ordenar_esp
       B5 = False
    End If
    Call long_tramo
    i = MsgBox("Ahora Debe Introducir el Tipo de Conductor", vbInformation +
vbOKOnly)
    Form2.Visible = False
    Form4.Visible = True

End Sub
Private Sub Command6_Click()
    'introduce una distancia predeterminada en un tramo
    If BAND = 4 Then
    Else
       Form2.Text1.Text = ""
       Form2.Text1.Visible = True
       Form2.Text1.SetFocus
       BAND = 4
    End If

End Sub
Private Sub Command7_Click()

    If BAND <> 11 Then
       BAND = 11
       Form2.Text1.Visible = True
    Else
       BAND = 1
       'B1A = False
```

```
    B2A = False
    'B5 = False
  End If

End Sub
Private Sub Command8_Click()

  If EncuadrarB = False Then
    EncuadrarB = True
  Else
    EncuadrarB = False
  End If
  BAND = 1
  B1A = False


End Sub
Private Sub Command9_Click()

  If Mantener_EncB = True Then
    MousePointer = vbHourglass
    Mantener_EncB = False
    Call Cambia_escala(Form2.Picture1)
    'Call Param_Dib_transf(Form2.Picture1)
    Form2.Picture1.Cls
    Call dibujar
    Call Dibujar_dxf(Form2.Picture1)
    MousePointer = vbDefault
  End If

End Sub
Private Sub Form_Activate()

  Elemento_Dimensiones Form2, 100, 100, 0, 0
  Call Centrar_Pantalla(Form2.Picture1): Elemento_Dimensiones
Form2.Command1, 5, 5, 3, 6: Elemento_Dimensiones Form2.Command2, 5, 5, 3, 12
  Elemento_Dimensiones Form2.Command3, 5, 5, 3, 18: Elemento_Dimensiones
Form2.Command4, 5, 5, 3, 24: Elemento_Dimensiones Form2.Command5, 5, 5, 3,
30
  Elemento_Dimensiones Form2.Command6, 5, 5, 3, 36: Elemento_Dimensiones
Form2.Command7, 5, 5, 3, 42: Elemento_Dimensiones Form2.Command8, 5, 5, 3,
48
  Elemento_Dimensiones Form2.Command9, 5, 5, 3, 54
```

```
    B1A = False
    B2A = False
    B5 = False
    MousePointer = vbHourglass
    If ModificarTB = False Then
       Erase XPosi
       Erase YPosi
       Erase YPosf
       Erase XPosf
       c = 0
       Form2.Picture1.Cls
       BAND = 1
    End If
    Call Cambia_Y
    If Mantener_EncB = True Then
       If ModificarTB = True Then
          X1 = izquierda1: Y1 = tope1: X2 = ancho1: Y2 = alto1
       End If
       Form2.Picture1.Scale (X1, Y1)-(X2, Y2)
    Else
       Call Cambia_escala(Form2.Picture1)
    End If
    Call Radianes
    Form2.Picture1.Cls
    Call dibujar
    Call Dibujar_dxf(Form2.Picture1)
    pi = 3.141592654
    Call Param_Dib_transf(Form2.Picture1)
    Form1.Visible = False
    MousePointer = vbDefault

End Sub

Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As Single,
Y As Single)

 If Button = vbLeftButton Then
    Select Case BAND
       Case 1
       fDibujarOK = Not (Dibujarok)
       If EncuadrarB = False Then
          c = c + 1
       End If
```

```
    ReDim Preserve XPosi(c)
    ReDim Preserve YPosi(c)
    ReDim Preserve XPosf(c)
    ReDim Preserve YPosf(c)
    ReDim Preserve Atriv(c)
    Case 11
    fDibujarOK = Not (Dibujarok)
    If EncuadrarB = False Then
       c = c + 1
    End If
    ReDim Preserve XPosi(c)
    ReDim Preserve YPosi(c)
    ReDim Preserve XPosf(c)
    ReDim Preserve YPosf(c)
    ReDim Preserve Atriv(c)
    Case Else
       'si no se introducen vectores nuevos en los
       'procedimientos, entonces en este evento no
       'se debe hacer nada
  End Select

End If

End Sub
Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single,
Y As Single)

  If B2A = True Then
    Form2.Picture1.MousePointer = 10
  Else
    Form2.Picture1.MousePointer = 0
  End If
  Select Case BAND
    Case 1
    If EncuadrarB = False Then
      If B1A = True Then

        Form2.Picture1.PSet (XPosi(w), YPosi(w))
        Form2.Text1.Visible = True
        If c = 0 Or 1 Then
          d6 = Sqr((XPosi(w) - X) ^ 2 + (YPosi(w) - Y) ^ 2)
        Else
          d6 = Sqr((XPosf(c) - X) ^ 2 + (YPosf(c) - Y) ^ 2)
```

```
            End If
            d6 = (CLng(d6 * 1000)) / 1000
            Form2.Text1.Text = "d=" & d6
          Else
            Form2.Text1.Visible = False
          End If
        End If
        Case 3
          Form2.Picture1.Cls
          Call dibujar
          Call Dibujar_dxf(Form2.Picture1)
          Form2.Picture1.PSet (X, Y): Form2.Picture1.Circle (X + xx, Y + yy), radio *
0.008, vbBlue: Form2.Picture1.Circle (X + xx1, Y + yy1), radio * 0.008, vbBlue
        Case 4
          Form2.Text1.Visible = True
        Case 11
          Form2.Text1.Text = "X=" & X
          Form2.Text1.Text = Form2.Text1.Text & " Y=" & Y
        Case Else

    End Select

End Sub
Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y
As Single)

    Dim d1 As Double, d2 As Double, d3 As Double
    Select Case BAND
      Case 1
        If Button = 1 Then
          If B1A = True Then
            If EncuadrarB = False Then
              XPosf(c) = X
              YPosf(c) = Y
              Atriv(c) = False
              Call ubicar
            Else
              X2 = X
              Y2 = Y
              Call Encuadrar(CDbl(X1), CDbl(Y1), CDbl(X2), CDbl(Y2))
              Call Nueva_Escala(X1, Y1, X2, Y2)
              Form2.Picture1.Scale (X1, Y1)-(X2, Y2)
              Call Param_Dib_transf(Form2.Picture1)
```

```
                EncuadrarB = False
                Mantener_EncB = True
                Form2.Picture1.Cls
                Call dibujar
                Call Dibujar_dxf(Form2.Picture1)
              End If
              If fDibujarOK Then
                If c = 1 Then
                  Call dibujar
                Else
                  Form2.Picture1.Line (XPosi(c), YPosi(c))-(XPosf(c), YPosf(c)),
QBColor(15)
                  Form2.Picture1.Circle (XPosf(c), YPosf(c)), (radio * 0.0027),
QBColor(15)
                End If
                B1A = False
              End If
            Else:
              B1A = True
              If EncuadrarB = False Then
                XPosi(c) = X
                YPosi(c) = Y
                'Form2.Picture1.PSet (XPosi(c), YPosi(c))
                w = c
                c = c - 1
              Else
                X1 = X
                Y1 = Y
              End If

            End If
          Else
          End If
        Case 3
          If Button = 1 Then
            For j = 1 To c
              d1 = (Sqr((XPosi(j) - X) ^ 2 + (YPosi(j) - Y) ^ 2))
              'd2 = (Sqr((XPosf(i) - X) ^ 2 + (YPosf(i) - Y) ^ 2))
              If d1 <= (0.05 * radio) Then
                d3 = XPosi(j)
                XPosi(j) = XPosi(1)
                XPosi(1) = d3
                d3 = YPosi(j)
```

```
            YPosi(j) = YPosi(1)
            YPosi(1) = d3
            d3 = XPosf(j)
            XPosf(j) = XPosf(1)
            XPosf(1) = d3
            d3 = YPosf(j)
            YPosf(j) = YPosf(1)
            YPosf(1) = d3
            Exit For
        End If
        'If d2 <= (0.05 * radio) Then
            'd3 = XPosi(i)
            'XPosi(i) = XPosi(1)
            'XPosi(1) = d3
            'd3 = YPosi(i)
            'YPosi(i) = YPosi(1)
            'YPosf(1) = d3
            'd3 = XPosf(i)
            'XPosf(i) = XPosf(1)
            'XPosf(1) = d3
            'd3 = YPosf(i)
            'YPosf(i) = YPosf(1)
            'YPosf(1) = d3
            'hacer_reubicacion = True
            'Exit For
        'End If
    Next
  ElseIf Button = 2 Then
    posx = X
    posy = Y
    Call Ubicar_transf
    Form2.Picture1.Cls
    BAND = 1
    B1A = False
    B2A = False
    Call dibujar
    Call Dibujar_dxf(Form2.Picture1)
  End If
Case 2
    aa = X
    cc = Y
    Call seleccionar
    Call dibujar
```

```
            Call Dibujar_dxf(Form2.Picture1)
            B1A = False

        Case 11
          If Button = 1 Then
            If B1A = True Then
              XPosf(c) = X
              YPosf(c) = Y
              Atriv(c) = False
              Call ubicar
              Form2.Picture1.Cls
              If fDibujarOK Then
                Call dibujar
                Call Dibujar_dxf(Form2.Picture1)
                B1A = False
              End If
            Else:
              B1A = True
              XPosi(c) = X
              YPosi(c) = Y
              Form2.Picture1.PSet (XPosi(c), YPosi(c))
              w = c
              c = c - 1

            End If
          Else

      End If
    End Select

End Sub
Private Sub Text1_KeyPress(KeyAscii As Integer)

      Select Case KeyAscii
          Case Asc("0") To Asc("9")
            Cadena = Cadena & Chr$(KeyAscii)
          Case 8
            'No cambiar
          Case 13
            Cadena = Cadena & Chr$(KeyAscii)
          Case Else
            KeyAscii = 0
            Beep
```

```
    End Select
    If BAND = 4 Then
       If KeyAscii = vbKeyReturn Then
          dist = CDbl(Form2.Text1.Text)
          pi = 3.141592654
          Call Forz_cursor
          Form2.Picture1.Cls
          Call dibujar
          Call Dibujar_dxf(Form2.Picture1)
          Form2.Text1.Visible = False
          BAND = 1
       End If
    End If

End Sub
```

```
Dim Cadena, fpo As String
Private Sub situarcuadrotexto()

   Text1.Left = MSFlexGrid1.CellLeft + MSFlexGrid1.Left
   Text1.Top = MSFlexGrid1.CellTop + MSFlexGrid1.Top
   Text1.Width = MSFlexGrid1.CellWidth
   Text1.Height = MSFlexGrid1.CellHeight
   Text1.Visible = True
   Text1.SetFocus

End Sub

Private Sub Command1_Click()

   Dim B1 As Boolean
   If F <> 0 Then
      B1 = False
      If Carga(1) = 0 Then
         Form3.MSFlexGrid1.Col = 2
         For i = 1 To F - 1
            Form3.MSFlexGrid1.Row = i
            If Form3.MSFlexGrid1.Text = "" Then
               B1 = True
            Else
               Carga(i) = CDbl(Form3.MSFlexGrid1.Text)
            End If
         Next

      End If

   End If
   If B1 = True Then
      i = MsgBox("Dejó de Introducir por lo Menos un Dato", vbCritical, "Mensaje")
      Erase Carga
      ReDim Carga(F + 1)
      Form3.Show
      Exit Sub
   End If
   If ModificarTB = False Then
      Call Convertir_Nodos1
   Else
      Call Convertir_Nodos2
      Guardar_CamB = True
```

```
  End If
  BAND = 1
  Form1.Visible = True
  'nuevo codigo
  'ModificarTB = False
  ModificarCB = False
  ModificarUTB = False
  Call Invisible
  If BAND = 1 Or BAND = 6 Or BAND = 7 Or BAND = 11 Then
     BAND = 1
     Form1.Picture1.Visible = True
     Call Centrar_Pantalla(Form1.Picture1)
     Form1.Picture1.Cls
     If N <> 0 Then
        Form1.Command3.Enabled = True: Form1.Command4.Enabled = True
        If COND <> "" Then
           If Mantener_EncB = True Then
              Form1.Picture1.Scale (izquierda1, tope1)-(ancho1, alto1)
           Else
              Form1.Picture1.ScaleHeight = Form2.Picture1.ScaleHeight
              Form1.Picture1.ScaleTop = Form2.Picture1.ScaleTop
              Form1.Picture1.ScaleWidth = Form2.Picture1.ScaleWidth
              Form1.Picture1.ScaleLeft = Form2.Picture1.ScaleLeft
           End If
           Call Dibujar_dxf(Form1.Picture1)
           Call Dibujo_esp(Form1.Picture1)
           Call Numerar_nodos(Form1.Picture1)
        Else
           BAND = 0
           Form1.Picture1.Visible = False
        End If
     Else
        BAND = 0
        Form1.Picture1.Visible = False
     End If
  Else
     Form1.Picture1.Visible = False: Form1.Command3.Enabled = False:
Form1.Command4.Enabled = False
  End If
  Form1.CaidadetensiónG.Enabled = False
  Form1.CaidadetensiónT.Enabled = False
  Form1.Modificar.Enabled = False
  BAND = 0
```

```
    MousePointer = vbDefault
    Form3.Visible = False


End Sub

Private Sub Form_Activate()
  Elemento_Dimensiones Form3, 100, 100, 0, 0
  Dim altura As Single, Ayuda As Long
  altura = ((Form3.Height) * 0.8)
  Elemento_Dimensiones Form3.Command1, 8, 6, 45, 5
  If F <> 0 Then
    Set Text1.Font = MSFlexGrid1.Font
    Form3.MSFlexGrid1.Rows = F + 2
    Form3.MSFlexGrid1.Cols = 5
    Form3.MSFlexGrid1.Rows = F + 1
    Form3.MSFlexGrid1.Cols = 4
    Select Case F
       Case 2, 3
         Form3.MSFlexGrid1.Width = (4 * 1.02 * Form3.MSFlexGrid1.CellWidth)
         Form3.MSFlexGrid1.Height = (1.12 * (F + 1) *
Form3.MSFlexGrid1.CellHeight)
       Case Else
         Form3.MSFlexGrid1.Width = (4 * 1.02 * Form3.MSFlexGrid1.CellWidth)
         If altura > Form3.MSFlexGrid1.Height Then
           Form3.MSFlexGrid1.Height = (1.08 * (F + 1) *
Form3.MSFlexGrid1.CellHeight)
         Else
           Form3.MSFlexGrid1.Height = Form3.Height
         End If
    End Select
    Form3.MSFlexGrid1.Top = ((altura - Form3.MSFlexGrid1.Height) / 2)
    Form3.MSFlexGrid1.Left = ((Form3.Width - Form3.MSFlexGrid1.Width) / 2)
     For i = 1 To 3
       Form3.MSFlexGrid1.Row = 0
       Form3.MSFlexGrid1.Col = i
       Select Case i
         Case 1
             Form3.MSFlexGrid1.Text = "Nodos:"
         Case 2
             Form3.MSFlexGrid1.Text = "Cargas:"

         Case Else
```

```
            Form3.MSFlexGrid1.Text = ""

      End Select
    Next
    If ModificarTB = True Or ModificarCB = True Then
      Ayuda = Referencia(Numero_Matriz)
    Else
      Ayuda = Contador1 + 1
    End If
    For j = 1 To F - 1
      nodos(j) = j
      Form3.MSFlexGrid1.Col = 1
      Form3.MSFlexGrid1.Row = j
      Form3.MSFlexGrid1.Text = nodos(j) + Ayuda
    Next
    If (ModificarCB = True) Or (ModificarTB = True) Or (fp = 0) Then
      For j = 1 To F - 1
        Form3.MSFlexGrid1.Col = 2
        Form3.MSFlexGrid1.Row = j
        Form3.MSFlexGrid1.Text = ""
      Next
      Form3.MSFlexGrid1.Col = 1
      Form3.MSFlexGrid1.Row = F
      Form3.MSFlexGrid1.Text = ""
      Form3.MSFlexGrid1.Col = 2
      Form3.MSFlexGrid1.Row = F
      Form3.MSFlexGrid1.Text = ""
    End If
  Show
    Form3.MSFlexGrid1.Col = 2
    Form3.MSFlexGrid1.Row = 1
    situarcuadrotexto
Else
    Form3.Visible = False
    Form2.Visible = True
End If

End Sub

Private Sub MSFlexGrid1_EnterCell()

If F <> 0 Then
    Text1.Text = MSFlexGrid1.Text
```

```
  situarcuadrotexto
    If Form3.MSFlexGrid1.Row <> F - 1 Then
      If Form3.MSFlexGrid1.Col = 3 Then
        Show
        MSFlexGrid1.Col = 2
        MSFlexGrid1.Row = 1 + CInt(MSFlexGrid1.Row)
      End If
    End If
    If Form3.MSFlexGrid1.Col = 3 And Form3.MSFlexGrid1.Row = F - 1 Then
      Dim SiNo As Integer
      SiNo = MsgBox("Indique ! Si ¡ en caso de no realizar cambios en los
datos,Indique ! No ¡ en caso de desear algun cambio en los datos ", vbYesNo +
vbExclamation, " Mensaje ")
      If SiNo = vbYes Then
       If fp <> 0 Then
        fpo = InputBox("¡ Factor de Potencia de las Cargas ¡,fp=" & fp)
       Else
        fpo = InputBox("¡ Factor de Potencia de las Cargas ¡")
       End If
       Call Revisión_fp(fpo, fp)
       Form3.Command1.SetFocus
      End If
    End If
  End If

End Sub



Private Sub Text1_Change()

  MSFlexGrid1.Text = Text1.Text

End Sub
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)



  If KeyCode = vbKeyDown Then

    MSFlexGrid1.Row = MSFlexGrid1.Row + 1 Mod MSFlexGrid1.Rows

  End If
```

```
  If KeyCode = vbKeyUp Then

     MSFlexGrid1.Row = MSFlexGrid1.Row - 1 Mod MSFlexGrid1.Rows

  End If
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)

  Dim Tamaño%, periodos%
  If KeyAscii = vbKeyReturn Then
    MSFlexGrid1.Col = MSFlexGrid1.Col + 1 Mod MSFlexGrid1.Cols
  End If

  Static PuntoDecimalUtilizado As Integer
  Select Case KeyAscii
   Case Asc("0") To Asc("9")
     Cadena = Cadena & Chr$(KeyAscii)
   Case Asc(".")
     KeyAscii = Asc(",")
     If PuntoDecimalUtilizado Then
        KeyAscii = 0
        KeyAscii = MsgBox("¡ Ya Existe un punto decimal, debe revisar el valor que
cstá colocando !", vbCritical, "Mensaje")
        Else
        PuntoDecimalUtilizado = True
        Cadena = Cadena & Chr$(KeyAscii)
     End If
    Case 8
     'No cambiar
    Case 13
     Cadena = 0
     PuntoDecimalUtilizado = 0

    Case Else
     KeyAscii = 0
   End Select

 End Sub
```

```
Private Sub Command1_Click()

   Form4.Visible = False
   Form1.Visible = True

End Sub

Private Sub Form_Activate()

   Dim B1 As Boolean
   Elemento_Dimensiones Form4, 100, 100, 0, 0
   Elemento_Dimensiones Form4.Frame1, 40, 50, 30, 20: Elemento_Dimensiones
Form4.Option1, 10, 8, 35, 22
   Elemento_Dimensiones Form4.Command1, 10, 6, 45, 5: Elemento_Dimensiones
Form4.Picture1, 12, 17, 5, 7
   Elemento_Dimensiones Form4.Label4, 12, 8, 5, 2
   B1 = False
   If BAND = 1 Then
      Form4.Option1.Value = False
   End If
   Form4.Command1.SetFocus
   Form4.Label4.Visible = True
   x0 = 100
   y0 = Form4.Picture1.Height * 0.25
   With Form4
      .Picture1.BackColor = vbWhite
      .Picture1.Circle (x0 + 500, y0), 40, vbRed
      .Picture1.Circle (x0 + 500, y0 + 400), 40, vbRed
      .Picture1.Circle (x0 + 500, y0 + 800), 40, vbRed
      .Picture1.PSet (x0 + 700, y0)
      .Picture1.Line -(x0 + 700, y0 + 800), vbBlue
      .Picture1.PSet (x0 + 600, y0)
      .Picture1.Line -(x0 + 800, y0), vbBlue
      .Picture1.PSet (x0 + 600, y0 + 400)
      .Picture1.Line -(x0 + 800, y0 + 400), vbBlue
      .Picture1.PSet (x0 + 600, y0 + 800)
      .Picture1.Line -(x0 + 800, y0 + 800), vbBlue
      .Picture1.CurrentX = 900
      .Picture1.CurrentY = 500
      .Picture1.Print "20cm"
      .Picture1.CurrentX = 900
      .Picture1.CurrentY = 900
      .Picture1.Print "20cm"
```

```
  End With
  If BAND = 6 Then

    Form4.Option1.Value = False
    Do
      If Form4.Text1.Text = COND Then
        B1 = True
      Else
        Data1.Recordset.MoveNext
      End If

    Loop Until B1 = True
  Else
    Data1.Recordset.MoveFirst

  End If

End Sub


Private Sub Option1_Click()

  COND = Form4.Text1.Text
  Res = Form4.Text2.Text: Rea = Form4.Text3.Text
  If Res > 1.5 Then
    Res = Replace(Form4.Text2.Text, ",", ".")
  End If
  If Rea > 0.5 Then
    Rea = Replace(Form4.Text3.Text, ",", ".")
  End If
  Rea = CDbl(Rea)
  Res = CDbl(Res)
  Call parametro1
  Call parametro2
  If BAND = 6 Then
    Form4.Visible = False
    Form1.Visible = True
  Else
    Call busqueda
    ReDim Carga(N)
    i = MsgBox("Ahora debe Introducir las Cargas en KVA", vbExclamation, "")
    Form4.Visible = False
```

```
        Form3.Visible = True
    End If

End Sub
```

```
Private Sub Command1_Click()

   If Printer.Orientation = 2 Then
      Printer.Orientation = 1
   End If
   Printer.ColorMode = 2
   Paginas = 0
   If Form5.Check1.Value = 1 Then
      Call Datos_Proy
   End If
   If Form5.Check2.Value = 1 Then
      Call Imprimir_Resul
      If Controlador_dat = False Then
         Call Imprimir_Mayor
      End If
   End If
   If Form5.Check3.Value = 1 Then
      Printer.ColorMode = 1
      Call Imprimir_Dib
   End If
   If Form5.Check1.Value = 1 Or Form5.Check2.Value = 1 Or Form5.Check3.Value
= 1 Then
      Printer.EndDoc
   End If
   Form5.Visible = False

End Sub

Private Sub Command2_Click()

   Form5.Visible = False

End Sub

Private Sub Command3_Click()

   Form5.CommonDialog1.CancelError = True
   On Error GoTo quehacer
   Form5.CommonDialog1.ShowPrinter
quehacer:
   Exit Sub

End Sub
```

```
Private Sub Form_Activate()

   Form5.Command1.Visible = False: Form5.Command2.Visible = False:
Form5.Command3.Visible = False: Form5.SSTab1.Visible = False: Form5.Caption =
""
   If ImprimirB = True Then
      Form5.Command1.Visible = True: Form5.Command2.Visible = True:
Form5.Command3.Visible = True: Form5.SSTab1.Visible = True:
Form5.BorderStyle = 2
      Form5.Caption = "Imprimir"
      Form5.Timer1.Enabled = False
   Else
      Elemento_Dimensiones Form5, 80, 55, 10, 15: Form5.Timer1.Enabled = True
      Form5.Timer1.Interval = 4000
   End If

End Sub
Private Sub Timer1_Timer()
   Form5.Timer1.Enabled = False
   Form1.Visible = True
   Form5.Visible = False

End Sub
```

Public N As Integer, i As Long, j As Integer, k As Integer, l As Integer, p As Integer, q As Integer, w As Integer, cont As Integer, c As Long, F As Integer, pe As Integer, Aux As Integer, nd As Integer
Public r() As Double, X() As Double, Longitud() As Double, Rea As Double, Res As Double, Resistencia() As Double, VoltajeR() As Double, VoltajeI() As Double, Qu() As Double, Voltaje() As Double, Reactancia() As Double
Public E() As Double, Be() As Double, Ge() As Double, CorrienteR() As Double, Corriente1() As Double, CorrienteI() As Double
Public Conductancia() As Double, Suceptancia() As Double, Gauss() As Double, Z() As Double, Carga() As Double, Corriente() As Double, Real() As Double, Imag() As Double
Public NI() As Integer, nodos() As Integer, Elementos() As String, ElementoC As Integer
Public NF() As Integer, Tramo() As Integer, Existente() As Integer
Public B1 As Boolean, B2 As Boolean, B3 As Boolean, B4 As Boolean, B5 As Boolean
Public b As Double, d As Double, m As Double, Gee As Double, G() As Double, Carga1 As Double, Carga2 As Double, s As Double, Bo() As Double, Goo() As Double, Mayor As Double
Public fp As Single, inicio() As Long, Referencia() As Long, ReferenciaC As Integer
Public XPosi() As Double, YPosi() As Double, XPosf() As Double, YPosf() As Double
Public XPosi1() As Double, YPosi1() As Double, XPosf1() As Double, YPosf1() As Double, Voltaje1() As Double, Corr1() As Double, Longitud1() As Double, COND1() As String, fp1() As Single, Perdidas1() As Double, Mayor1() As Double
Public NI1() As Long, NF1() As Long, Tramo1() As Long, Nombre_Pro As String, Nombre As String, Nombre_DXF As String
Public d1 As Double, d2 As Double, d3 As Double, d4 As Double, d5 As Double, d6 As Double, aa As Double, bb As Double, mm As Double, cc As Double, dd As Double, xx As Double, yy As Double
Public Atriv(), B2A As Boolean, AbrirlosB As Boolean, DibujaProB As Boolean, Porcentaje1 As Single, Porcentaje2 As Single
Public BAND As Byte, Aqui As Long, Contador As Long, Contador1 As Long
Public yy1 As Double, xx1 As Double, dist As Double, CorrR() As Double, CorrI() As Double, Corr() As Double
Public posx As Double, posy As Double, radio As Double, Nombre_Matriz As String, Numero_Matriz As Integer
Public Cadena As String, COND As String, Ruta As String, Variable As String, Ruta_DXF As String
Public Xarc() As Double, Yarc() As Double, aarc() As Double, barc() As Double, radioarc() As Double
Public Xcircle() As Double, Ycircle() As Double, radiocircle() As Double

Public Xlinei() As Double, Ylinei() As Double, Xlinef() As Double, Ylinef() As Double
Public Paginas As Integer, karc As Long, kcircle As Long, kline As Long, kpoline As Long
Public ancho1 As Double, alto1 As Double, tope1 As Double, izquierda1 As Double
Public ancho As Double, alto As Double, tope As Double, izquierda As Double
Public mas_negativox As Double, mas_negativoy As Double, mas_positivox As Double, mas_positivoy As Double, menos_positivox As Double, menos_positivoy As Double
Public menos_negativox As Double, menos_negativoy As Double
Public Vector() As String, Perdidas As Single, angB As Boolean, CambiaYB As Boolean, ModificarCB As Boolean, EncuadrarB As Boolean, Mantener_EncB As Boolean
Public Xarc1() As Double, Yarc1() As Double, aarc1() As Double, barc1() As Double, radioarc1() As Double, ModificarTB As Boolean
Public Xcircle1() As Double, Ycircle1() As Double, radiocircle1() As Double
Public Xlinei1() As Double, Ylinei1() As Double, Xlinef1() As Double, Ylinef1() As Double
Public karc1 As Long, kcircle1 As Long, kline1 As Long, kpoline1 As Long, ruta1 As String, Controlador_dat As Boolean, Controlador_res As Boolean
Public Guardar_CamB As Boolean, DiseñarB As Boolean, VoltajeB As Boolean, CorrB As Boolean, ImprimirB As Boolean

```
Sub parametro1()

For i = 1 To N
   Resistencia(i) = (Res * Longitud(i)) / 1000
Next

End Sub
Sub parametro2()

For i = 1 To N
   Reactancia(i) = (Rea * Longitud(i)) / 1000
Next

End Sub


Sub BusquedaMayor()

   For i = 1 To F - 1
      If i = 1 Then
         Mayor = Voltaje(i)
         nd = i
      ElseIf Voltaje(i) > Mayor Then
         Mayor = Voltaje(i)
         If Numero_Matriz > 1 Then
            nd = i + Referencia(Numero_Matriz)
         Else
            nd = i + 1
         End If
      End If
   Next

End Sub

Sub Delta_V()

  ReDim r(N + 1, N + 1)
  ReDim X(N + 1, N + 1)

    i = 0
      For w = 1 To N
         i = i + 1
         B1 = False
```

```
B2 = False
If w = 1 Then
   Call primero
   Call uno
   c = 1
   Existente(c) = NI(w)
   c = c + 1
   Existente(c) = NF(w)
   GoTo 10
End If
For j = 1 To c
      If NI(w) = Existente(j) Then
         B1 = True
      End If
      If NF(w) = Existente(j) Then
         B2 = True
      End If

Next
   If NI(w) = 0 And B2 = False Then
   Call primero
   Call uno
   c = c + 1
   Existente(c) = NF(w)
   End If
   If B1 = True And NF(w) = 0 Then
   Call cuarto
   Call cuatro
   Call Inversa
   Call Multiplicacion
   Call Kron
   i = i - 1
   ElseIf B1 = True And B2 = True Then
   Call tercero
   Call tres
   Call Inversa
   Call Multiplicacion
   Call Kron
   i = i - 1
   End If
   If B1 = True And NI(w) <> 0 Then
   If B2 = False Then
   Call segundo
```

```
            Call dos
            c = c + 1
            Existente(c) = NF(w)
            End If
            End If

10
      Next

   Call busqueda
   ReDim CorrienteR(F)
   ReDim CorrienteI(F)
   ReDim VoltajeR(F)
   ReDim VoltajeI(F)
   ReDim Voltaje(F)
   ReDim Corriente(F)
   ReDim Real(F)
   ReDim Imag(F)
   If fp = 1 Then
    s = 0
   Else
    s = Atn(-fp / Sqr(-fp * fp + 1)) + 2 * Atn(1)
   End If
   For j = 1 To F - 1
      CorrienteR(j) = (Carga(j) / (0.208)) * Cos(-s)
      CorrienteI(j) = (Carga(j) / (0.208)) * Sin(-s)
      Corriente(j) = Sqr(CorrienteR(j) * CorrienteR(j) + CorrienteI(j) * CorrienteI(j))

   Next

   Aux = N
   Call Adicional1

   For i = 1 To F - 1
      Real(i) = 0
      Imag(i) = 0

      For j = 1 To F - 1

        Real(i) = ((208 * ((r(i, j) * Goo(j, 1)) - (X(i, j) * Bo(j, 1)))) + Real(i))
        Imag(i) = ((208 * ((r(i, j) * Bo(j, 1)) + (X(i, j) * Goo(j, 1)))) + Imag(i))
      Next
```

```
 Next
For i = 1 To F - 1
   VoltajeR(i) = 0
   VoltajeI(i) = 0
   For j = 1 To F - 1

    VoltajeR(i) = (((r(i, j) * CorrienteR(j)) - (X(i, j) * CorrienteI(j)))) + (VoltajeR(i))
    VoltajeI(i) = (((r(i, j) * CorrienteI(j)) + (X(i, j) * CorrienteR(j)))) + (VoltajeI(i))

    Next

    VoltajeR(i) = Real(i) - VoltajeR(i)
    VoltajeI(i) = Imag(i) - VoltajeI(i)
    Voltaje(i) = Sqr(VoltajeR(i) ^ 2 + VoltajeI(i) ^ 2)
    Voltaje(i) = (208 - Voltaje(i))
    Voltaje(i) = (Voltaje(i) / 208) * 100
    Voltaje(i) = Redondear(Voltaje(i), 4)
  Next


End Sub
Public Sub Corriente_tramos()
Dim Perdidas2 As Double

If (AbrirlosB = False) Then
   Perdidas = 0
   VoltajeR(NI(1)) = 208
   VoltajeI(NF(1)) = 0
   Call Transformacion1
   For i = 1 To Aux

      CorrR(i) = ((VoltajeR(NF(i)) - VoltajeR(NI(i))) * Conductancia(i)) -
((VoltajeI(NF(i)) - VoltajeI(NI(i))) * Suceptancia(i))
      CorrI(i) = ((VoltajeR(NF(i)) - VoltajeR(NI(i))) * Suceptancia(i)) +
((VoltajeI(NF(i)) - VoltajeI(NI(i))) * Conductancia(i))
      Corr(i) = Sqr((CorrR(i) ^ 2) + (CorrI(i) ^ 2))
      Corr(i) = Redondear(Corr(i), 4)

   Next
End If
      Call Perdidas_Red
      Perdidas = (CLng(100 * Perdidas)) / 100
```

```
    Perdidas2 = Perdidas
    If Perdidas >= 1000 Then
       Perdidas2 = (Perdidas2 / 1000)
       Form1.Label5.Caption = "Perdida de la Red(kW):"
    Else
       Form1.Label5.Caption = "Perdida de la Red(W):"
    End If
    Perdidas2 = (CLng(100 * Perdidas2)) / 100
    Form1.Text3.Visible = True
    Form1.Label5.Visible = True
    Form1.Text3.Text = Perdidas2
    Corr3 = False

End Sub
Sub Transformacion()

For i = 1 To N
   Conductancia(i) = CDbl(Resistencia(i) / ((Resistencia(i) * Resistencia(i)) +
(Reactancia(i) * Reactancia(i))))
   Suceptancia(i) = CDbl(-Reactancia(i) / ((Resistencia(i) * Resistencia(i)) +
(Reactancia(i) * Reactancia(i))))
Next

End Sub
Sub Transformacion1()
   ReDim Conductancia(Aux)
   ReDim Suceptancia(Aux)
   For i = 1 To N
      Conductancia(i) = Resistencia(i) / ((Resistencia(i) * Resistencia(i)) +
(Reactancia(i) * Reactancia(i)))
      Suceptancia(i) = -Reactancia(i) / ((Resistencia(i) * Resistencia(i)) +
(Reactancia(i) * Reactancia(i)))
   Next

End Sub
Sub Adicional1()
   ReDim Goo(F, 1)
   ReDim Bo(F, 1)
   For i = 1 To F - 1
      Goo(i, 1) = 0
      Bo(i, 1) = 0
         For j = 1 To Aux
            If NI(j) = nodos(i + 1) Or NF(j) = nodos(i + 1) Then
```

```
            If NI(j) = 0 Or NF(j) = 0 Then
               Resistencia(j) = (CLng(10000 * Resistencia(j))) / 10000
               Reactancia(j) = (CLng(10000 * Reactancia(j))) / 10000
               Goo(i, 1) = (Resistencia(j)) / (Resistencia(j) ^ 2 + Reactancia(j) ^ 2)
               Bo(i, 1) = (-Reactancia(j)) / (Resistencia(j) ^ 2 + Reactancia(j) ^ 2)
            End If
         End If
      Next

   Next

End Sub
Public Sub Invisible()

   Form1.Label1.Visible = False
   Form1.MSFlexGrid1.Visible = False
   Form1.Text1.Visible = False
   Form1.Text2.Visible = False: Form1.Text3.Visible = False
   Form1.Label4.Visible = False
   Form1.Label5.Visible = False

End Sub

Public Function Redondear(X As Variant, ene As Integer) As Double

   Redondear = (CLng((10 ^ ene) * X)) / (10 ^ ene)

End Function

Public Sub Perdidas_Red()

   For i = 1 To Aux
      Perdidas = Perdidas + CSng((Corr(i) ^ 2) * Resistencia(i))
   Next

End Sub
```

```
Sub uno()
  'Elemento entre referencia y un nodo nuevo
   For p = 1 To i
     For q = 1 To i
        If p = i And q = i Then
           r(p, q) = Resistencia(w)
        End If
        If p = i And q <> i Then
           r(p, q) = 0
        End If
        If p <> i And q = i Then
           r(p, q) = 0
        End If
     Next
   Next
End Sub

Sub dos()
  'Elemento entre un nodo existente y  un nodo nuevo

   u = i - 1 'asigna a la variable u el valor del nodo inicial
   For p = 1 To i - 1
      r(p, i) = r(p, u)
   Next
   For q = 1 To i - 1
      r(i, q) = r(u, q)
   Next
   r(i, i) = CDbl(Resistencia(w)) + CDbl(r(u, u))

End Sub

Sub tres()

   'elememtos entre dos nodos existentes
   u = NI(w)
   V = NF(w)
   For t = 1 To i - 1
    r(t, i) = r(t, u) - r(t, V)
   Next
   For t = 1 To i - 1
    r(i, t) = r(u, t) - r(V, t)
   Next
    r(i, i) = CDbl(r(u, u)) + CDbl(r(V, V)) - 2 * CDbl(r(u, V)) + CDbl(Resistencia(w))
```

```
    'luego se aplica Kron

End Sub

Sub cuatro()

    'Elemento entre un nodo existente y  nodo de referencia
    u = NI(w) 'asigna a la variable u el valor del nodo inicial
    For p = 1 To i - 1
        r(p, i) = r(p, u)
    Next
    For q = 1 To i - 1
        r(i, q) = r(u, q)
    Next
    r(i, i) = Resistencia(w) + r(u, u)
    'luego se aplica Kron
End Sub

Sub primero()
    'Elemento entre referencia y un nodo nuevo
    For p = 1 To i
        For q = 1 To i
            If p = i And q = i Then
                X(p, q) = Reactancia(w)
            End If
            If p = i And q <> i Then
                X(p, q) = 0
            End If
            If p <> i And q = i Then
                X(p, q) = 0
            End If
        Next
    Next
End Sub

Sub segundo()
    'Elemento entre un nodo existente y  un nodo nuevo
    u = i - 1 'asigna a la variable u el valor del nodo inicial
    For p = 1 To i - 1
        X(p, i) = X(p, u)
    Next
    For q = 1 To i - 1
```

```
      X(i, q) = X(u, q)
    Next
    X(i, i) = Reactancia(w) + X(u, u)

End Sub

Sub tercero()

  'elememtos entre dos nodos existentes
  u = NI(w)
  V = NF(w)
  For t = 1 To i - 1
   X(t, i) = X(t, u) - X(t, V)
  Next
  For t = 1 To i - 1
   X(i, t) = X(u, t) - X(V, t)
  Next
   X(i, i) = CDbl(X(u, u)) + CDbl(X(V, V)) - 2 * CDbl(X(u, V)) +
CDbl(Reactancia(w))
   'luego se aplica Kron
End Sub
Sub Inversa()

   b = (r(i, i)) / (r(i, i) ^ 2 + X(i, i) ^ 2)
   d = (-X(i, i)) / (r(i, i) ^ 2 + X(i, i) ^ 2)

End Sub
Sub Multiplicacion()
ReDim E(i, i)
ReDim G(i, i)
For p = 1 To i - 1
   For q = 1 To i - 1
      E(p, q) = r(p, i) * (b * r(i, q) - d * X(i, q)) - X(p, i) * (d * r(i, q) + b * X(i, q))
      G(p, q) = X(p, i) * (b * r(i, q) - d * X(i, q)) + r(p, i) * (d * r(i, q) + b * X(i, q))
   Next
Next

End Sub

Sub Kron()
      'Reducción de Kron
      For p = 1 To i - 1
         For q = 1 To i - 1
```

```
            r(p, q) = r(p, q) - E(p, q)
            X(p, q) = X(p, q) - G(p, q)
            Next
        Next

End Sub

Sub cuarto()

    'Elemento entre un nodo existente y  el nodo de referencia
    u = NI(w) 'asigna a la variable u el valor del nodo inicial
    For p = 1 To i - 1
        X(p, i) = X(p, u)
    Next
    For q = 1 To i - 1
        X(i, q) = X(u, q)
    Next
    X(i, i) = Reactancia(w) + X(u, u)
    'luego se aplica Kron
End Sub
```

```
Sub busqueda()

   For i = 1 To N
   B1 = False
   B2 = False
   If i = 1 Then
      nodos(i) = NI(i)
      nodos(i + 1) = NF(i)
      w = 2
   Else
      For k = 1 To w
         If NI(i) = nodos(k) Then
            B1 = True
         End If
         If NF(i) = nodos(k) Then
            B2 = True
         End If
      Next
      If B1 = False Then
         If B2 = False Then
            w = w + 1
            nodos(w) = NF(i)
            w = w + 1
            nodos(w) = NI(i)
          Else
            w = w + 1
            nodos(w) = NI(i)
         End If
       ElseIf B2 = False Then
         w = w + 1
         nodos(w) = NF(i)
    End If
   End If
 F = w
 Next

End Sub

Sub ubicar()

   For i = c To 1 Step -1
      For j = 1 To c
         If i <> j Then
```

```
            d1 = Sqr((XPosi(i) - XPosi(j)) ^ 2 + (YPosi(i) - YPosi(j)) ^ 2)
            d2 = Sqr((XPosf(i) - XPosf(j)) ^ 2 + (YPosf(i) - YPosf(j)) ^ 2)
            d3 = Sqr((XPosf(i) - XPosi(j)) ^ 2 + (YPosf(i) - YPosi(j)) ^ 2)
            d4 = Sqr((XPosi(i) - XPosf(j)) ^ 2 + (YPosi(i) - YPosf(j)) ^ 2)
          If d1 <= (0.025 * Form2.Picture1.ScaleWidth) Then
            XPosi(i) = XPosi(j)
            YPosi(i) = YPosi(j)

          End If
          If d2 <= (0.025 * Form2.Picture1.ScaleWidth) Then
            XPosf(i) = XPosf(j)
            YPosf(i) = YPosf(j)

          End If
          If d3 <= (0.025 * Form2.Picture1.ScaleWidth) Then
            XPosf(i) = XPosi(j)
            YPosf(i) = YPosi(j)

          End If
          If d4 <= (0.025 * Form2.Picture1.ScaleWidth) Then
            XPosi(i) = XPosf(j)
            YPosi(i) = YPosf(j)

          End If
        End If
        Next
    Next

End Sub
Sub dibujar()

    Dim a As Integer
    If F <> 0 Then
      a = N
    Else
      a = c
    End If
    For i = 1 To a

          If i = 1 Then
            If BAND = 3 Then
              Form2.Picture1.PSet (XPosi(i), YPosi(i))
            Else
```

```
            Form2.Picture1.PSet (XPosi(i), YPosi(i)): Form2.Picture1.Circle
(XPosi(i) + xx, YPosi(i) + yy), radio * 0.008, QBColor(9): Form2.Picture1.Circle
(XPosi(i) + xx1, YPosi(i) + yy1), radio * 0.008, QBColor(9)
          End If
        Else
          Form2.Picture1.PSet (XPosi(i), YPosi(i))
        End If

        If Atriv(i) = False Then
          Form2.Picture1.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)),
QBColor(15): Form2.Picture1.Circle (XPosf(i), YPosf(i)), radio * 0.0027,
QBColor(15)
        Else
          Form2.Picture1.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)), vbCyan:
Form2.Picture1.Circle (XPosf(i), YPosf(i)), radio * 0.0027, vbCyan
        End If
    Next

End Sub
Sub seleccionar()

    For i = 1 To c
      bb = Sqr((aa - XPosi(i)) ^ 2 + (cc - YPosi(i)) ^ 2)
      If XPosi(i) = XPosf(i) Then
       mm = pi / 2
      Else
       mm = (YPosi(i) - YPosf(i)) / (XPosi(i) - XPosf(i))
       mm = Atn(mm)
      End If
      If aa = XPosi(i) Then
       dd = pi / 2
      Else
       dd = (cc - YPosi(i)) / (aa - XPosi(i))
       dd = Atn(dd)
      End If
      d1 = mm - dd
      d2 = bb * (Sin(d1))
      If Abs(d2) < (0.0021 * d3) Then
       Atriv(i) = True
      End If
    Next

End Sub
```

```
Sub Ubicar_transf()


   pi = 3.141592654
   If XPosi(1) = posx Then
      mm = pi / 2
   Else
      mm = (posy - YPosi(1)) / (XPosi(1) - posx)
      mm = Atn(mm)
   End If
   If mm > (pi / 2) Then
      mm = pi - mm
   End If
   aa = Sqr(xx ^ 2 + yy ^ 2)
   bb = Sqr(xx1 ^ 2 + yy1 ^ 2)
   xx = aa * Cos(mm)
   yy = aa * Sin(mm)
   xx1 = bb * Cos(mm)
   yy1 = bb * Sin(mm)
   posx = posx - XPosi(1)
   posy = YPosi(1) - posy
   If (posx > 0) And (posy > 0) Then
      yy = yy * -1
      yy1 = yy1 * -1
   End If
   If (posx < 0) And (posy > 0) Then
     xx = xx * -1
     xx1 = xx1 * -1
   End If
   If (posx < 0) And (posy < 0) Then
      xx = xx * -1
      xx1 = xx1 * -1
   End If
   If (posx > 0) And (posy < 0) Then
      yy = yy * -1
      yy1 = yy1 * -1
   End If

End Sub
Public Sub Manejo_flexgrip()

   Dim altura, anchura As Single
```

```
  altura = Form1.Height * 0.5
  anchura = Form1.Width * 0.15
  Form1.MSFlexGrid1.Rows = F + 2 'total de filas
  Form1.MSFlexGrid1.Cols = 4 'total de columnas
  For i = 0 To 3
     Form1.MSFlexGrid1.Col = i
     For j = 1 To F + 1
        Form1.MSFlexGrid1.Row = j
        Form1.MSFlexGrid1.Text = ""
     Next
   Next
  Form1.MSFlexGrid1.Cols = 3 'total de filas
  Form1.MSFlexGrid1.Rows = F + 1 'total de columnas
   Select Case F
     Case 1 To 2
        Form1.MSFlexGrid1.Height = ((F + 1) * 1.03 *
Form1.MSFlexGrid1.CellHeight)
        Form1.MSFlexGrid1.Width = (1.1 * 3 * Form1.MSFlexGrid1.CellWidth)
     Case 3 To 29
        Form1.MSFlexGrid1.Height = ((F + 1) * 1.08 *
Form1.MSFlexGrid1.CellHeight)
        Form1.MSFlexGrid1.Width = (1.03 * 3 * Form1.MSFlexGrid1.CellWidth)
     Case Else
        Form1.MSFlexGrid1.Height = (0.9 * altura)
  End Select

     For i = 0 To 2
        Form1.MSFlexGrid1.Col = i
        For j = 0 To F - 1
           Form1.MSFlexGrid1.Row = j
           Select Case i
              Case 0
                 Select Case j
                    Case 0
                       Form1.MSFlexGrid1.Text = "Nodo:"
                    Case Else
                       Form1.MSFlexGrid1.Text = (j + Referencia(Numero_Matriz))
                 End Select
              Case 1
                 Select Case j
                    Case 0
                       Form1.MSFlexGrid1.Text = "Voltaje(%):"
                    Case Else
```

```
                    Form1.MSFlexGrid1.Text = Voltaje(j)
                End Select
            Case 2
                Form1.MSFlexGrid1.Text = ""
            End Select
        Next
    Next

    Call Centrar_3
    Elemento_Dimensiones Form1.Label1, 28, 10, 37, 24: Elemento_Dimensiones
Form1.Label4, 20, 5, 65, 10: Elemento_Dimensiones Form1.Text2, 10, 3, 84, 10
    Elemento_Dimensiones Form1.Label5, 28, 5, 65, 15: Elemento_Dimensiones
Form1.Text3, 10, 3, 84, 15

End Sub

Private Sub Centrar_3()

    Form1.MSFlexGrid1.Left = ((Form1.Width - Form1.MSFlexGrid1.Width) / 2)

End Sub

Public Sub Centrar_Pantalla(Objeto As Object)

    Objeto.Width = (Screen.Width) * 0.8
    Objeto.Height = (Screen.Height) * 0.65
    Objeto.Left = (Screen.Width) * 0.125
    Objeto.Top = (Screen.Height) * 0.1

End Sub

Public Sub Param_Dib_transf(Objeto As Object)

    xx = -(0.01 * Objeto.ScaleWidth)
    yy = 0
    xx1 = -(0.025 * Objeto.ScaleWidth)
    yy1 = 0
    radio = (Objeto.ScaleWidth)

End Sub
Public Sub Nueva_Escala(X1 As Single, Y1 As Single, X2 As Single, Y2 As Single)

    tope1 = Y1
```

```
  izquierda1 = X1
  ancho1 = X2
  alto1 = Y2

End Sub

Public Sub Dibuja_Pro(Objeto As Object)

      Dim Mayor As Double, Bandera As Boolean, j As Long, I As Long
      Bandera = False
      If Mantener_EncB = False Then
         yy = 0: yy1 = 0: xx = -(0.001 * Objeto.ScaleWidth): xx1 = -(0.0025 *
Objeto.ScaleWidth): radio = 0.1 * (Objeto.ScaleWidth)
      End If
      For j = 1 To Contador
        For I = 1 To ReferenciaC
          If Referencia(I) = NI1(j) Then
            Bandera = True
            Exit For
          End If
        Next
         If Bandera = True Then
          Objeto.PSet (XPosi1(j), YPosi1(j)): Objeto.Circle (XPosi1(j) + xx,
YPosi1(j) + yy), radio * 0.008, QBColor(9): Objeto.Circle (XPosi1(j) + xx1,
YPosi1(j) + yy1), radio * 0.008, QBColor(9): Bandera = False
         Else
          Objeto.PSet (XPosi1(j), YPosi1(j))
         End If

         If BAND = 0 Or BAND = 1 Then
          Objeto.Line (XPosi1(j), YPosi1(j))-(XPosf1(j), YPosf1(j)), QBColor(15):
Objeto.Circle (XPosf1(j), YPosf1(j)), radio * 0.0027, QBColor(15)
         Else
          If Voltaje1(NI1(j)) > Voltaje1(NF1(j)) Then
            Mayor = Voltaje1(NI1(j))
           Else
            Mayor = Voltaje1(NF1(j))
          End If
          If Mayor < 3 Then
            BAND = 8
          ElseIf Mayor > 5 Then
            BAND = 10
          Else
```

```
            BAND = 9
          End If
          Select Case BAND
            Case 8
              Objeto.Line (XPosi1(j), YPosi1(j))-(XPosf1(j), YPosf1(j)),
QBColor(10): Objeto.Circle (XPosf1(j), YPosf1(j)), radio * 0.0027, QBColor(10)
            Case 9
              Objeto.Line (XPosi1(j), YPosi1(j))-(XPosf1(j), YPosf1(j)),
QBColor(14): Objeto.Circle (XPosf1(j), YPosf1(j)), radio * 0.0027, QBColor(14)
            Case 10
              Objeto.Line (XPosi1(j), YPosi1(j))-(XPosf1(j), YPosf1(j)),
QBColor(12): Objeto.Circle (XPosf1(j), YPosf1(j)), radio * 0.0027, QBColor(12)
          End Select
        End If
    Next
  BAND = 1

End Sub

Public Sub Elemento_Dimensiones(Objeto As Object, Porcentaje1 As Integer,
Porcentaje2 As Integer, Porcentaje3 As Integer, Porcentaje4 As Integer)

  'dimencsiones
  Objeto.Width = (Screen.Width) * (Porcentaje1 / 100): Objeto.Height =
(Screen.Height) * (Porcentaje2 / 100)
  'ubicación
  Objeto.Left = (Screen.Width) * (Porcentaje3 / 100): Objeto.Top = (Screen.Width)
* (Porcentaje4 / 100)

End Sub
```

```
Sub digitalizar()

  Do
    i = i + 1
    Tramo(i) = i
    If i = 1 Then
      NI(i) = 0
      NF(i) = 1
    Else
      B1 = False
      B2 = False
      B3 = False
      B4 = False
      For j = i - 1 To 1 Step -1
        If (XPosi(i) = XPosi(j)) And (YPosi(i) = YPosi(j)) Then
          B1 = True
          k = NI(j)
          Exit For
        End If
      Next
      For j = i - 1 To 1 Step -1
        If (XPosf(i) = XPosf(j)) And (YPosf(i) = YPosf(j)) Then
          B2 = True
          l = NF(j)
          Exit For
        End If
      Next
      For j = i - 1 To 1 Step -1
        If (XPosf(i) = XPosi(j)) And (YPosf(i) = YPosi(j)) Then
          B3 = True
          p = NI(j)
          Exit For
        End If
      Next
      For j = i - 1 To 1 Step -1
        If (XPosi(i) = XPosf(j)) And (YPosi(i) = YPosf(j)) Then
          B4 = True
          q = NF(j)
          Exit For
        End If
      Next
        If B1 = True Then
          NI(i) = k
```

```
            End If
            If B2 = True Then
               NF(i) = l
            End If
            If B3 = True Then
               NF(i) = p
            End If
            If B4 = True Then
               NI(i) = q
            End If
            If B1 = False And B4 = False Then
               w = w + 1
               NI(i) = w
               If B2 = False And B3 = False Then
                  w = w + 1
                  NF(i) = w
               End If
            ElseIf B2 = False And B3 = False Then
                  w = w + 1
                  NF(i) = w
            End If
         End If
   Loop Until i = N

End Sub

Sub long_tramo()

   For i = 1 To N
      Longitud(i) = (Sqr((XPosi(i) - XPosf(i)) ^ 2 + (YPosi(i) - YPosf(i)) ^ 2))
   Next

End Sub

Sub Forz_cursor()

   pi = 3.141592654
   If XPosi(c) = XPosf(c) Then
      mm = pi / 2
   Else
      mm = (YPosi(c) - YPosf(c)) / (XPosi(c) - XPosf(c))
      mm = Atn(mm)
   End If
```

```
    If XPosf(c) < XPosi(c) Then
       mm = pi + mm
    End If
    d = Sqr((XPosi(c) - XPosf(c)) ^ 2 + (YPosi(c) - YPosf(c)) ^ 2)
    cc = (d - dist) * Cos(mm)
    dd = (d - dist) * Sin(mm)
    XPosf(c) = XPosf(c) - cc
    YPosf(c) = YPosf(c) - dd

End Sub

Sub Dibujo_esp(Objeto As Object)

    Dim Mayor As Double

       For i = 1 To N
          If i = 1 Then
            Objeto.PSet (XPosi(i), YPosi(i)): Objeto.Circle (XPosi(i) + xx, YPosi(i) +
yy), radio * 0.008, QBColor(9): Objeto.Circle (XPosi(i) + xx1, YPosi(i) + yy1), radio
* 0.008, QBColor(9)
          Else
            Objeto.PSet (XPosi(i), YPosi(i))
          End If

          If BAND = 0 Or BAND = 1 Then
            Objeto.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)), QBColor(15):
Objeto.Circle (XPosf(i), YPosf(i)), radio * 0.0027, QBColor(15)
          Else
             Voltaje(0) = 0
            For j = 1 To F
              If NI(i) = nodos(j) Then
                 p = j - 1
                 Exit For
              End If
            Next
            For j = 1 To F
              If NF(i) = nodos(j) Then
                 q = j - 1
                 Exit For
              End If
            Next

            If Voltaje(p) > Voltaje(q) Then
```

```
            Mayor = Voltaje(p)
          Else
            Mayor = Voltaje(q)
        End If
        If Mayor < Porcentaje1 Then
          BAND = 8
        ElseIf Mayor > Porcentaje2 Then
          BAND = 10
        Else
          BAND = 9
        End If
        Select Case BAND
          Case 8
            Objeto.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)), QBColor(10):
Objeto.Circle (XPosf(i), YPosf(i)), radio * 0.0027, QBColor(10)
          Case 9
            Objeto.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)), QBColor(14):
Objeto.Circle (XPosf(i), YPosf(i)), radio * 0.0027, QBColor(14)
          Case 10
            Objeto.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)), QBColor(12):
Objeto.Circle (XPosf(i), YPosf(i)), radio * 0.0027, QBColor(12)
        End Select
      End If
    Next


End Sub
Sub Numerar_nodos(Objeto As Object)

  For i = (Tramo1(Numero_Matriz)) To ((Tramo1(Numero_Matriz) + N) - 1)
    Objeto.PSet (XPosi1(i), YPosi1(i) + (0.005 * Objeto.ScaleWidth))
    Objeto.Print " " & NI1(i)
    Objeto.PSet (XPosf1(i), YPosf1(i) + (0.005 * Objeto.ScaleWidth))
    Objeto.Print " " & NF1(i)
  Next

End Sub



Public Sub Ordenar_esp()
```

```
    Dim d2, p, q As Double
    Dim B1, B2 As Boolean
    i = 0
    Do
       i = i + 1
       If i > 1 Then
          B1 = False
          B2 = False
          For j = i - 1 To 1 Step -1
             If (NI(i) = NI(j)) Or (NI(i) = NF(j)) Then
                B1 = True
                GoTo 10
             End If
10
             If (NF(i) = NI(j)) Or (NF(i) = NF(j)) Then
                B2 = True
                If B1 = True Then
                   Exit For
                End If
             End If
          Next
          If (B1 = False) And (B2 = True) Then
             d2 = NI(i)
             NI(i) = NF(i)
             NF(i) = d2
             d2 = XPosi(i)
             XPosi(i) = XPosf(i)
             XPosf(i) = d2
             d2 = YPosi(i)
             YPosi(i) = YPosf(i)
             YPosf(i) = d2
          ElseIf (B1 = False) And (B2 = False) Then
             For j = i + 1 To N
                For k = i - 1 To 1 Step -1
                   If (NI(j) = NI(k)) Or (NI(j) = NF(k)) Then
                      B1 = True
                      Exit For
                   End If
                   If (NF(j) = NI(k)) Or (NF(j) = NF(k)) Then
                      B2 = True
                      Exit For
                   End If
                Next
```

```
      If B1 = True Or B2 = True Then
        Exit For
      End If
    Next
      If B1 = True And B2 = False Then
        d2 = NI(i)
        NI(i) = NI(j)
        NI(j) = d2
        d2 = NF(i)
        NF(i) = NF(j)
        NF(j) = d2
        d2 = XPosi(i)
        XPosi(i) = XPosi(j)
        XPosi(j) = d2
        d2 = YPosi(i)
        YPosi(i) = YPosi(j)
        YPosi(j) = d2
        d2 = XPosf(i)
        XPosf(i) = XPosf(j)
        XPosf(j) = d2
        d2 = YPosf(i)
        YPosf(i) = YPosf(j)
        YPosf(j) = d2
      ElseIf (B1 = False) And (B2 = True) Then
        d2 = NI(j)
        NI(j) = NF(j)
        NF(j) = d2
        d2 = XPosi(j)
        XPosi(j) = XPosf(j)
        XPosf(j) = d2
        d2 = YPosi(j)
        YPosi(j) = YPosf(j)
        YPosf(j) = d2
        d2 = NI(i)
        NI(i) = NI(j)
        NI(j) = d2
        d2 = NF(i)
        NF(i) = NF(j)
        NF(j) = d2
        d2 = XPosi(i)
        XPosi(i) = XPosi(j)
        XPosi(j) = d2
        d2 = YPosi(i)
```

```
                YPosi(i) = YPosi(j)
                YPosi(j) = d2
                d2 = XPosf(i)
                XPosf(i) = XPosf(j)
                XPosf(j) = d2
                d2 = YPosf(i)
                YPosf(i) = YPosf(j)
                YPosf(j) = d2

            End If

        End If
    End If

  Loop Until i = N

End Sub
Public Sub Encuadrar(X1 As Double, Y1 As Double, X2 As Double, Y2 As Double)

  Dim Es_Correcto1 As Boolean, Es_Correcto2 As Boolean, m As Double, b As
Double, Sol As Double
  For i = 1 To kline
    Es_Correcto1 = False
    Es_Correcto2 = False
    If i = 1 Then
      kline1 = 0
    End If
    If (Dominio(Xlinei(i), X1, X2) = True) And (Dominio(Ylinei(i), Y1, Y2) =
True) Then
      Es_Correcto1 = True
      GoTo 100
    End If
    If (Dominio(Xlinef(i), X1, X2) = True) And (Dominio(Ylinef(i), Y1, Y2) =
True) Then
      Es_Correcto1 = True
      GoTo 100
    End If
    If (Dominio(Xlinef(i), X1, X2) = True) And (Dominio(Ylinei(i), Y1, Y2) =
True) Then
      Es_Correcto1 = True
      GoTo 100
    End If
```

```
    If (Dominio(Xlinei(i), X1, X2) = True) And (Dominio(Ylinef(i), Y1, Y2) =
True) Then
        Es_Correcto1 = True
        GoTo 100
    End If
    If (Dominio(X1, Xlinei(i), Xlinef(i)) = True) And (Dominio(Y1, Ylinei(i),
Ylinef(i)) = True) Then
        Es_Correcto1 = True
        GoTo 100
    End If
    If (Dominio(X2, Xlinei(i), Xlinef(i)) = True) And (Dominio(Y2, Ylinei(i),
Ylinef(i)) = True) Then
        Es_Correcto1 = True
        GoTo 100
    End If
    If (Dominio(X1, Xlinei(i), Xlinef(i)) = True) And (Dominio(Y2, Ylinei(i),
Ylinef(i)) = True) Then
        Es_Correcto1 = True
        GoTo 100
    End If
    If (Dominio(X2, Xlinei(i), Xlinef(i)) = True) And (Dominio(Y1, Ylinei(i),
Ylinef(i)) = True) Then
        Es_Correcto1 = True
        GoTo 100
    End If
100
    If Xlinef(i) <> Xlinei(i) Then
        m = (Ylinef(i) - Ylinei(i)) / (Xlinef(i) - Xlinei(i))
        b = (Ylinei(i) * Xlinef(i) - Ylinef(i) * Xlinei(i)) / (Xlinef(i) - Xlinei(i))
        If m <> 0 Then
            'paso I
            Sol = (Y1 - b) / m
            If Dominio(Sol, X1, X2) = True Then
                Es_Correcto2 = True
                GoTo 101
            End If
            'paso II
            Sol = X2 * m + b
            If Dominio(Sol, Y1, Y2) = True Then
                Es_Correcto2 = True
                GoTo 101
            End If
            'paso III
```

```
            Sol = (Y2 - b) / m
            If Dominio(Sol, X1, X2) = True Then
               Es_Correcto2 = True
               GoTo 101
            End If
            'paso IV
            Sol = X1 * m + b
            If Dominio(Sol, Y1, Y2) = True Then
               Es_Correcto2 = True
               GoTo 101
            End If
         ElseIf Dominio(Ylinei(i), Y1, Y2) Then
            Es_Correcto2 = True
            GoTo 101
         End If
      ElseIf Dominio(Xlinei(i), X1, X2) = True Then
         Es_Correcto2 = True
         GoTo 101
      End If
101
      If (Es_Correcto1 = True) And (Es_Correcto2 = True) Then
         kline1 = kline1 + 1
         ReDim Preserve Xlinei1(kline1)
         ReDim Preserve Xlinef1(kline1)
         ReDim Preserve Ylinei1(kline1)
         ReDim Preserve Ylinef1(kline1)
         Xlinei1(kline1) = Xlinei(i)
         Xlinef1(kline1) = Xlinef(i)
         Ylinei1(kline1) = Ylinei(i)
         Ylinef1(kline1) = Ylinef(i)
      End If
   Next
   For i = 1 To kcircle
      Es_Correcto1 = False
      If i = 1 Then
         kcircle1 = 0
      End If
      If (Dominio(Xcircle(i) + radiocircle(i), X1, X2) = True) And
(Dominio(Ycircle(i) + radiocircle(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
```

```
      If (Dominio(Xcircle(i) - radiocircle(i), X1, X2) = True) And
(Dominio(Ycircle(i) - radiocircle(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(Xcircle(i) + radiocircle(i), X1, X2) = True) And
(Dominio(Ycircle(i) - radiocircle(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(Xcircle(i) - radiocircle(i), X1, X2) = True) And
(Dominio(Ycircle(i) + radiocircle(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(X1, Xcircle(i) - radiocircle(i), Xcircle(i) + radiocircle(i)) = True)
And (Dominio(Y1, Ycircle(i) - radiocircle(i), Ycircle(i) + radiocircle(i)) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(X2, Xcircle(i) - radiocircle(i), Xcircle(i) + radiocircle(i)) = True)
And (Dominio(Y2, Ycircle(i) - radiocircle(i), Ycircle(i) + radiocircle(i)) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(X2, Xcircle(i) - radiocircle(i), Xcircle(i) + radiocircle(i)) = True)
And (Dominio(Y1, Ycircle(i) - radiocircle(i), Ycircle(i) + radiocircle(i)) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
      If (Dominio(X1, Xcircle(i) - radiocircle(i), Xcircle(i) + radiocircle(i)) = True)
And (Dominio(Y1, Ycircle(i) - radiocircle(i), Ycircle(i) + radiocircle(i)) = True) Then
         Es_Correcto1 = True
         GoTo 102
      End If
102
   If (Es_Correcto1 = True) Then
      kcircle1 = kcircle1 + 1
      ReDim Preserve Xcircle1(kcircle1)
      ReDim Preserve Ycircle1(kcircle1)
      ReDim Preserve radiocircle1(kcircle1)
      Xcircle1(kcircle1) = Xcircle(i)
      Ycircle1(kcircle1) = Ycircle(i)
```

```
      radiocircle1(kcircle1) = radiocircle(i)
   End If
   Next
   For i = 1 To karc
      Es_Correcto1 = False
      If i = 1 Then
         karc1 = 0
      End If
      If (Dominio(Xarc(i) + radioarc(i), X1, X2) = True) And (Dominio(Yarc(i) +
radioarc(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
      If (Dominio(Xarc(i) - radioarc(i), X1, X2) = True) And (Dominio(Yarc(i) -
radioarc(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
      If (Dominio(Xarc(i) + radioarc(i), X1, X2) = True) And (Dominio(Yarc(i) -
radioarc(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
      If (Dominio(Xarc(i) - radioarc(i), X1, X2) = True) And (Dominio(Yarc(i) +
radioarc(i), Y1, Y2) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
      If (Dominio(X1, Xarc(i) - radioarc(i), Xarc(i) + radioarc(i)) = True) And
(Dominio(Y1, Yarc(i) - radioarc(i), Yarc(i) + radioarc(i)) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
       If (Dominio(X2, Xarc(i) - radioarc(i), Xarc(i) + radioarc(i)) = True) And
(Dominio(Y2, Yarc(i) - radioarc(i), Yarc(i) + radioarc(i)) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
      If (Dominio(X2, Xarc(i) - radioarc(i), Xarc(i) + radioarc(i)) = True) And
(Dominio(Y1, Yarc(i) - radioarc(i), Yarc(i) + radioarc(i)) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
```

```
      If (Dominio(X1, Xarc(i) - radioarc(i), Xarc(i) + radioarc(i)) = True) And
(Dominio(Y1, Yarc(i) - radioarc(i), Yarc(i) + radioarc(i)) = True) Then
         Es_Correcto1 = True
         GoTo 103
      End If
103
   If (Es_Correcto1 = True) Then
      karc1 = karc1 + 1
      ReDim Preserve Xarc1(karc1)
      ReDim Preserve Yarc1(karc1)
      ReDim Preserve radioarc1(karc1)
      ReDim Preserve aarc1(karc1)
      ReDim Preserve barc1(karc1)
      Xarc1(karc1) = Xarc(i)
      Yarc1(karc1) = Yarc(i)
      radioarc1(karc1) = radioarc(i)
      aarc1(karc1) = aarc(i)
      barc1(karc1) = barc(i)
   End If
   Next
   BAND = 1

End Sub


Private Function Dominio(X As Double, Compare_Con1 As Double, Compare_Con2
As Double) As Boolean

   Dim d1 As Double, Cambio As Boolean
   Cambio = False
   If Compare_Con1 > Compare_Con2 Then
      d1 = Compare_Con1
      Compare_Con1 = Compare_Con2
      Compare_Con2 = d1
      Cambio = True
   End If
   Dominio = False
   If X >= Compare_Con1 And X <= Compare_Con2 Then
      Dominio = True
   End If
   If Cambio = True Then
      'porque las coordenadas fuera de la función cambian de valor
```

```
    d1 = Compare_Con1
    Compare_Con1 = Compare_Con2
    Compare_Con2 = d1
  End If

End Function

Public Sub Revisión_fp(Variable1 As String, Variable2 As Single)

  Dim Variable As String, Cadena As String
  Dim Tamaño As Integer, i As Integer
  Dim B1 As Boolean, B2 As Boolean, B3 As Boolean
10
  If Variable1 = "" And Variable2 <> 0 Then
    Exit Sub
  End If
  Cadena = Variable1 'porque quiero conservar el valor original de fpo
  Tamaño = Len(Variable1)
  B1 = False
  B2 = False
  B3 = False
  For i = 1 To Tamaño
    Variable = Mid(Variable1, i, 1)
    If ((Variable = ",") Or (Variable = ".")) And (B3 = True) Then
      B2 = True
      Exit For
    End If
    Select Case Variable
      Case 0 To 9
        'no hacer nada
      Case ".", ","
        B3 = True
      Case Else
        B2 = True
        Exit For
    End Select
  Next
  If B2 = True Then
    Variable2 = MsgBox("Introdujo Valores no Numéricos", vbCritical, "")
    Variable1 = InputBox("¡ Factor de Potencia de las Cargas ¡")
    GoTo 10
  End If
  For i = 1 To Tamaño
```

```
    Variable = Mid(Variable1, i, 1)
      If Variable = "." Or Variable = "," Then
        B1 = True
        Exit For
      End If
    Next
    If B1 = True Then
      Cadena = Mid(Cadena, i + 1, Tamaño - i)
      Variable2 = Cadena
    Else
      Variable2 = MsgBox("Introdujo Mal el Factor de Potencia, Faltó el Punto
Decimal", vbCritical, " ")
      Variable1 = InputBox("¡ Factor de Potencia de las Cargas ¡")
      GoTo 10
    End If
    Tamaño = Len(Variable1)
    Tamaño = Tamaño - i
    If Variable2 > 1 Then
      Variable2 = Variable2 * 10 ^ (-Tamaño)
    End If
    Variable2 = CSng(Variable2)

End Sub
```

```
Public largo As Double, factor As Double
Public Sub Cambia_escala(Objeto As Object)

   Objeto.ScaleWidth = (ancho * 1.01)
   Objeto.ScaleHeight = (alto * 1.01)
   Objeto.ScaleLeft = (izquierda * (0.5))
   Objeto.ScaleTop = (tope * (-0.5))
    'Objeto.Scale (izquierda, tope)-(ancho, alto)
End Sub

Public Sub Cambia_Y()

   Dim d1 As Double
   If CambiaYB = False Then
      alto = 0
      ancho = 0
      izquierda = 0
      tope = 0
      Call Busca_Extremos
      d1 = alto
      For i = 1 To kline
         Ylinei(i) = d1 - Ylinei(i)
         Ylinef(i) = d1 - Ylinef(i)
      Next
      For i = 1 To kcircle
         Ycircle(i) = d1 - Ycircle(i)
      Next
      For i = 1 To karc
         Yarc(i) = d1 - Yarc(i)
      Next
      CambiaYB = True
   End If

End Sub

Private Sub Cambiar_a_Double(V As String, X As Double)

   V = Replace(V, ".", ",")
   X = CDbl(V)

End Sub
```

```
Public Sub Dibujar_dxf(Objeto As Object)

   pi = 3.141592654
   If (Objeto.ScaleHeight) >= (Objeto.ScaleWidth) Then
      largo = (Objeto.ScaleHeight)
   Else
      largo = (Objeto.ScaleWidth)
   End If

   If Mantener_EncB = False Then
      If largo > 20 Then
         factor = 50
      Else
         factor = 500
      End If
      For i = 1 To klinc
         Objeto.PSet (Xlinei(i), Ylinei(i))
         Objeto.Line (Xlinei(i), Ylinei(i))-(Xlinef(i), Ylinef(i))
      Next
      For i = 1 To kcircle
         Objeto.Circle (Xcircle(i), Ycircle(i)), radiocircle(i)
      Next
      For i = 1 To karc
         Call Arco(Objeto, Xarc(i), Yarc(i), radioarc(i), aarc(i), barc(i))
      Next
   Else
      factor = 500
      For i = 1 To kline1
         Objeto.PSet (Xlinei1(i), Ylinei1(i))
         Objeto.Line (Xlinei1(i), Ylinei1(i))-(Xlinef1(i), Ylinef1(i))
      Next
      For i = 1 To kcircle1
         Objeto.Circle (Xcircle1(i), Ycircle1(i)), radiocircle1(i)
      Next
      For i = 1 To karc1
         Call Arco(Objeto, Xarc1(i), Yarc1(i), radioarc1(i), aarc1(i), barc1(i))
      Next
   End If

End Sub

Public Sub Digitalizar_DXF()
```

```
Dim segmentos As Integer, N As Integer, probando As Integer
karc = 0
kcircle = 0
kline = 0
kpoline = 0
probando = 0
i = Aqui
Do
   i = i + 1
      Select Case Vector(i)
         Case "ARC"
            karc = karc + 1
            ReDim Preserve Xarc(karc)
            ReDim Preserve Yarc(karc)
            ReDim Preserve aarc(karc)
            ReDim Preserve barc(karc)
            ReDim Preserve radioarc(karc)
            N = 0
            Call Buscar_Principio(N)
            Call Cambiar_a_Double(Vector(i + N), Xarc(karc))
            Call Cambiar_a_Double(Vector(i + N + 2), Yarc(karc))
            Call Cambiar_a_Double(Vector(i + N + 6), radioarc(karc))
            Call Cambiar_a_Double(Vector(i + N + 10), aarc(karc))
            Call Cambiar_a_Double(Vector(i + N + 12), barc(karc))
            i = i + N + 13
         Case "CIRCLE"
            kcircle = kcircle + 1
            ReDim Preserve Xcircle(kcircle)
            ReDim Preserve Ycircle(kcircle)
            ReDim Preserve radiocircle(kcircle)
            N = 0
            Call Buscar_Principio(N)
            Call Cambiar_a_Double(Vector(i + N), Xcircle(kcircle))
            Call Cambiar_a_Double(Vector(i + N + 2), Ycircle(kcircle))
            Call Cambiar_a_Double(Vector(i + N + 6), radiocircle(kcircle))
            i = N + 7 + i
         Case "LINE"
            kline = kline + 1
            ReDim Preserve Xlinei(kline)
            ReDim Preserve Ylinei(kline)
            ReDim Preserve Xlinef(kline)
            ReDim Preserve Ylinef(kline)
            N = 0
```

```
            Call Buscar_Principio(N)
            Call Cambiar_a_Double(Vector(i + N), Xlinei(kline))
            Call Cambiar_a_Double(Vector(i + N + 2), Ylinei(kline))
            Call Cambiar_a_Double(Vector(i + N + 6), Xlinef(kline))
            Call Cambiar_a_Double(Vector(i + N + 8), Ylinef(kline))
            i = i + N + 9
         Case "LWPOLYLINE"

            segmentos = 0
            Do
               segmentos = segmentos + 1
               'segmentos = CInt(Vector(i + 10))
            Loop Until (Vector(i + segmentos) = "AcDbPolyline")
            segmentos = segmentos + 2
            segmentos = CInt(Vector(i + segmentos))
            N = 0
            Call Buscar_Principio(N)
            probando = N
            'If Verificar_Poli(segmentos, N - 2) = False Then
               j = 0
               Do Until j = segmentos - 1
                  j = j + 1
                  If Verificar_Poliarc(CInt(Vector(i + N - 1)), CInt(Vector(i + N +
1)), CInt(Vector(i + N + 3)), CInt(Vector(i + N + 5))) = False Then
                     kline = kline + 1
                     ReDim Preserve Xlinei(kline)
                     ReDim Preserve Ylinei(kline)
                     ReDim Preserve Xlinef(kline)
                     ReDim Preserve Ylinef(kline)
                     Call Cambiar_a_Double(Vector(i + N), Xlinei(kline))
                     Call Cambiar_a_Double(Vector(i + 2 + N), Ylinei(kline))
                     Call Cambiar_a_Double(Vector(i + 4 + N), Xlinef(kline))
                     Call Cambiar_a_Double(Vector(i + 6 + N), Ylinef(kline))
                  End If
                  N = N + 4
               Loop
            'End If
            i = i + probando + (segmentos) * 4
      End Select

   Loop Until Vector(i) = "ENDSEC"

'Call Verificar
```

```
End Sub


Public Sub Arco(Objeto As Object, X As Double, Y As Double, r As Double, a As
Double, b As Double)

   Dim s As Double, div As Double, B1 As Boolean, l As Double
   Dim ang1 As Double, ang2 As Double, contar As Long
   contar = 0: l = Abs(a - b) * r
   ang1 = a
   ang2 = b
   Call Reducir_ang(ang1)
   Call Reducir_ang(ang2)
   pi = 3.141592654
   B1 = False
   div = largo
   If l < (0.225 * div) Then
      If div < 10 Then
         div = 1000
      ElseIf div > 10 And div < 1000 Then
      div = factor * 2
      ElseIf div > 1000 Then
         div = 15
      End If
   Else
      div = 1000
   End If
   s = Abs(ang2 - ang1)
   s = s / (div)
   If s >= ang2 Then
      s = 0.1 * ang2
   End If
   If s = 0 Then
      Exit Sub
   End If
   If ang2 > ang1 Then
      Do
         contar = contar + 1
         If contar >= 50000 Then
            contar = MsgBox("Desbordamiento en el Sub Arco, Línea" & i, vbCritical,
"")
            End
         End If
```

```
        Objeto.PSet (X + (r * Cos(ang1)), Y - (r * Sin(ang1)))
        ang1 = ang1 + s
      Loop Until ang1 >= ang2
    Else
      Do
        Objeto.PSet (X + (r * Cos(ang1)), Y - (r * Sin(ang1)))
        If B1 = True Then
          If ang1 >= ((Abs((1 - s))) * ang2) Then
            B1 = False
            'ang1 = ang1 - s
            ang1 = 0
          End If
          If ang2 = 0 Then
            Exit Sub
          End If
        End If
        If ang1 > ((2 * pi) - s) And ang1 < ((2 * pi) + s) Then
          ang1 = -s
          B1 = True
        End If
        ang1 = ang1 + s
        contar = contar + 1
        If contar >= 50000 Then
          contar = MsgBox("Desbordamiento en el Sub Arco, Línea" & i, vbCritical,
"")
          End
        End If
      Loop Until ang1 <= (ang2) And B1 = False
    End If

End Sub

Public Sub Radianes()

  If angB = False Then
    For i = 1 To karc
      Call Convertir_ang(aarc(i))
      Call Convertir_ang(barc(i))
    Next
  End If
   angB = True

End Sub
```

```
Private Sub Convertir_ang(a As Double)

    pi = 3.141592654
    a = (a / 180) * pi
    If a <= ((2 * pi) / (360 * 2)) And a > 0 Then
      a = 0
    End If

End Sub

Private Sub Busca_Extremos()

   mas_negativox = 0
   mas_negativoy = 0
   mas_positivox = 0
   mas_positivoy = 0
   For i = 1 To kline
     Call Negativos_Positivos(Xlinei(i), mas_positivox, mas_negativox,
menos_positivox, menos_negativox)
     Call Negativos_Positivos(Ylinei(i), mas_positivoy, mas_negativoy,
menos_positivoy, menos_negativoy)
     Call Negativos_Positivos(Xlinef(i), mas_positivox, mas_negativox,
menos_positivox, menos_negativox)
     Call Negativos_Positivos(Ylinef(i), mas_positivoy, mas_negativoy,
menos_positivoy, menos_negativoy)
   Next
   For i = 1 To kcircle
     Call Negativos_Positivos(Xcircle(i) + (radiocircle(i)) * (Sgn(Xcircle(i))),
mas_positivox, mas_negativox, menos_positivox, menos_negativox)
     Call Negativos_Positivos(Ycircle(i) + radiocircle(i) * (Sgn(Ycircle(i))),
mas_positivoy, mas_negativoy, menos_positivoy, menos_negativoy)
   Next
   For i = 1 To karc
     Call Negativos_Positivos(Xarc(i) + (radioarc(i)) * (Sgn(Xarc(i))),
mas_positivox, mas_negativox, menos_positivox, menos_negativox)
     Call Negativos_Positivos(Yarc(i) + (radioarc(i)) * (Sgn(Yarc(i))),
mas_positivoy, mas_negativoy, menos_positivoy, menos_negativoy)
   Next
   'alto = mas_positivoy
   'ancho = mas_positivox
   'izquierda = mas_negativox
   'tope = mas_negativoy
   alto = mas_positivoy - mas_negativoy
```

```
ancho = mas_positivox - mas_negativox
Select Case mas_negativox
   Case 0
      Select Case mas_negativoy
         Case 0
            'no existen negativos
            If mas_positivox = 0 Or mas_positivoy = 0 Then
               'no existen dimensiones
               i = MsgBox("El Dibujo No Tiene Dimensiones", vbCritical, "")
               End
            Else
               'existen positivos
               izquierda = menos_positivox * 0.5
               tope = menos_positivoy
            End If
         Case Else

      End Select
      Case Else
         'existen negativos en x
         Select Case mas_negativoy
            Case 0
               'no existen negativos en y
               If mas_positivox = 0 And mas_positivoy = 0 And mas_negativoy = 0
Then
                  'no existen dimensiones
                  i = MsgBox("El Dibujo No Tiene Dimensiones", vbCritical, "")
                  End
               ElseIf mas_negativoy <> 0 Then
                  izquierda = mas_negativox * 0.5
                  tope = mas_negativoy
               ElseIf mas_negativoy = 0 Then
                  izquierda = mas_negativox * 0.5
                  tope = menos_positivoy * 0.5
               End If
            Case Else
               'existen negativos
               izquierda = mas_negativox
               tope = mas_negativoy

         End Select
   End Select
```

```
ancho = mas_positivox - mas_negativox
Select Case mas_negativox
  Case 0
    Select Case mas_negativoy
      Case 0
        'no existen negativos
        If mas_positivox = 0 Or mas_positivoy = 0 Then
          'no existen dimensiones
          i = MsgBox("El Dibujo No Tiene Dimensiones", vbCritical, "")
          End
        Else
          'existen positivos
          izquierda = menos_positivox * 0.5
          tope = menos_positivoy
        End If
      Case Else

    End Select
  Case Else
    'existen negativos en x
    Select Case mas_negativoy
      Case 0
        'no existen negativos en y
        If mas_positivox = 0 And mas_positivoy = 0 And mas_negativoy = 0
Then
          'no existen dimensiones
          i = MsgBox("El Dibujo No Tiene Dimensiones", vbCritical, "")
          End
        ElseIf mas_negativoy <> 0 Then
          izquierda = mas_negativox * 0.5
          tope = mas_negativoy
        ElseIf mas_negativoy = 0 Then
          izquierda = mas_negativox * 0.5
          tope = menos_positivoy * 0.5
        End If
      Case Else
        'existen negativos
        izquierda = mas_negativox
        tope = mas_negativoy

    End Select
End Select
```

```
End Sub


Private Sub Negativos_Positivos(X As Double, a As Double, b As Double, c As
Double, d As Double)

   If X >= 0 Then
      If X > a Then
         a = X
         If c = 0 Then
            c = X
         ElseIf X < c Then
            c = X
         End If
      End If
   ElseIf X < b Then
      b = X
      If d = 0 Then
         d = X
      ElseIf X > d Then
         d = X
      End If
   End If

End Sub

Private Sub Buscar_Principio(Valor As Integer)

   Dim B1 As Boolean
   Valor = 1
   B1 = False
   Do Until B1 = True
      If Solo_Numeros((Vector(i + Valor))) = True Then
         If CInt(Vector(i + Valor)) = 10 Then
            If Solo_Numeros(Vector(i + Valor + 2)) = True Then
               If CInt(Vector(i + Valor + 2)) = 20 Then
                  B1 = True
                  Valor = Valor + 1
               Else
                  Valor = Valor + 1

               End If
            Else
```

```
            Valor = Valor + 1

          End If
        Else
          Valor = Valor + 1

        End If
      Else
        Valor = Valor + 1

      End If

  Loop
End Sub


Private Function Solo_Numeros(Expresion As String) As Boolean

  Dim j As Integer, Cadena As String, Cadena1 As String
  Solo_Numeros = True
  For j = 1 To Len(Expresion)
    Cadena1 = (Mid(Expresion, j, 1))
    If Cadena1 = " " Then
      'no hacer nada
    Else
      Cadena = Cadena & Cadena1
    End If
  Next
  Expresion = Cadena
  For j = 1 To Len(Expresion)
    If IsNumeric(Mid(Expresion, j, 1)) = False Then
      Solo_Numeros = False
    End If
  Next

End Function

Private Sub Reducir_ang(a As Double)
  pi = 3.141592654
  If a > (2 * pi) Then
    Do Until a <= (2 * pi)
      a = a - 2 * pi
    Loop
```

```
    End If

End Sub
Public Sub Verificar()

    For i = 1 To kline
        Debug.Print "Xiline(" & i; ")=" & Xlinei(i); " Yiline(" & i; ")=" & Ylinei(i)
        Debug.Print "Xfline(" & i; ")=" & Xlinef(i); " Yfline(" & i; ")=" & Ylinef(i)
    Next
    For i = 1 To kcircle
        Debug.Print "Xcircle(" & i; ")=" & Xcircle(i); " Ycircle(" & i; ")=" & Ycircle(i)
        Debug.Print "radiocircle(" & i; ")=" & radiocircle(i)
    Next
    For i = 1 To karc
        Debug.Print "Xarc(" & i; ")=" & Xarc(i); " Yarc(" & i; ")=" & Yarc(i)
        Debug.Print "radioarc(" & i; ")=" & radioarc(i); " aarc(" & i; ")=" & aarc(i); "
barc(" & i; ")=" & barc(i)
    Next

End Sub
Private Function Verificar_Poli(seg As Integer, ene As Integer) As Boolean
    Dim I As Integer
    Verificar_Poli = False
    For I = 1 To seg * 4 Step 2
        If (Vector(i + ene + 1) <> " 10") And (Vector(i + ene + 1) <> " 20") Then
            Verificar_Poli = True
            Exit Function
        End If
    Next
End Function

Private Function Verificar_Poliarc(Cadena1 As Integer, Cadena2 As Integer,
Cadena3 As Integer, Cadena4 As Integer) As Boolean
    Verificar_Poliarc = False
    If (Cadena1 <> 10) Or (Cadena2 <> 20) Or (Cadena3 <> 10) Or (Cadena4 <> 20)
Then
        Verificar_Poliarc = True
    End If

End Function
```

```
'Aquí va los subprogramas del manejo de archivos
Dim NumeroArchivo As Integer
Public Sub Abrir_mzc()

   DiseñarB = False: VoltajeB = False: CorrB = False
   Form1.CommonDialog1.CancelError = True
   On Error GoTo quehacer
   Form1.CommonDialog1.FileName = ""
   Form1.CommonDialog1.DialogTitle = "Cargar Sector"
   Form1.CommonDialog1.DefaultExt = "mzc"
   Form1.CommonDialog1.Filter = " Matriz de Conexión (*.mzc)|*.mzc|"
   Form1.CommonDialog1.Flags = &H2& 'FileMustExist
   Form1.CommonDialog1.ShowOpen 'para abrir un archivo existente
   If Form1.CommonDialog1.FileName <> "" Then 'Abrir fichero para leetuta por el
canal 1
      Variable = Form1.CommonDialog1.FileName
      Call Obtener_Ruta(Variable)
      Call Obtener_Nombre(Ruta, Variable)
      Nombre_Matriz = Nombre
      NumeroArchivo = FreeFile
      Open Form1.CommonDialog1.FileName For Input As #NumeroArchivo
      MousePointer = vbHourglass
      Input #NumeroArchivo, N, F, COND, Res, Rea, fp, Mayor, Perdidas
      Input #NumeroArchivo, alto1, ancho1, tope1, izquierda1, xx, yy, xx1, yy1, radio
      If N > 0 Then ReDim NI(N)
      If N > 0 Then ReDim NF(N)
      If N > 0 Then ReDim Longitud(N)
      If N > 0 Then ReDim Tramo(N)
      If N > 0 Then ReDim Resistencia(N)
      If N > 0 Then ReDim Reactancia(N)
      If N > 0 Then ReDim Existente(N + 1)
      If N > 0 Then ReDim Carga(N)
      If N > 0 Then ReDim X(N + 1, N + 1)
      If N > 0 Then ReDim r(N + 1, N + 1)
      If N > 0 Then ReDim nodos(F + 1)
      If N > 0 Then ReDim XPosi(N)
      If N > 0 Then ReDim YPosi(N)
      If N > 0 Then ReDim XPosf(N)
      If N > 0 Then ReDim YPosf(N)
      If N > 0 Then ReDim Atriv(N)
      If N > 0 Then ReDim Voltaje(F)
      If N > 0 Then ReDim Corr(N)
      If N > 0 Then ReDim Corriente(F)
```

```
      Aux = N
      For i = 1 To N
         Input #NumeroArchivo, Tramo(i), NI(i), NF(i), XPosi(i), YPosi(i), XPosf(i),
YPosf(i), Longitud(i), Resistencia(i), Reactancia(i), Corr(i)
      Next
      For i = 1 To F - 1
         Input #NumeroArchivo, nodos(i), Carga(i), Voltaje(i), Corriente(i)
      Next
      Close #NumeroArchivo 'cierra el fichero
      For i = 1 To N
         Atriv(i) = False
      Next
      Else

   End If
   Form1.Label7.Visible = True
   Elemento_Dimensiones Form1.Label7, 12, 5, 61, 3
   Form1.Label7.Caption = "Sector:" & Nombre_Matriz
   Form1.Picture1.Cls
   Call Encuadrar(izquierda1, tope1, ancho1, alto1)
   Form1.Picture1.Scale (izquierda1, tope1)-(ancho1, alto1)
   Call Dibujo_esp(Form1.Picture1)
   Call Dibujar_dxf(Form1.Picture1)
   Call Numerar_Matriz(Nombre)
   Call Numerar_nodos(Form1.Picture1)
   MousePointer = vbDefault
   Mantener_EncB = True
   'Escribir_Nombre Form1.Picture1
quehacer:
      Exit Sub

End Sub

Public Sub Nuevos_Valores()

   If Mantener_EncB = True Then

      alto = alto1
      ancho = ancho1
      izquierda = izquierda1
      tope = tope1

   End If
```

```
End Sub
Public Sub Nuevos()

    DiseñarB = False
    angB = False
    CambiaYB = False
    EncuadrarB = False
    Mantener_EncB = False
    AbrirlosB = False
    Call Invisible
    ElementoC = 1
    Form1.CommonDialog1.CancelError = True
    On Error GoTo quehacer
    c = 0
    With Form1.CommonDialog1
        .FileName = ""
        .DialogTitle = "Cargar Archivo DXF"
        .DefaultExt = "dxf"
        .Filter = " Archivos DXF (*.dxf)|*.dxf"
        .Flags = &H2& 'FileMustExist
        .ShowOpen 'para abrir un archivo existente
        Variable = .FileName
        Call Obtener_Ruta(Variable)
        Ruta_DXF = Ruta
        Call Obtener_Nombre(Ruta, Variable)
        Nombre_DXF = Nombre
        If .FileName <> "" Then 'Abrir fichero para leetuta por el canal 1
            Form1.MousePointer = vbHourglass
            ReDim Elementos(2)
            Elementos(1) = Nombre
            NumeroArchivo = FreeFile
            Open .FileName For Input As #NumeroArchivo
            MousePointer = vbHourglass
            Do Until EOF(1)
                Line Input #NumeroArchivo, Cadena
                c = c + 1
                ReDim Preserve Vector(c)
                Vector(c) = Cadena
                If Cadena = "ENTITIES" Then
                    Aqui = c
                End If
            Loop
            Close #NumeroArchivo 'cierra el fichero
```

```
    End If
  End With

  Form1.MousePointer = vbDefault
  Form1.Command4.Visible = False: Porcentaje1 = 3: Porcentaje2 = 5
  Form1.Picture1.Cls
  Form1.MousePointer = vbHourglass
  Call Digitalizar_DXF
  Erase Vector
  If N <> 0 Then
     Erase NI
     Erase NF
     Erase Longitud
     Erase Tramo
     Erase Resistencia
     Erase Reactancia
     Erase Existente
     Erase Carga
     Erase X
     Erase r
     N = 0
     F = 0
     COND = ""
  End If
  DibujaProB = True
  Form1.MousePointer = vbDefault
  Form1.Visible = False
  Form2.Visible = True
quehacer:
  Exit Sub

End Sub
Public Sub Guardarlos()

  DiseñarB = False
  Form1.CommonDialog1.CancelError = True
  On Error GoTo quehacer
  Call Genera_Nombre
  Form1.CommonDialog1.FileName = Ruta & Nombre_Matriz
  With Form1.CommonDialog1
     .DialogTitle = "Guadar Sector"
     .DefaultExt = "mzc"
```

```
    .Filter = "Matriz de Conexción (*.mzc)|*.mzc"
    .Flags = &H2& 'FileMustExist
    .ShowSave 'para abrir un archivo existente
    Variable = .FileName
    Call Obtener_Ruta(Variable)
    Call Obtener_Nombre(Ruta, Variable)
    If .FileName <> "" Then
      'Abrir fichero para lectuta por el canal NumeroArchivo
      If ElementoC = 1 Then
        ElementoC = ElementoC + 1
        Elementos(ElementoC) = Nombre
      Else
        If ModificarTB = False Then
          ElementoC = ElementoC + 1
        Else
          ModificarTB = False
        End If
        ReDim Preserve Elementos(ElementoC)
        Elementos(ElementoC) = Nombre
      End If
      NumeroArchivo = FreeFile
      Open .FileName For Output As #NumeroArchivo
        Write #NumeroArchivo, N, F, COND, Res, Rea, fp, Redondear(Mayor, 2),
Redondear(Perdidas, 2)
        Write #NumeroArchivo, Redondear(alto1, 4), Redondear(ancho1, 4),
Redondear(tope1, 4), Redondear(izquierda1, 4), Redondear(xx, 6), Redondear(yy, 6),
Redondear(xx1, 6), Redondear(yy1, 6), Redondear(radio, 6)
      For i = 1 To N
        Write #NumeroArchivo, Tramo(i), NI(i), NF(i), Redondear(XPosi(i), 4),
Redondear(YPosi(i), 4), Redondear(XPosf(i), 4), Redondear(YPosf(i), 4),
Redondear(Longitud(i), 4), Redondear(Resistencia(i), 4), Redondear(Reactancia(i),
4), Redondear(Corr(i), 4)
      Next
      For i = 1 To F - 1
        Write #NumeroArchivo, nodos(i + 1), Carga(i), Redondear(Voltaje(i), 4),
Redondear(Corriente(i), 4)
      Next
      Close #NumeroArchivo 'cierra el fichero
    End If
    .FileName = Ruta & Nombre_Pro
      If .FileName <> "" Then
      'Abrir fichero para lectuta por el canal NumeroArchivo
      NumeroArchivo = FreeFile
```

```
        Open .FileName For Output As #NumeroArchivo
        Write #NumeroArchivo, ElementoC, Porcentaje1, Porcentaje2
        Write #NumeroArchivo, Elementos(1)
        For i = 2 To ElementoC
          Write #NumeroArchivo, Elementos(i)
        Next
        Close #NumeroArchivo 'cierra el fichero
      End If
     End With
   If Err.Number = cdlCancel Then
     'no hacer nada
   End If
quehacer:
    Exit Sub
End Sub
Public Sub GSuper_Proyec()

   DiseñarB = False
   Form1.CommonDialog1.CancelError = True
   On Error GoTo quehacer
   With Form1.CommonDialog1
      .FileName = Ruta & Nombre_Pro
      .DialogTitle = "Guardar Proyecto"
      .DefaultExt = "pvd"
      .Filter = " Proyecto de VBCAD  (*.pvd)|*.pvd"
      .Flags = &H2& 'FileMustExist
      .ShowSave 'para abrir un archivo existente
      Variable = .FileName
      Call Obtener_Ruta(Variable)
      Call Obtener_Nombre(Ruta, Variable)
      Nombre_Pro = Nombre
     End With

quehacer:
    Exit Sub

End Sub

Public Sub ASuper_Proyec()

   DiseñarB = False: VoltajeB = False: CorrB = False
   Form1.CommonDialog1.CancelError = True
   On Error GoTo quehacer
```

```
  With Form1.CommonDialog1
    .FileName = ""
    .DialogTitle = "Abrir Proyecto"
    .DefaultExt = "pvd"
    .Filter = " Proyecto de VBCAD  (*.pvd)|*.pvd"
    .Flags = &H2& 'FileMustExist
    .ShowOpen 'para abrir un archivo existente
    Variable = .FileName
    Call Obtener_Ruta(Variable)
    Call Obtener_Nombre(Ruta, Variable)
    ruta1 = Ruta
    Nombre_Pro = Nombre
    If .FileName <> "" Then
      'Abrir fichero para lectuta por el canal NumeroArchivo
      NumeroArchivo = FreeFile
      Open .FileName For Input As #NumeroArchivo
      Input #NumeroArchivo, ElementoC, Porcentaje1, Porcentaje2
      ReDim Elementos(ElementoC)
      Input #NumeroArchivo, Elementos(1)
      For i = 2 To ElementoC
        Input #NumeroArchivo, Elementos(i)
      Next

      Close #NumeroArchivo 'cierra el fichero
    End If
   End With
  Form1.MousePointer = vbHourglass
  Form1.CommonDialog1.FileName = Ruta & Elementos(1)
  If Form1.CommonDialog1.FileName <> "" Then 'Abrir fichero para lectuta por el
canal 1
    NumeroArchivo = FreeFile
    Open Form1.CommonDialog1.FileName For Input As #NumeroArchivo
    Do Until EOF(1)
      Line Input #NumeroArchivo, Cadena
      c = c + 1
      ReDim Preserve Vector(c)
      Vector(c) = Cadena
      If Cadena = "ENTITIES" Then
        Aqui = c
      End If
    Loop
    Close #NumeroArchivo 'cierra el fichero
  End If
```

```
   Contador = 0
   Contador1 = 0
   ReDim Referencia(ElementoC - 1): ReDim Tramo1(ElementoC - 1): ReDim
Preserve COND1(ElementoC - 1): ReDim Preserve fp1(ElementoC - 1)
   ReferenciaC = ElementoC - 1: ReDim Preserve Perdidas1(ReferenciaC): ReDim
Preserve Mayor1(ReferenciaC)
   For i = 2 To ElementoC
      Form1.CommonDialog1.FileName = Ruta & Elementos(i)
      If Form1.CommonDialog1.FileName <> "" Then 'Abrir fichero para lectuta por
el canal 1
         NumeroArchivo = FreeFile
         Open Form1.CommonDialog1.FileName For Input As #NumeroArchivo
         Input #NumeroArchivo, N, F, COND, Res, Rea, fp, Mayor, Perdidas
         Input #NumeroArchivo, alto1, ancho1, tope1, izquierda1, xx, yy, xx1, yy1,
radio
         If N > 0 Then ReDim NI(N)
         If N > 0 Then ReDim NF(N)
         If N > 0 Then ReDim Longitud(N)
         If N > 0 Then ReDim Tramo(N)
         If N > 0 Then ReDim Resistencia(N)
         If N > 0 Then ReDim Reactancia(N)
         If N > 0 Then ReDim Existente(N + 1)
         If N > 0 Then ReDim Carga(N)
         If N > 0 Then ReDim X(N + 1, N + 1)
         If N > 0 Then ReDim r(N + 1, N + 1)
         If N > 0 Then ReDim nodos(F + 1)
         If N > 0 Then ReDim XPosi(N)
         If N > 0 Then ReDim YPosi(N)
         If N > 0 Then ReDim XPosf(N)
         If N > 0 Then ReDim YPosf(N)
         If N > 0 Then ReDim Voltaje(F)
         If N > 0 Then ReDim Corr(N)
         If N > 0 Then ReDim Corriente(F)
         Aux = N
         For j = 1 To N
            Input #NumeroArchivo, Tramo(j), NI(j), NF(j), XPosi(j), YPosi(j),
XPosf(j), YPosf(j), Longitud(j), Resistencia(j), Reactancia(j), Corr(j)
         Next
         For j = 1 To F - 1
            Input #NumeroArchivo, nodos(j), Carga(j), Voltaje(j), Corriente(j)
         Next
         Close #NumeroArchivo 'cierra el fichero
         If Contador = 0 Then
```

```
            ReDim XPosi1(N): ReDim XPosf1(N): ReDim YPosi1(N): ReDim
YPosf1(N): ReDim Voltaje1(F + 1)
            ReDim NI1(N): ReDim NF1(N): ReDim Corr1(N): ReDim Longitud1(N)
          Else
            ReDim Preserve XPosi1(N + Contador): ReDim Preserve XPosf1(N +
Contador): ReDim Preserve YPosi1(N + Contador): ReDim Preserve YPosf1(N +
Contador): ReDim Preserve Voltaje1(F + Contador1 + 1)
            ReDim Preserve NI1(N + Contador): ReDim Preserve NF1(N + Contador):
ReDim Preserve Corr1(N + Contador): ReDim Preserve Longitud1(N + Contador)
          End If
          Call Convertidor_Nodos

        End If
    Next
    CambiaYB = False: angB = False: EncuadrarB = False: Mantener_EncB = False:
'AbrirlosB = True
    Call Centrar_Pantalla(Form1.Picture1)
    Form1.Picture1.Cls
    Call Digitalizar_DXF
    Erase Vector
    Call Cambia_Y
    Call Cambia_escala(Form1.Picture1)
    Call Radianes
    Call Dibujar_dxf(Form1.Picture1)
    Form1.Label6.Visible = True: Form1.Label7.Visible = False
    Elemento_Dimensiones Form1.Label6, 12, 5, 31, 3
    Form1.Picture1.Visible = True: Form1.Label6.Caption = "Proyecto:" &
Nombre_Pro
    'Escribir_Nombre Form1.Picture1
    Call Dibuja_Pro(Form1.Picture1)
    Call Borrar: Numero_Matriz = ElementoC - 1
    Form1.MousePointer = vbDefault
    Discmzc.Enabled = True: DibujaProB = True

quehacer:
    Exit Sub

End Sub

Public Sub Convertidor_Nodos()

    Dim nodos1() As Integer, w As Integer, l As Integer, k As Integer, p As Integer
```

```
  Dim B1 As Boolean, B2 As Boolean, valor1 As Long, valor2 As Long, inicio As
Integer
  ReDim nodos1(F + 1, 2)
  p = 0
  For j = 1 To N
    B1 = False: B2 = False
    Contador = Contador + 1
    XPosi1(Contador) = XPosi(j)
    YPosi1(Contador) = YPosi(j)
    XPosf1(Contador) = XPosf(j)
    YPosf1(Contador) = YPosf(j)
    Corr1(Contador) = Corr(j)
    Longitud1(Contador) = Longitud(j)
    If j = 1 Then
      p = p + 1
      nodos1(p, 1) = NI(j)
      Contador1 = Contador1 + 1
      Referencia(i - 1) = Contador1
      COND1(i - 1) = COND: fp1(i - 1) = fp
      Tramo1(i - 1) = Contador: Perdidas1(i - 1) = Perdidas
      Perdidas1(i - 1) = Redondear(Perdidas1(i - 1), 2)
      nodos1(p, 2) = Contador1: Mayor1(i - 1) = Mayor
      NI1(Contador) = Contador1
      Voltaje1(Contador1) = Voltaje(NI(j))
      Contador1 = Contador1 + 1
      p = p + 1
      nodos1(p, 1) = NF(j)
      nodos1(p, 2) = Contador1
      NF1(Contador) = Contador1
      Voltaje1(Contador1) = Voltaje(NF(j))
      w = 2
    Else
      For k = w To 1 Step -1
        If NI(j) = nodos1(k, 1) Then
          B1 = True
          valor1 = nodos1(k, 2)
        End If
        If NF(j) = nodos1(k, 1) Then
          B2 = True
          valor2 = nodos1(k, 2)
        End If
      Next
      If B1 = True Then
```

```
            NI1(Contador) = valor1
          Else
            p = p + 1
            nodos1(p, 1) = NI(j)
            Contador1 = Contador1 + 1
            NI1(Contador) = Contador1
            nodos1(p, 2) = Contador1
            Voltaje1(Contador1) = Voltaje(NI(j))
          End If
          If B2 = True Then
            NF1(Contador) = valor2
          ElseIf B1 = True Then
            p = p + 1
            Contador1 = Contador1 + 1
            NF1(Contador) = Contador1
            nodos1(p, 1) = NF(j)
            nodos1(p, 2) = Contador1
            Voltaje1(Contador1) = Voltaje(NF(j))
            w = w + 1
            Else
              Contador1 = Contador1 + 1
              NF1(Contador) = Contador1
              p = p + 1
              nodos1(p, 1) = NF(j)
              nodos1(p, 2) = Contador1
              Voltaje1(Contador1) = Voltaje(NF(j))
              w = w + 1
          End If
        End If
      Next

End Sub

Public Sub Obtener_Ruta(Variable As String)
    Dim a As Integer, Arreglo As String, bande As Boolean, Empezar As Integer
    bande = False: Ruta = ""
    For a = Len(Variable) To 1 Step -1
      Arreglo = Mid(Variable, a, 1)
      Select Case Arreglo
        Case "\"
          bande = True
          Empezar = a
          Exit For
```

```
      End Select
    Next
    For a = 1 To Empezar
      Arreglo = Mid(Variable, a, 1)
      Ruta = Ruta & Arreglo
    Next
End Sub
Public Sub Obtener_Nombre(Ruta As String, Variable As String)
    Dim a As Integer, b As Integer, c As Integer, Var As String
    a = Len(Ruta): b = Len(Variable): Nombre = ""
    For c = a + 1 To b
      If Mid(Variable, c, 1) <> "*" Then
        Var = Mid(Variable, c, 1)
        Nombre = Nombre & Var
      Else
        Var = "pvd"
        Nombre = Nombre & Var
      End If
    Next

End Sub
```

```
'Aqui va los subprogramas del manejo de archivos
'Private Nodo_Inicial As Long, Tramo_Inicial As Long, Numero_Tramos As Integer,
Numero_Nodos As Integer
Public Sub Reabrir_Proy()

   Contador = 0
   Contador1 = 0: DiseñarB = False: VoltajeB = False: CorrB = False
   ReDim Referencia(ElementoC - 1): ReDim Tramo1(ElementoC - 1): ReDim
Preserve COND1(ElementoC - 1): ReDim Preserve fp1(ElementoC - 1)
   ReferenciaC = ElementoC - 1: ReDim Preserve Perdidas1(ReferenciaC): ReDim
Preserve Mayor1(ReferenciaC)
   For i = 2 To ElementoC
      Form1.CommonDialog1.FileName = Ruta & Elementos(i)
      If Form1.CommonDialog1.FileName <> "" Then 'Abrir fichero para lectuta por
el canal 1
         NumeroArchivo = FreeFile
         Open Form1.CommonDialog1.FileName For Input As #NumeroArchivo
         Input #NumeroArchivo, N, F, COND, Res, Rea, fp, Mayor, Perdidas
         Input #NumeroArchivo, alto1, ancho1, tope1, izquierda1, xx, yy, xx1, yy1,
radio
         If N > 0 Then ReDim NI(N)
         If N > 0 Then ReDim NF(N)
         If N > 0 Then ReDim Longitud(N)
         If N > 0 Then ReDim Tramo(N)
         If N > 0 Then ReDim Resistencia(N)
         If N > 0 Then ReDim Reactancia(N)
         If N > 0 Then ReDim Existente(N + 1)
         If N > 0 Then ReDim Carga(N)
         If N > 0 Then ReDim X(N + 1, N + 1)
         If N > 0 Then ReDim r(N + 1, N + 1)
         If N > 0 Then ReDim nodos(F + 1)
         If N > 0 Then ReDim XPosi(N)
         If N > 0 Then ReDim YPosi(N)
         If N > 0 Then ReDim XPosf(N)
         If N > 0 Then ReDim YPosf(N)
         If N > 0 Then ReDim Voltaje(F)
         If N > 0 Then ReDim Corr(N)
         If N > 0 Then ReDim Corriente(F)
         Aux = N
         For j = 1 To N
            Input #NumeroArchivo, Tramo(j), NI(j), NF(j), XPosi(j), YPosi(j),
XPosf(j), YPosf(j), Longitud(j), Resistencia(j), Reactancia(j), Corr(j)
         Next
```

```
        For j = 1 To F - 1
          Input #NumeroArchivo, nodos(j), Carga(j), Voltaje(j), Corriente(j)
        Next
        Close #NumeroArchivo 'cierra el fichero
        If Contador = 0 Then
          ReDim XPosi1(N): ReDim XPosf1(N): ReDim YPosi1(N): ReDim
YPosf1(N): ReDim Voltaje1(F + 1)
          ReDim NI1(N): ReDim NF1(N): ReDim Corr1(N): ReDim Longitud1(N)
        Else
          ReDim Preserve XPosi1(N + Contador): ReDim Preserve XPosf1(N +
Contador): ReDim Preserve YPosi1(N + Contador): ReDim Preserve YPosf1(N +
Contador): ReDim Preserve Voltaje1(F + Contador1 + 1)
          ReDim Preserve NI1(N + Contador): ReDim Preserve NF1(N + Contador):
ReDim Preserve Corr1(N + Contador): ReDim Preserve Longitud1(N + Contador)
        End If
        Call Convertidor_Nodos

      End If
    Next
    CambiaYB = False: angB = False: EncuadrarB = False: Mantener_EncB = False:
'AbrirlosB = True
    Form1.Command1.Visible = False: Form1.Label7.Visible = False
    Form1.Command4.Visible = False
    Form1.Picture1.Visible = True
    Call Centrar_Pantalla(Form1.Picture1)
    Form1.Picture1.Cls
    Call Cambia_escala(Form1.Picture1)
    Call Dibujar_dxf(Form1.Picture1)
    Form1.MousePointer = vbDefault

End Sub
Public Sub Genera_Nombre()

Dim La_Cadena As String, I As Integer
    La_Cadena = ""
    For I = 1 To Len(Nombre_Pro)
      If Mid(Nombre_Pro, I, 1) <> "." Then
        La_Cadena = La_Cadena & Mid(Nombre_Pro, I, 1)
      Else
        Exit For
      End If
    Next
    La_Cadena = La_Cadena & "_t"
```

```
    La_Cadena = La_Cadena & CInt(Numero_Matriz)
    La_Cadena = La_Cadena & ".mzc"
    Nombre_Matriz = La_Cadena

End Sub

Public Sub Crea_Carpeta()


    Dim Objeto As FileSystemObject, Carpeta As Folder, Nombre_Carp As String,
contar As Integer
    Dim Archivo As File, letra As String
    Form1.CommonDialog1.CancelError = True
    On Error GoTo quehacer
    With Form1.CommonDialog1
       .FileName = ""
       .DialogTitle = "Guardar Carpeta"
       .DefaultExt = "*"
       .Filter = " Todos los Archivos  (*.*)|*.*"
       .Flags = &H2& 'FileMustExist
       .ShowSave 'para abrir un archivo existente
       Variable = .FileName
       Call Obtener_Ruta(Variable)
       Call Obtener_Nombre(Ruta, Variable)
       Nombre_Pro = Nombre
    End With
    Nombre_Carp = ""
    For contar = 1 To Len(Nombre_Pro)
       If Mid(Nombre_Pro, contar, 1) <> "." Then
          Nombre_Carp = Nombre_Carp & Mid(Nombre_Pro, contar, 1)
       Else
          Exit For
       End If
    Next
    Ruta_DXF = Ruta_DXF & Nombre_DXF
    Set Objeto = New FileSystemObject
    Set Carpeta = Objeto.CreateFolder(Nombre_Carp)
    Ruta = Ruta & Nombre_Carp
    Ruta = Ruta & "\"
    letra = Ruta & Nombre_DXF
    FileCopy Ruta_DXF, letra
    Objeto = Nothing
```

```
quehacer:
   Exit Sub
End Sub
Public Sub Borrar()

   Erase NI: Erase r_N = 0
   Erase NF: F = 0: COND = "": fp = 0: fdp = ""
   Erase Longitud: Erase Voltaje: Erase VoltajeR: Erase VoltajeI
   Erase Tramo: Erase CorrienteR: Erase CorrienteI: Erase Corriente
   Erase Resistencia: Erase Corr: Erase CorrR: Erase CorrI
   Erase Reactancia: Erase Existente: Erase Carga: Erase X


End Sub

Public Sub Numerar_Matriz(Nombre As String)
   Dim cuenta As Integer
   Numero_Matriz = 0
   For cuenta = 2 To ElementoC
      If Elementos(cuenta) = Nombre Then
         Numero_Matriz = cuenta - 1
         Exit For
      End If
   Next
End Sub

Public Sub Convertir_Nodos1()

   Dim nodos1() As Integer, w As Integer, I As Integer, k As Integer, p As Integer
   Dim B1 As Boolean, B2 As Boolean, valor1 As Long, valor2 As Long, inicio As
Integer
   If Contador = 0 Then
      ReDim XPosi1(N): ReDim XPosf1(N): ReDim YPosi1(N): ReDim YPosf1(N):
ReDim Voltaje1(F + 1)
      ReDim NI1(N): ReDim NF1(N): ReDim nodos1(F + 1, 2): ReDim
Referencia(1): Numero_Matriz = 1: ReDim Tramo1(1)
   Else
      ReDim Preserve XPosi1(N + Contador): ReDim Preserve XPosf1(N +
Contador): ReDim Preserve YPosi1(N + Contador): ReDim Preserve YPosf1(N +
Contador): ReDim Preserve Voltaje1(F + Contador1 + 1)
      ReDim Preserve NI1(N + Contador): ReDim Preserve NF1(N + Contador):
ReDim nodos1(F + 1, 2)
      If ModificarTB = False Then
```

```
        Numero_Matriz = Numero_Matriz + 1
      End If
      ReDim Preserve Referencia(Numero_Matriz), ReDim Preserve
Tramo1(Numero_Matriz)
    End If
    p = 0
    For j = 1 To N
      B1 = False: B2 = False
      Contador = Contador + 1
      XPosi1(Contador) = XPosi(j)
      YPosi1(Contador) = YPosi(j)
      XPosf1(Contador) = XPosf(j)
      YPosf1(Contador) = YPosf(j)
      If j = 1 Then
        p = p + 1
        nodos1(p, 1) = NI(j)
        Contador1 = Contador1 + 1
        Referencia(Numero_Matriz) = Contador1
        Tramo1(Numero_Matriz) = Contador
        nodos1(p, 2) = Contador1
        NI1(Contador) = Contador1
        'Voltaje1(Contador1) = Voltaje(NI(j))
        Contador1 = Contador1 + 1
        p = p + 1
        nodos1(p, 1) = NF(j)
        nodos1(p, 2) = Contador1
        NF1(Contador) = Contador1
        'Voltaje1(Contador1) = Voltaje(NF(j))
        w = 2
      Else
        For k = w To 1 Step -1
          If NI(j) = nodos1(k, 1) Then
            B1 = True
            valor1 = nodos1(k, 2)
          End If
          If NF(j) = nodos1(k, 1) Then
            B2 = True
            valor2 = nodos1(k, 2)
          End If
        Next
        If B1 = True Then
          NI1(Contador) = valor1
        Else
```

```
            p = p + 1
            nodos1(p, 1) = NI(j)
            Contador1 = Contador1 + 1
            NI1(Contador) = Contador1
            nodos1(p, 2) = Contador1
            'Voltaje1(Contador1) = Voltaje(NI(j))
          End If
          If B2 = True Then
            NF1(Contador) = valor2
          ElseIf B1 = True Then
            p = p + 1
            Contador1 = Contador1 + 1
            NF1(Contador) = Contador1
            nodos1(p, 1) = NF(j)
            nodos1(p, 2) = Contador1
            'Voltaje1(Contador1) = Voltaje(NF(j))
            w = w + 1
            Else
              Contador1 = Contador1 + 1
              NF1(Contador) = Contador1
              p = p + 1
              nodos1(p, 1) = NF(j)
              nodos1(p, 2) = Contador1
              'Voltaje1(Contador1) = Voltaje(NF(j))
              w = w + 1
          End If
        End If
      Next

End Sub

Public Sub Convertir_Nodos2()

  Dim nodos1() As Integer, w As Integer, I As Integer, k As Integer, p As Integer
  Dim B1 As Boolean, B2 As Boolean, valor1 As Long, valor2 As Long, inicio As
Integer
  Dim Contador2 As Long, Contador3 As Long
  Contador2 = Tramo1(Numero_Matriz): Contador3 = Referencia(Numero_Matriz):
ReDim nodos1(F + 1, 2): p = 0
  If Numero_Matriz = Elemento(C - 1 Then
    ReDim Preserve XPosi1(N + Contador): ReDim Preserve XPosf1(N +
Contador): ReDim Preserve YPosi1(N + Contador): ReDim Preserve YPosf1(N +
Contador): ReDim Preserve Voltaje1(F + Contador1 + 1)
```

```
    ReDim Preserve NI1(N + Contador): ReDim Preserve NF1(N + Contador)
End If
For j = 1 To N
    B1 = False: B2 = False
    If j > 1 Then
        Contador2 = Contador2 + 1
    End If
    XPosi1(Contador2) = XPosi(j)
    YPosi1(Contador2) = YPosi(j)
    XPosf1(Contador2) = XPosf(j)
    YPosf1(Contador2) = YPosf(j)
    If j = 1 Then
        p = p + 1
        nodos1(p, 1) = NI(j)
        'Contador3 = Contador3 + 1
        nodos1(p, 2) = Contador3
        NI1(Contador2) = Contador3
        'Voltaje1(Contador1) = Voltaje(NI(j))
        Contador3 = Contador3 + 1
        p = p + 1
        nodos1(p, 1) = NF(j)
        nodos1(p, 2) = Contador3
        NF1(Contador2) = Contador3
        'Voltaje1(Contador1) = Voltaje(NF(j))
        w = 2
    Else
        For k = w To 1 Step -1
            If NI(j) = nodos1(k, 1) Then
                B1 = True
                valor1 = nodos1(k, 2)
            End If
            If NF(j) = nodos1(k, 1) Then
                B2 = True
                valor2 = nodos1(k, 2)
            End If
        Next
        If B1 = True Then
            NI1(Contador2) = valor1
        Else
            p = p + 1
            nodos1(p, 1) = NI(j)
            Contador3 = Contador3 + 1
            NI1(Contador2) = Contador3
```

```
            nodos1(p, 2) = Contador3
            'Voltaje1(Contador1) = Voltaje(NI(j))
          End If
          If B2 = True Then
            NF1(Contador2) = valor2
          ElseIf B1 = True Then
            p = p + 1
            Contador3 = Contador3 + 1
            NF1(Contador2) = Contador3
            nodos1(p, 1) = NF(j)
            nodos1(p, 2) = Contador3
            'Voltaje1(Contador1) = Voltaje(NF(j))
            w = w + 1
            Else
              Contador3 = Contador3 + 1
              NF1(Contador2) = Contador3
              p = p + 1
              nodos1(p, 1) = NF(j)
              nodos1(p, 2) = Contador3
              'Voltaje1(Contador1) = Voltaje(NF(j))
              w = w + 1
          End If
        End If
    Next

End Sub


Public Sub Elimina_Sec()

    Dim Variable As String
    Dim Objeto As FileSystemObject, letra As String
    On Error GoTo quehacer
    Form1.CommonDialog1.FileName = ""
    Form1.CommonDialog1.DialogTitle = "Eliminar Sector"
    Form1.CommonDialog1.DefaultExt = "mzc"
    Form1.CommonDialog1.Filter = " Matriz de Conexión (*.mzc)|*.mzc|"
    Form1.CommonDialog1.Flags = &H2& 'FileMustExist
    Form1.CommonDialog1.ShowOpen 'para abrir un archivo existente
    If Form1.CommonDialog1.FileName <> "" Then 'Abrir fichero para lectuta por el
canal 1
       Variable = Form1.CommonDialog1.FileName
       Call Obtener_Ruta(Variable)
```

```
    Call Obtener_Nombre(Ruta, Variable)
  End If
  SiNo = MsgBox("Esta Seguro que Desea Eliminar " & Nombre, vbYesNo, "")
  If SiNo = 6 Then
    If ruta1 = Ruta Then
      For i = 2 To ElementoC
        If Elementos(i) = Nombre Then
          If i < ElementoC Then
            For j = i To ElementoC - 1
              Elementos(i) = Elementos(j + 1)
            Next
            Exit For
          End If
        End If
      Next
      ElementoC = ElementoC - 1
      ReDim Preserve Elementos(ElementoC)
      Form1.CommonDialog1.FileName = Ruta & Nombre_Pro
        If Form1.CommonDialog1.FileName <> "" Then
          'Abrir fichero para lectuta por el canal NumeroArchivo
          NumeroArchivo = FreeFile
          Open Form1.CommonDialog1.FileName For Output As
#NumeroArchivo
          Write #NumeroArchivo, ElementoC
          Write #NumeroArchivo, Elementos(1)
            For i = 2 To ElementoC
            Write #NumeroArchivo, Elementos(i)
          Next
          Close #NumeroArchivo 'cierra el fichero
        End If
      Set Objeto = New FileSystemObject
      letra = Ruta & Nombre
      Objeto.DeleteFile (letra)
      If (Nombre = Nombre_Matriz) Then
        Call Borrar_Nombre_Matriz = "": DiseñarB = False: Call Invisible
        Form1.Picture1.Visible = True: Form1.MousePointer = vbHourglass
        Call Reabrir_Proy
        CambioProyB = False
        Form1.Picture1.Visible = True
        Call Dibuja_Pro(Form1.Picture1)
        DibujaProB = True
        Form1.MousePointer = vbDefault
      End If
```

```
        Objeto = Nothing
      Else
        i = MsgBox("Ud. Desea Eliminar un Sector de un Proyecto Distinto al que
Está Abierto", vbCritical)
        Exit Sub
      End If
    Else
      Exit Sub
    End If
quehacer:
      Exit Sub

End Sub
```

```
'Impresora
Private contar As Integer
Private Esp1 As Integer, Esp2 As Integer, Esp3 As Integer, Esp4 As Integer
'Datos del Proyecto

Public Sub Imprimir_Resul()

   'Voltaje
   Call Imprimir_ResulV

End Sub
Public Sub Imprimir_ResulV()

   Dim contar_aux As Integer: Controlador_dat = False
   For contar = 1 To Contador1
      If contar Mod 60 = 1 Then
         For j = 1 To 3
            Printer.Print ""
         Next
         Printer.Print Spc(60); "Caidas de Voltaje"
         Printer.Print Spc(30);
         For i = 5 To 75
            Printer.Print "_";
         Next
         Printer.Print ""
         Printer.Print ""
         Printer Print Spc(38); "Nodo"; Spc(37); "Caida de Voltaje(%)"
         Printer.Print Spc(30);
         For i = 5 To 75
            Printer.Print "_";
         Next
         Printer.Print ""
      End If
      'espacios
      If contar < 10 Then
         Esp1 = 40
      ElseIf contar < 100 Then
         Esp1 = 39
      ElseIf contar < 1000 Then
         Esp1 = 38
      ElseIf contar < 10000 Then
         Esp1 = 37
      End If
```

```
    Printer.Print Spc(Esp1); contar; Spc(45); Voltaje1(contar)
    If contar Mod 60 = 0 Then
    '----última linea----'
    Printer.Print ""
    Printer.Print Spc(30);
    For j = 5 To 75
       Printer.Print "_";
    Next
    '------------'
    Paginas = Paginas + 1
    Printer.CurrentX = 900
    Printer.CurrentY = 13500
    For j = 1 To 3
       Printer.Print
    Next
    Printer.Print Spc(72); "Pag "; Paginas + 133
    Printer.NewPage
    End If
Next

If Contador1 Mod 60 <> 0 Then

    '-----ultima linea-----'
    Printer.Print ""
    Printer.Print Spc(30);
    For j = 5 To 75
       Printer.Print "_";
    Next
    '----------------'
    If (ElementoC + Contador1) < 30 Then
       For contar_aux = 1 To ElementoC
       If contar_aux = 1 Then
          For j = 1 To 3
             Printer.Print ""
          Next
          Printer.Print Spc(60); "Estudio de las Redes"
          Printer.Print Spc(20);
          For i = 5 To 95
             Printer.Print "_";
          Next
          Printer.Print ""
          Printer.Print ""
```

```
            Printer.Print Spc(28); "Red"; Spc(23); "Perdidas(w)"; Spc(20); "Máxima
Caida de Voltaje(%)"
            Printer.Print Spc(20);
            For i = 5 To 95
            Printer.Print "_";
            Next
            Printer.Print ""
          Else
            Esp1 = 30 - Tamaño(contar_aux - 1)
            Esp2 = 30 - Tamaño(Perdidas1(contar_aux - 1))
            Esp3 = 35 - Tamaño(Mayor1(contar_aux - 1))
            Printer.Print Spc(Esp1); contar_aux - 1; Spc(Esp2); Perdidas1(contar_aux -
1); Spc(Esp3); Mayor1(contar_aux - 1)
            Controlador_dat = True
          End If
        Next
        For j = 1 To 3
          Printer.Print ""
        Next
        Printer.Print Spc(20);
        For i = 5 To 95
          Printer.Print "_";
        Next
      End If
        Paginas = Paginas + 1
        Printer.CurrentX = 900
        Printer.CurrentY = 13500
        For j = 1 To 3
          Printer.Print
        Next
        Printer.Print Spc(72); "Pag="; Paginas + 133
        Printer.NewPage

      End If

End Sub
Private Function Tamaño(Valor As Variant) As Integer
    Dim Cadena As String
    Cadena = CStr(Valor)
    Tamaño = Len(Cadena)
End Function

Public Sub Gestiona_Tamaño()
```

```
Form5.Width = 0.75 * (Form1.Width): Form5.Height = 0.5 * (Form1.Height)
Form5.Top = 0.25 * (Form1.Height): Form5.Left = 0.125 * (Form1.Height)
Form5.SSTab1.Width = Form5.Width: Form5.SSTab1.Height = Form5.Height
Form5.Command1.Left = (Form5.Width - Form5.Command1.Width) * 0.5:
Form5.Command2.Left = Form5.Width * 0.8
Form5.Command1.Top = (Form5.Height - Form5.Command1.Height) * 0.8:
Form5.Command2.Top = (Form5.Height - Form5.Command2.Height) * 0.8
Form5.Command3.Left = (Form5.Width - Form5.Command1.Width) * 0.82:
Form5.Command3.Top = (Form5.Height - Form5.Command1.Height) * 0.3

End Sub

Public Sub Datos_Proy()


  For contar = 1 To ElementoC
    If contar = 1 Then
      For j = 1 To 3
        Printer.Print ""
      Next
      Printer.Print Spc(65); "Redes del Proyecto"
      Printer.Print Spc(10);
      For i = 15 To 145
        Printer.Print "_";
      Next
      Printer.Print ""
      Printer.Print ""
      Printer.Print Spc(28); "Red"; Spc(20); "Nodo de Referencia"; Spc(16);
"Conductor"; Spc(18); "Factor de Potencia"
      Printer.Print Spc(10);
      For i = 15 To 145
        Printer.Print "_";
      Next
      Printer.Print ""
    Else
      Esp1 = 30 - Tamaño(contar - 1)
      Esp2 = 30 - Tamaño(Referencia(contar - 1))
      Esp3 = 30 - Tamaño(COND1(contar - 1))
      Esp4 = 30 - Tamaño(fp1(contar - 1))
      Printer.Print Spc(Esp1); contar - 1; Spc(Esp2); Referencia(contar - 1);
Spc(Esp3); COND1(contar - 1); Spc(Esp4); fp1(contar - 1)
    End If
```

```
        'espacios
        If contar Mod 60 = 0 Then
        '----ultima linea----'
        Printer.Print ""
        Printer.Print Spc(10);
        For j = 15 To 145
           Printer.Print "_";
        Next
        '------------'
        Paginas = Paginas + 1
        Printer.CurrentX = 900
        Printer.CurrentY = 13500
        For j = 1 To 3
           Printer.Print
        Next
        Printer.Print Spc(72); "Pag. ", Paginas + 133
        Printer.NewPage
     End If
  Next
     If Contador1 Mod 60 <> 0 Then

        '----ultima linea----'
        Printer.Print ""
        Printer.Print Spc(10);
        For j = 15 To 145
           Printer.Print "_";
        Next
        '------------'

        Paginas = Paginas + 1
        Printer.CurrentX = 900
        Printer.CurrentY = 13500
        For j = 1 To 3
           Printer.Print
        Next
        Printer.Print Spc(72); "Pag. ", Paginas + 133
        Printer.NewPage

     End If

End Sub

Public Sub Imprimir_Mayor()
```

```
For contar = 1 To 1.Elemento(
    If contar = 1 Then
      For j = 1 To 3
        Printer.Print ""
      Next
      Printer.Print Spc(60); "Estudio de las Redes"
      Printer.Print Spc(70);
      For i = 5 To 95
        Printer.Print "_";
      Next
      Printer.Print ""
      Printer.Print ""
      Printer.Print Spc(28); "Red"; Spc(23); "Perdidas(w)"; Spc(20); "Máxima
Caida de Voltaje(%)"
      Printer.Print Spc(20);
      For i = 5 To 95
        Printer.Print "_";
      Next
      Printer.Print ""
      Else
        Esp1 = 30 - Tamaño(contar - 1)
        Esp2 = 30 - Tamaño(Perdidas1(contar - 1))
        Esp3 = 35 - Tamaño(Mayor1(contar - 1))
        Printer.Print Spc(Esp1); contar - 1; Spc(Esp2); Perdidas1(contar - 1);
Spc(Esp3); Mayor1(contar - 1)
      End If
    'espacios

    If contar Mod 60 = 0 Then
    '---última linea ---'
    Printer.Print ""
    Printer.Print Spc(20);
    For j = 5 To 95
      Printer.Print "_";
    Next
    '------------'
    Paginas = Paginas + 1
    Printer.CurrentX = 900
    Printer.CurrentY = 13500
    For j = 1 To 3
      Printer.Print
    Next
```

```
        Printer.Print Spc(72); "Pag "; Paginas + 133
        Printer.NewPage
        End If
    Next
    If Contador1 Mod 60 <> 0 Then

        '-----ultima linea-----'
        Printer.Print ""
        Printer.Print Spc(20);
        For j = 5 To 95
            Printer.Print "_";
        Next
        '-----------------'

        Paginas = Paginas + 1
        Printer.CurrentX = 900
        Printer.CurrentY = 13500
        For j = 1 To 2
            Printer.Print
        Next
        Printer.Print Spc(72); "Pag "; Paginas + 133
        Printer.NewPage

    End If
End Sub

Public Sub Imprimir_Dib()

    Printer.Orientation = 2
    If Mantener_EscB = False Then
        Call Cambia_escala(Printer)
        Call Dibuja_Pro1(Printer)
    Else
        Printer.Scale (izquierda1, tope1)-(ancho1, alto1)
        Call Dibujo_esp1(Printer)
        Call Numerar_nodos(Printer)
    End If
    Call Dibujar_dxf(Printer)
    Call Escribir_Nombre(Printer)
    Call Cuadro(Printer)

End Sub
Public Sub Dibuja_Pro1(Objeto As Object)
```

```
    Dim Mayor As Double, Bandera As Boolean, j As Long, I As Long
    Bandera = False: yy = 0: yy1 = 0: xx = -(0.001 * Objeto.ScaleWidth): xx1 = -
(0.0025 * Objeto.ScaleWidth): radio = 0.1 * (Objeto.ScaleWidth)
    For j = 1 To Contador
        For I = 1 To ReferenciaC
            If Referencia(I) = NI1(j) Then
                Bandera = True
                Exit For
            End If
        Next
        If Bandera = True Then
            Objeto.PSet (XPosi1(j), YPosi1(j)): Objeto.Circle (XPosi1(j) + xx, YPosi1(j)
+ yy), radio * 0.008: Objeto.Circle (XPosi1(j) + xx1, YPosi1(j) + yy1), radio * 0.008
Bandera = False
        Else
            Objeto.PSet (XPosi1(j), YPosi1(j))
        End If
        Objeto.Line (XPosi1(j), YPosi1(j))-(XPosf1(j), YPosf1(j)), Objeto.Circle
(XPosf1(j), YPosf1(j)), radio * 0.0027

    Next

End Sub
Sub Dibujo_esp1(Objeto As Object)

    For i = 1 To N
        If i = 1 Then
            Objeto.PSet (XPosi(i), YPosi(i)): Objeto.Circle (XPosi(i) + xx, YPosi(i) +
yy), radio * 0.008: Objeto.Circle (XPosi(i) + xx1, YPosi(i) + yy1), radio * 0.008
        Else
            Objeto.PSet (XPosi(i), YPosi(i))
        End If

        Objeto.Line (XPosi(i), YPosi(i))-(XPosf(i), YPosf(i)): Objeto.Circle (XPosf(i),
YPosf(i)), radio * 0.0027
    Next

End Sub


Public Sub Escribir_Nombre(Objeto As Object)
```

```
    Dim X11 As Single, Y11 As Single, fuente As Single
    fuente = Objeto.FontSize
    X11 = Objeto.ScaleWidth: Y11 = (Objeto.ScaleHeight)
    If X11 >= 1000 Then
       X11 = (X11 / 1000) * 2
    ElseIf X11 >= 100 Then
       X11 = 4
    Else
       X11 = 2
    End If
    If Y11 >= 1000 Then
       Y11 = (Y11 / 1000) * 2
    ElseIf (Y11 >= 100) Then
       Y11 = 4
    Else
       Y11 = 2
    End If
    Objeto.PSet (Objeto.ScaleLeft + X11, Objeto.ScaleTop + Y11)
    Objeto.Print "Nombre del Proyecto ", Spc(1), Nombre_Pro
    If Mantener_Enc3 = False Then
       'Objeto.Print "Vista General "
    Else
       Objeto.PSet (Objeto.ScaleLeft + X11, Objeto.ScaleTop + 2 * Y11)
       Objeto.Print "Nombre del Sector ", Spc(1), Nombre_Matriz
    End If

End Sub


Public Sub Cuadro(Objeto As Object)

    Objeto.DrawWidth = 4
    Objeto.FillStyle = 1
    Objeto.Line (Objeto.ScaleLeft, Objeto.ScaleTop)-(Objeto.ScaleLeft +
Objeto.ScaleWidth, Objeto.ScaleTop + Objeto.ScaleHeight), QBColor(0), B
    Objeto.DrawWidth = 1

End Sub
```

2.3 EJEMPLO DE UN PROBLEMA.

En el siguiente ejemplo se plantea una red sencilla con el propósito de comparar los resultados de caída de tensión con los que arroja el programa VBCAD.

Figura_2.2. Ejemplo de un Problema.

Tensión Nominal de Línea a Línea de Circuito: 208 Volt.

Factor de Potencia (Fp): 0,95 atrasado.

Longitud de los Tramos:

Tramo 1 (T1): 80 mts.

Tramo 2 (T2): 100 mts.

Tramo 3 (T3): 150 mts.

Tramo 4 (T4): 117 mts.

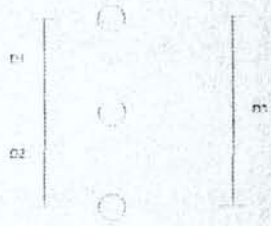
Conductor:

Conductor de cobre calibre 4 (Cu-4).

Según Tabla de Conductores (Apéndice A):

R= 0,8451 Ohm/Km

$X = 0{,}0754 * \mathrm{Ln}\left(\frac{DMG}{ds}\right)$ Ohm / Km a 60 Hz.



D1 = 20 cm. ; D2 = 20 cm. ; D3 = 40 cm.

$DMG = \sqrt[3]{20 * 20 * 40}$

DMG = 25,1984 cm.

En donde: ds = K * diametro del (Cu-4)

K = 0,363 para el Cu-4 (Apéndice A).

Luego: ds = 0,363 * 0,518 = 0,1880 cm.

Por lo que: $X = 0{,}0754 * \mathrm{Ln}\left(\frac{25{,}1984}{0{,}1880}\right) = 0{,}3693$ Ohm/Km.

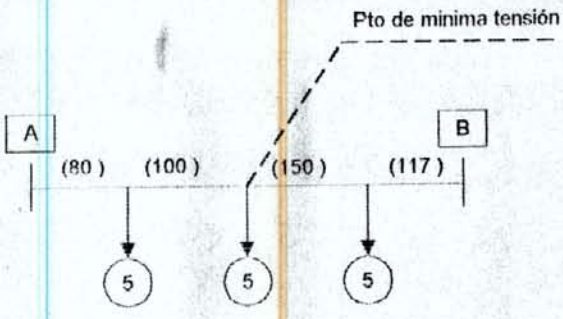
Calculo de la Constante de Distribución (Kd):

$$Kd = \frac{(R * \cos\theta + X * sen\theta)}{(10 * KV^2)}$$

$$Kd = \frac{(0,8451*0,95+0,3693*0,31)*10^{-3}}{10*(0,208)^2}$$

$$Kd = 2,12*10^{-3}.$$

Análisis de la red:



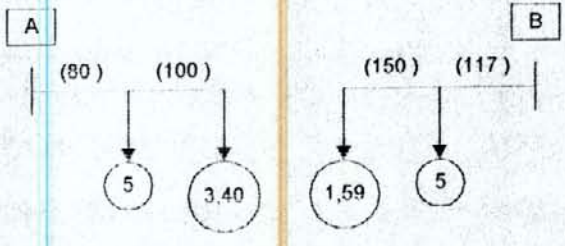
$$KVA_A = \frac{(117*5+267*5+367*5)}{447}$$

$$KVA_A = 8,40.$$

$$KVA_B = \frac{(80*5+180*5+330*5)}{447}$$

$$KVA_B = 6,59.$$

Por lo que:



Calculo de Caída de Tensión:

$\%V = Kd * (80 * 5 + 180 * 3,40)$

$\%V = 2,12 * 10^{-3} * (1012)$

$\%V = 2,14$.

CORRIDA DEL EJEMPLO EN EL PROGRAMA

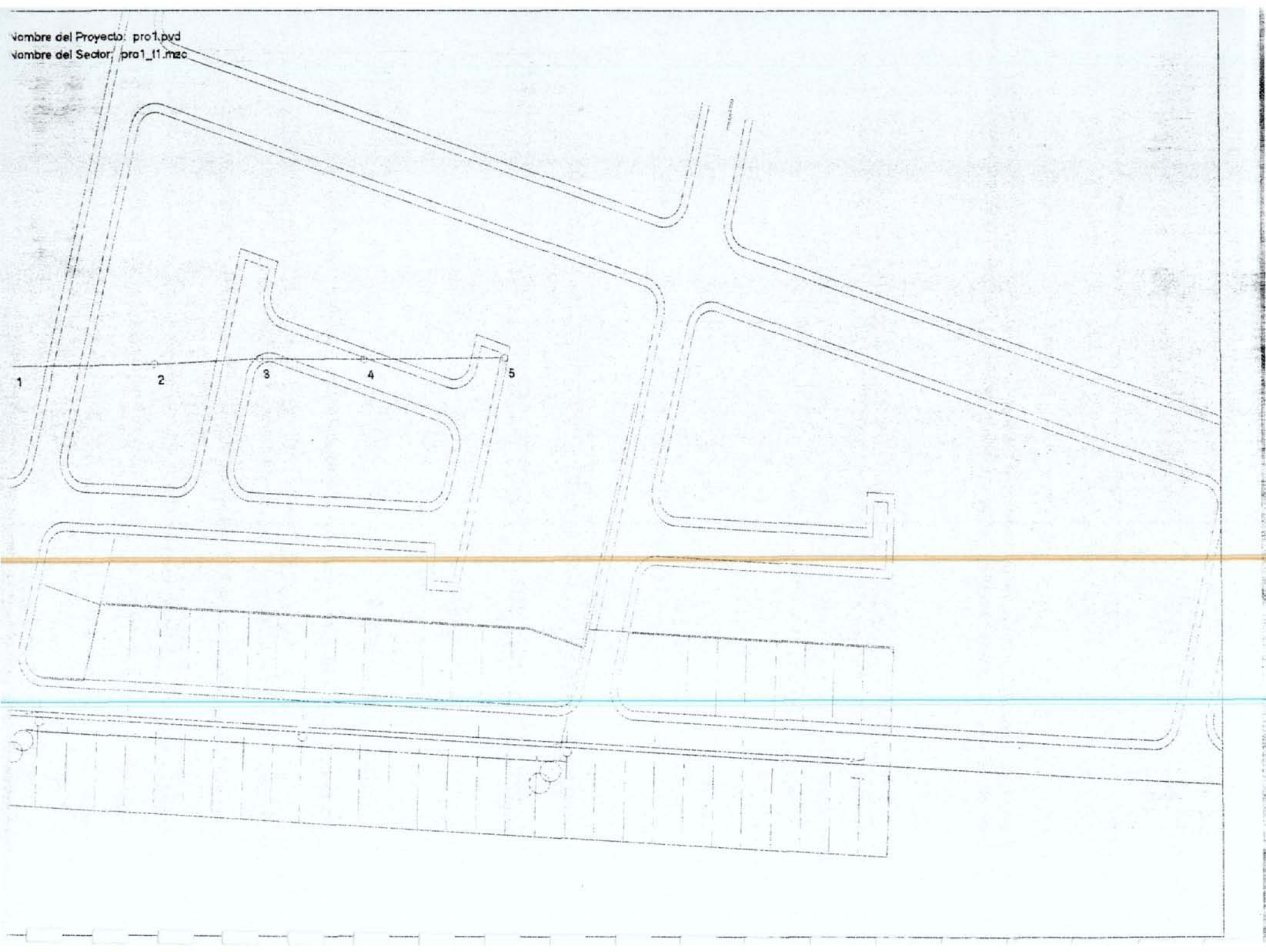
Red	Nodo de Referencia	Conductor	Factor de Potencia
1	1	CU-4	0,9

Caidas de Voltaje

Nodo	Caida de Voltaje(%)
1	0
2	1,7045
3	2,663
4	3,302
5	3,728

Estudio de la red

Red	Perdidas(w)	Máxima Caida de Voltaje(%)
1	523,36	3,75


Nombre del Proyecto: pro1.pyd
Nombre del Sector: pro1_l1.mzc
1
2
3
4
5

CAPÍTULO 3

MANUAL DEL USUARIO

MANUAL DEL USUARIO.

3.1 REQUISITOS PARA EJECUTAR VBCAD.

La lista siguiente describe los requisitos de software y hardware mínimos que requiere el sistema para ejecutar VBCAD con éxito.

· Versión 4.00.1111 de Microsoft MS-DOS o posterior.

· Una computadora personal con un procesador Intel Pentium (o superior), 26 megabytes (MB) de espacio libre de disco.

· Un adaptador de pantalla compatible con Windows.

· Una impresora compatible con Windows si desea imprimir informes.

· Un Mouse compatible con Windows. Aunque no sea necesario, es muy recomendable utilizarlo para aprovechar al máximo la sencilla interfaz grafica de Windows.

· AutoCAD 14.0 como mínimo.

3.2 FUNDAMENTOS DE VBCAD.

Se debe tener el conocimiento de algunos términos básicos para trabajar con VBCAD los cuales estarán cubiertos si se ha trabajado con Windows anteriormente.

Teniendo en cuenta que VBCAD es un programa realizado en Visual Basic 6.0 específicamente, trabaja bajo ambiente Windows de allí su manejo es muy similar al de éste.

Al empezar a trabajar con VBCAD el usuario podrá darse cuenta que es un software de fácil manejo, un ambiente amigable de trabajo y con resultados óptimos y rápidos en el área de monitoreo y diseño de sectores de baja tensión ahorrando largas horas de dedicación y estudios.

3.3 INICIO DE VBCAD.

Al iniciar VBCAD aparecerá una ventana de presentación (figura_3.1), nombre que se le ha colocado por ser la primera ventana en visualizarse al iniciar VBCAD, la cual estará activada un tiempo muy pequeño y luego desaparecerá quedando activada la ventana principal (Figura_3.2), se denomina ventana activada a la ventana que tiene el enfoque. En la ventana principal se encuentra presente ciertos menú los cuales los cuales representan los menú generales del programa por lo que se ha denominado a esta ventana Menú Principal y se explicaran detalladamente más adelante, un menú incluye ciertos comandos o acciones que puede ejecutar VBCAD y se encuentran ubicados debajo del nombre de la ventana. Ahora se muestran las ventanas de las que se ha venido hablando anteriormente, primero se muestra la ventana de presentación y luego la ventana principal.

Ventana de Presentación:

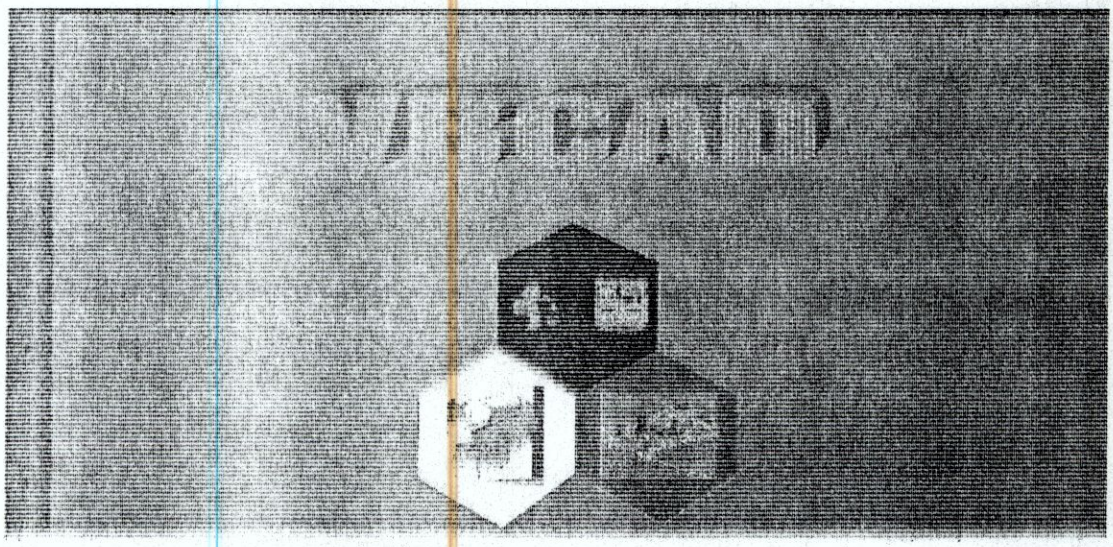


Figura_3.1. Ventana de Presentación.

Ventana Principal.

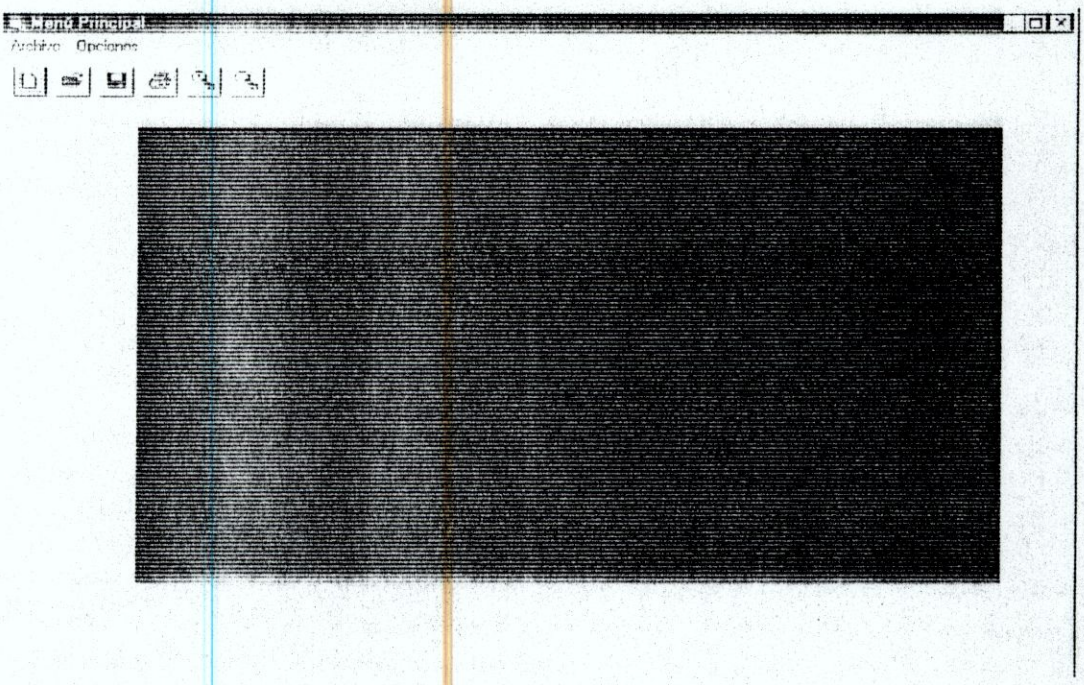


Figura_3.2. Ventana Principal.

Iconos o Botones presentes en la Ventana Principal.

Icono	Descripción
	Representa el acceso de nuevos datos.
	Representa la apertura de un archivo ya existente.
	Representa como guardar un archivo.
	Representa la forma de imprimir un informe.
	Representa un Zoom "agrandamiento" del recuadro formado por los puntos que sean seleccionados con el botón izquierdo del Mouse.
	Representa el volver a la vista principal del grafico.

"Es importante destacar que estos iconos o botones ejecutan la misma acción que los comandos con el mismo nombre dentro de cualquier menú."

En el siguiente Diagrama de Flujo (Figura_3.3) se representa la estructura de VBCAD.

Figura_3.3. Estructura de VBCAD.

Para seleccionar un nombre de un menú o de un comando, haga clic una vez en el nombre del menú o comando correspondiente.

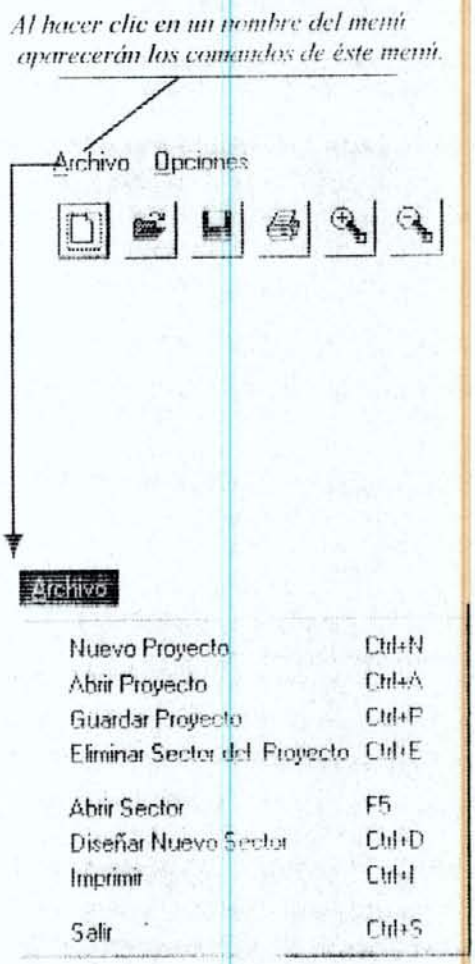


Figura_3.4. Despliegue del Menú Archivo.

Al hacer clic aquí aparecerán los siguientes comandos.

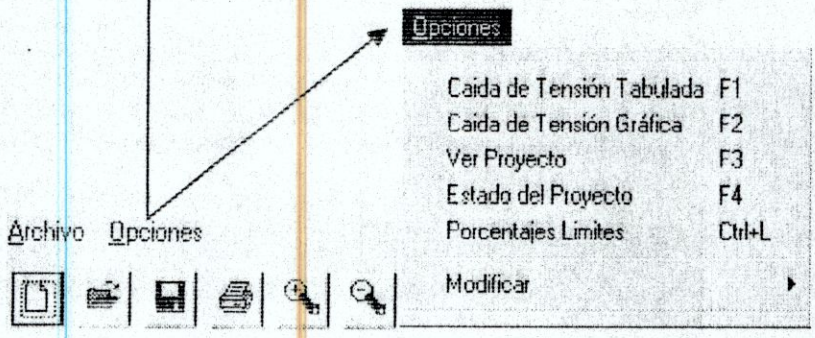


Figura_3.5. Despliegue del Menú Opciones.

3.4 DESPLIEGUE DE UN MENU.

Como anteriormente se había mencionado para seleccionar un nombre de menú o comando, haga clic en el nombre o comando correspondiente.

Se debe tener en cuenta que siempre que se hable de hacer clic se refiere hacerlo con el Mouse , desde el teclado con la tecla "Enter" una vez seleccionada o con las función "Alt" y luego la letra subrayada en el menú.

3.4.1 Archivo.

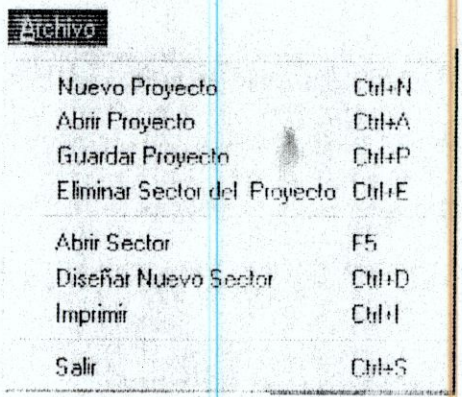


Figura_3.6. Menú Archivo.

Al hacer clic en el menú "*Archivo*" aparecerán los comandos presentes en éste menú.

Nuevo Proyecto o Ctrl + N ⟶ Representa crear o accesar un nuevo archivo.

Abrir Proyecto o Ctrl + A ⟶ Representa abrir un archivo ya existente.

Guardar Proyecto o Ctrl + P ⟶ Representa guardar el archivo.

Eliminar Sector del Proyecto o Ctrl + E ⟶ Representa eliminar un Sector de un Proyecto dado.

Abrir Sector o F6 ⟶ Representa abrir un Sector de un proyecto dado.

Diseñar Nuevo Sector o Ctrl + D ⟶ Representa crear o diseñar un nuevo sector.

Imprimir o Ctrl + I ⟶ Representa imprimir un archivo o informe.

Salir o Ctrl + S ⟶ Representa salir del programa.

" Se debe tomar en cuenta que cada proyecto es un archivo distinto pero con una misma extensión la cual es (.pvd) y estos proyectos a la vez cuentan con un numero determinado de sectores los cuales tienen una misma extensión la cual es (*.mzc) ".*

3.4.1.1 Nuevo Proyecto.

Al hacer clic en el comando con el nombre *"Nuevo Proyecto"* aparecerá el siguiente Cuadro de Dialogo I (Figura_3.7) con una extensión (*.dxf.).

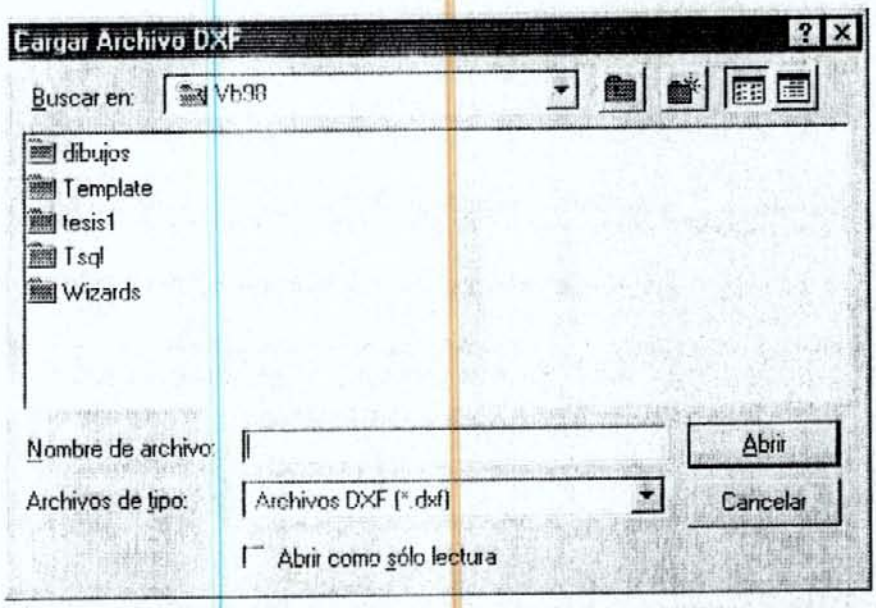


Figura_3.7. Cuadro de Dialogo I.

"Se debe tener en cuenta que VBCAD presenta una interfaz entre Visual Basic 6.0 y AutoCAD 14.0 específicamente en donde una rutina realizada en Visual Basic se encarga de leer los archivos DXF exportados desde AutoCAD por ende los únicos archivos con los que VBCAD puede trabajar son los archivos realizados en AutoCAD 14.0 y exportados como archivo DXF."

Para seleccionar un archivo en el Cuadro de Dialogo se debe colocar con el Mouse encima del archivo correspondiente y hacer clic con el botón izquierdo del Mouse, al realizar este proceso se carga la ventana cuyo nombre se ha denominado Ventana Diseño (Figura_3.8) en donde aparecerá el dxf seleccionado, y que se muestra a continuación.

Ventana Diseño.

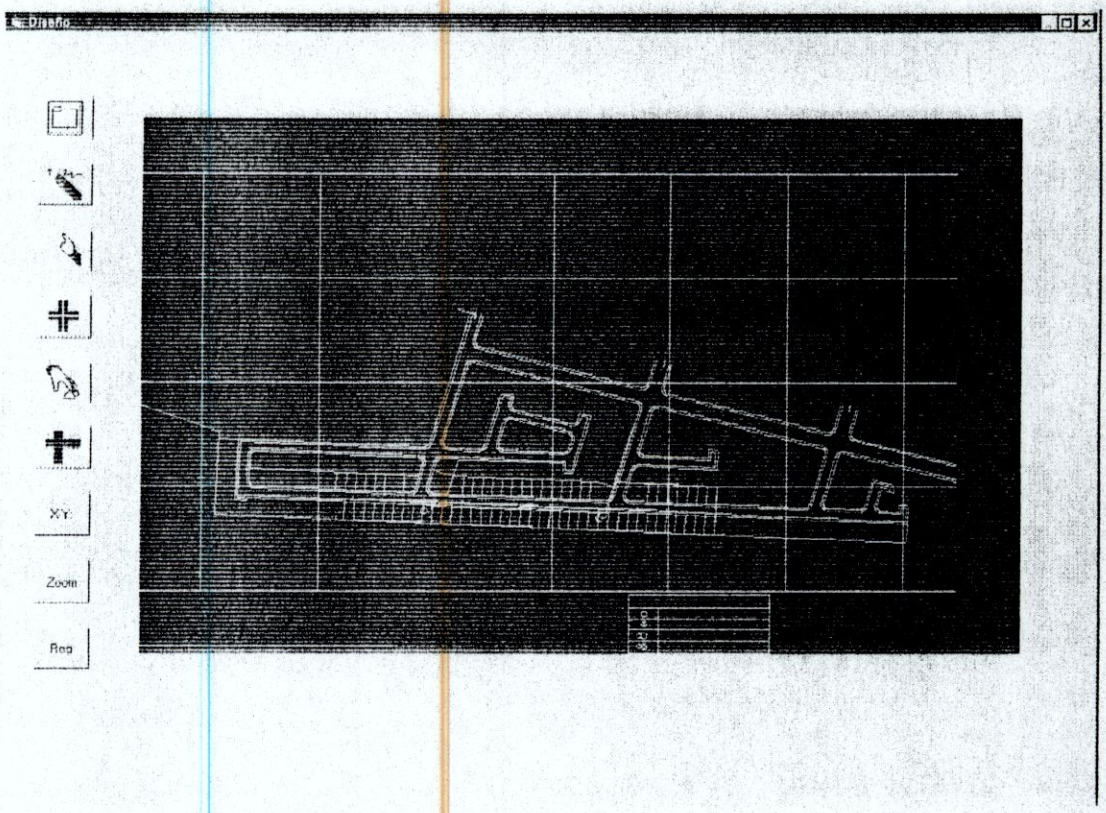
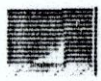
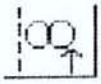


Figura_3.8. Ventana Diseño.

Iconos o Botones presentes en la Ventana Diseño.

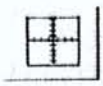
Icono	Descripción
	Representa volver a la ventana Menu Principal.
	Representa borrar uno o varios elementos.
	Representa la selección de uno o varios elementos.



Representa mover el transformador en cualquier dirección.

Representa la llamada a la Ventana Conductor para la selección del mismo, y la finalización del trabajo en la Ventana Diseño.

Representa el acceso, desde el teclado, de la distancia del tramo con el que se esta trabajando.



Representa la lecturas de las coordenadas X , Y de cualquier punto en el que se ubique el Mouse.

Representa un Zoom "agrandamiento" del recuadro formado por los puntos que sean seleccionados con el botón izquierdo del Mouse.

Representa el volver a la vista principal del grafico.

"La selección (clic) de todos los botones se realiza con el botón izquierdo del Mouse a menos que se exprese lo contrario."

"Se debe tener en cuenta que los tres últimos botones de la Ventana Diseño son los únicos que actúan en los dxf o plantilla como se le ha denominado por servir de piso o contenedor de las redes, por lo que no es posible seleccionar o borrar elementos de un dxf. Además el botón Mover Transformador después de ser seleccionado se coloca encima del punto deseado con el botón izquierdo del Mouse y luego se libera con el botón derecho del mismo. En cuanto al botón Zoom, el primer clic que se haga

con el Mouse debe ser en la parte superior izquierda de la zona donde se realice el zoom, el segundo clic debe ser en la parte inferior derecha, dando una forma de recuadro a la zona en cuestión."

"El proceso en el que se dibujan los tramos de las redes en la Ventana Diseño se ejecuta bajo el evento Mouse de Visual Basic por lo que es de suma importancia que los tramos se grafiquen de una forma pausada entre uno y otro, dando tiempo a que el programa reconozca y guarde las variables necesarias para un funcionamiento optimo y de esta forma evitar posibles errores. No obstante en caso de que ocurra algún evento fuera de lugar se recomienda seleccionar el botón Borrar y comenzar de nuevo."

Selección de Botón Digitalización de la Red.

Al seleccionar el botón digitalización de la red, aparecerá un Cuadro de Mensaje I (Figura 3.9) en donde se expresa el siguiente paso, que es la escogencia del conductor de la red, al hacer el clic en aceptar se carga la Ventana Conductor.

Figura_3.9. Cuadro de Mensaje I.

Ventana Conductor.

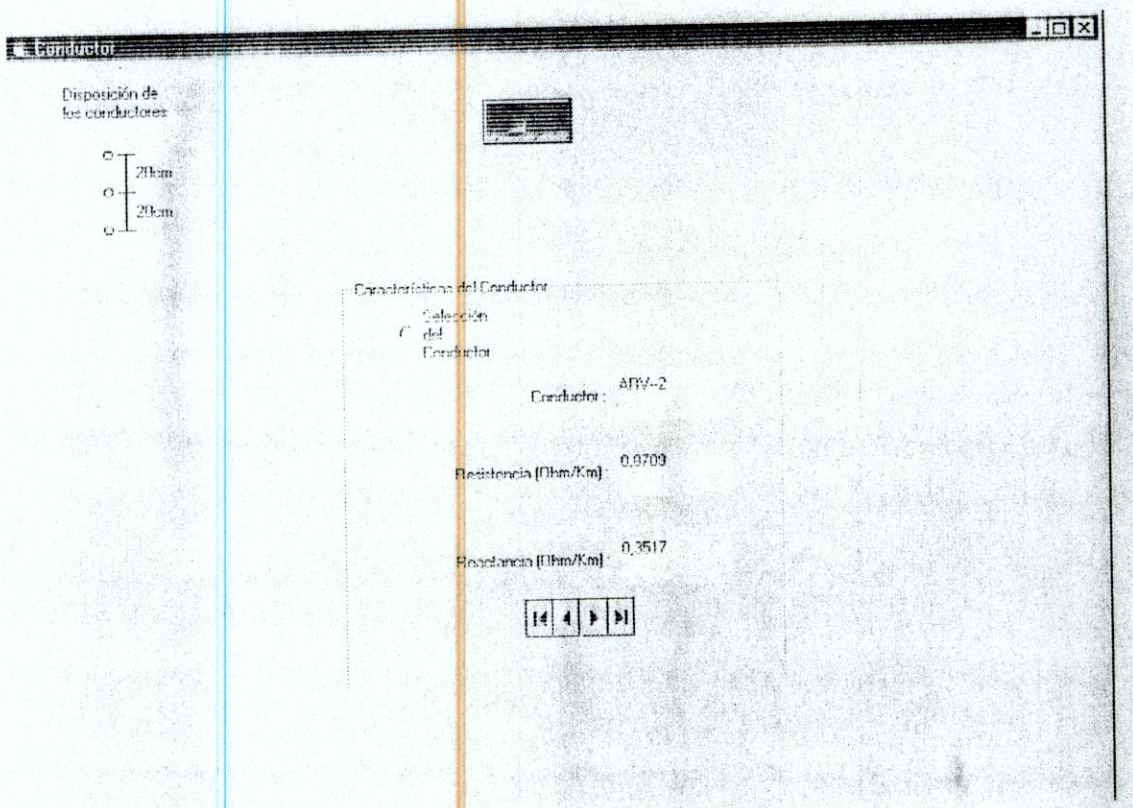


Figura_3.10. Ventana Conductor.

Elementos de la Ventana Conductor.

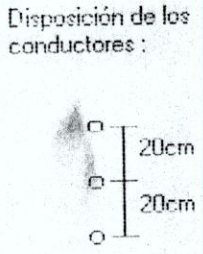


Figura_3.11. Disposición de los Conductores.

Representa como su nombre lo indica la Disposición de los Conductores con la que se trabaja en VBCAD.

Figura_3.12. Botón Menu Principal.

Representa como su nombre lo indica volver al menú principal al hacer clic con el Mouse sobre él.

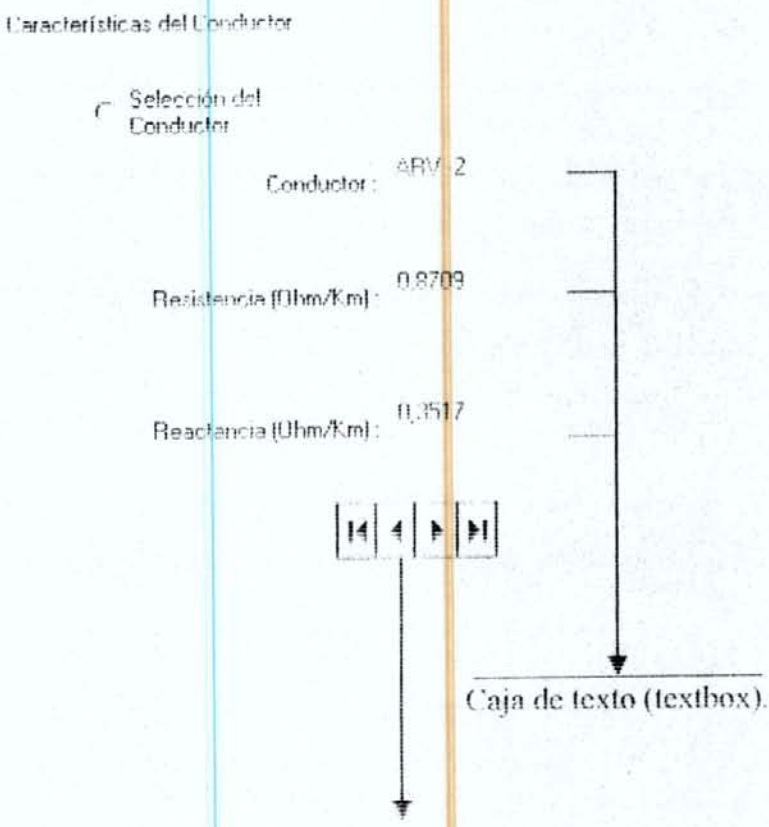


Figura_3.13. Características del Conductor.

Representa la Base de Datos del programa y consta de cuatro botones los cuales se explicaran a continuación, y dependiendo de cual sea seleccionado aparecerá un nuevo conductor con sus respectivas Resistencias y Reactancias en las cajas de texto correspondientes.

Botones de la Base de Datos:

Icono	Descripción
	Representa desplazarse al siguiente elemento en la Base de Datos.
	Representa desplazarse al ultimo elemento en la Base de Datos.
	Representa desplazarse al elemento anterior en la Base de Datos.
	Representa desplazarse al ultimo elemento en la Base de Datos.

Selección del Conductor

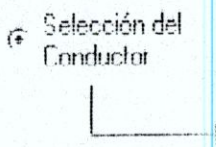


Figura_3.14. Botón Opción.

Representa el objeto Botón de Opción (Option Button) muy utilizado en Visual Basic y cuya finalidad es la escogencia definitiva del conductor de la red y que se activa con un clic sobre el circulo como lo muestra la Figura_3.14.

Al seleccionar el Botón de Opción, aparecerá un Cuadro de Mensaje II (Figura_3.15) en donde se expresa el siguiente paso, que es la escogencia de las cargas de la red, al hacer el clic en aceptar aparece la Ventana Cargas (Figura_3.16).

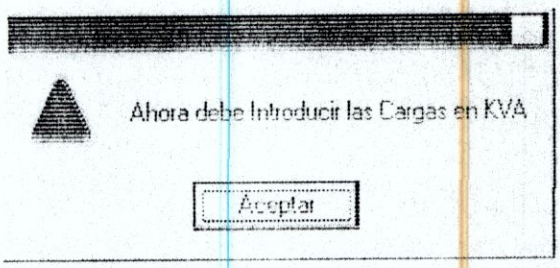


Figura_3.15. Cuadro de Mensaje II.

Ventana Cargas.

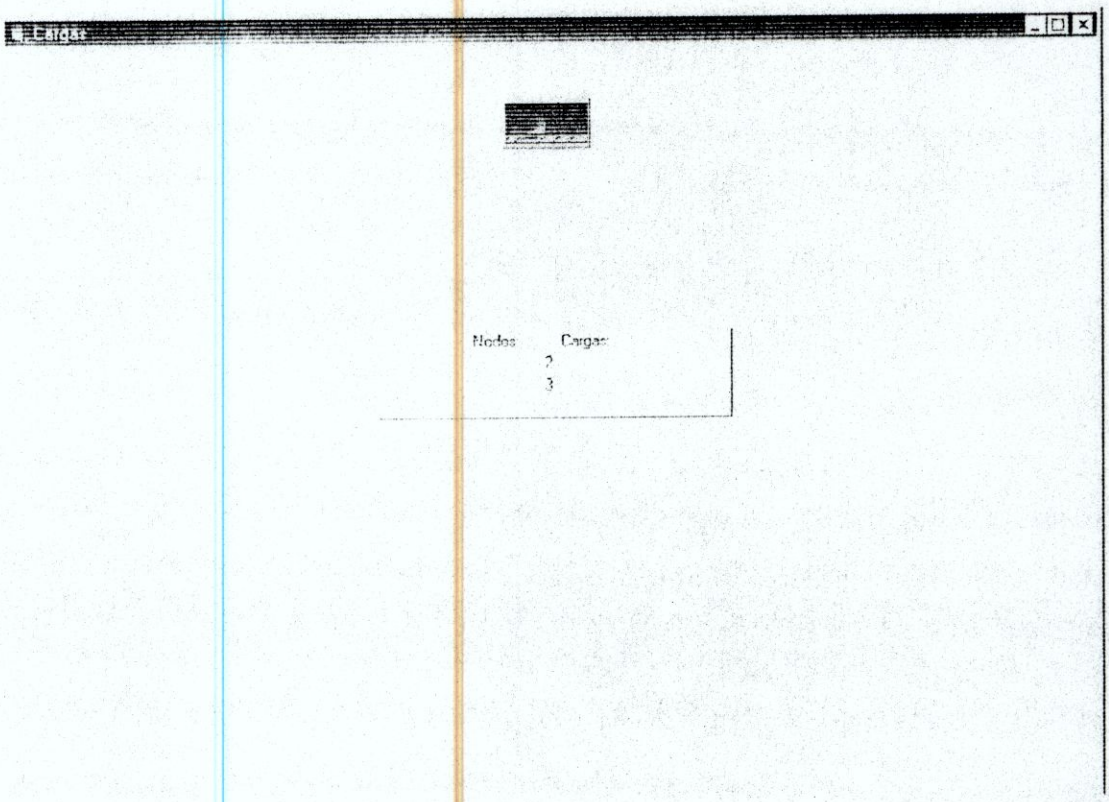


Figura_3.16. Ventana Cargas.

Elementos de la Ventana Cargas.

Figura_3.17. Botón Menú Principal.

Representa el mismo Botón y función que el de la Figura_3.12.

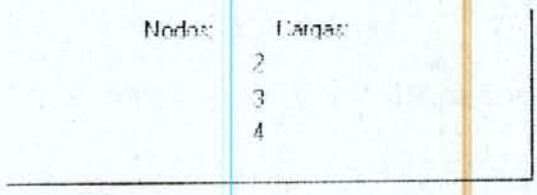


Figura_3.18. Objeto MsflexGrid.

Representa el objeto MsflexGrid en Visual Basic muy parecido en cuanto apariencia y forma de trabajo a una hoja Excel, utilizado en el programa para el acceso de cargas, la cual se realiza nodo por nodo, y en donde se encuentran Cuadros de Mensajes, utilizados como forma de alerta en caso de ejecutar una acción errónea al momento de accesar las cargas, y que se explicaran a continuación.

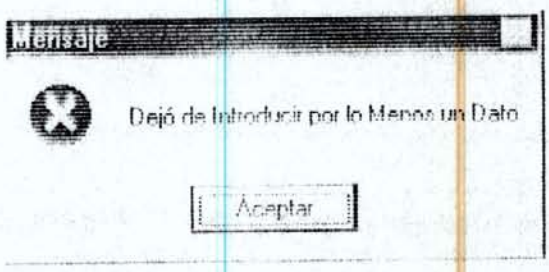


Figura_3.19. Cuadro de Mensaje III.

Representa como el mensaje lo indica que no es posible abandonar la Ventana Cargas hasta que no se incluyan todas las cargas, evitando errores en el momento de evaluar la red

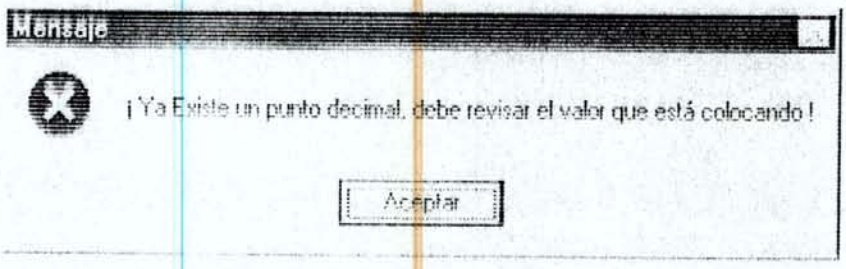


Figura_3.20. Cuadro de Mensaje IV.

Representa como el mensaje lo indica que existe un punto decimal en la cifra que esta colocando, por lo que se debe hacer clic en el Botón "Aceptar",verificar y seguir incluyendo las cargas.

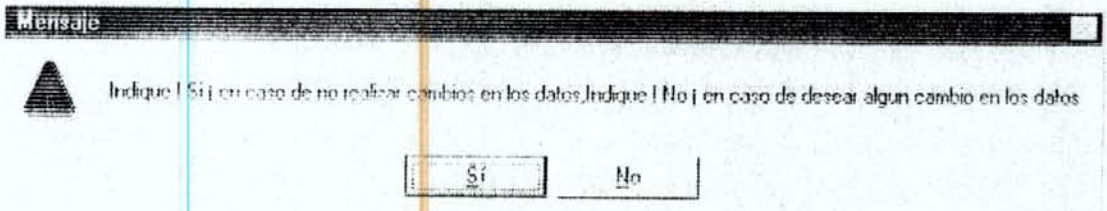


Figura_3.21. Cuadro de Mensaje V.

Representa como el mensaje lo indica que tiene una alternativa de cambiar alguna carga antes de proseguir con la corrida del programa, una vez activado el clic en el Botón "Si", aparecerá un nuevo Cuadro de Mensaje VI (Figura_3.22) que a diferencia de los anteriores permite el acceso de un dato, que en este caso es el Factor de Potencia "Fp".

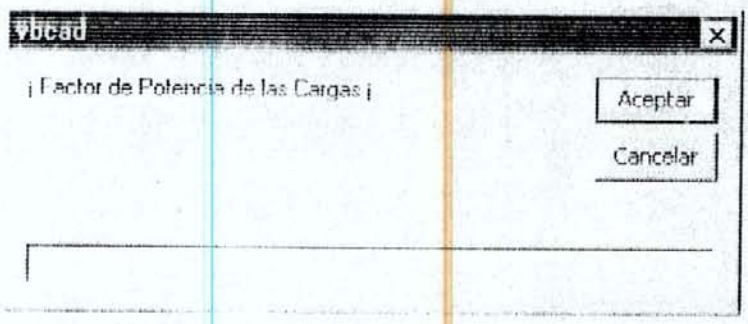


Figura_3.22. Cuadro de Mensaje VI.

- Representa como el mensaje lo indica la inclusión del Factor de Potencia, que por tratarse de cargas muy similares se toma igual para todas estas, con la salvedad que esta inclusión de Fp esta provisto con otros mensajes de alerta en caso de un error al momento de accesarlo. Además la inclusión de Fp esta provisto de un mecanismo que detecta si el Fp insertado es mayor que uno ">1", por lo que el mecanismo toma las primeras dos o una cifra según sea el caso, colocadas después del punto decimal y agrega un cero "0" a la izquierda del punto decimal, convirtiendo la cifra en un valor menor que uno "<1".

Mensajes de Alerta en el Fp.

Figura_3.23. Cuadro de Mensaje VII.

→ Representa como el mensaje lo indica que falta el punto decimal en la cifra que se acaba de insertar, ya que el Fp es un valor decimal y menor que uno como se había mencionado anteriormente.

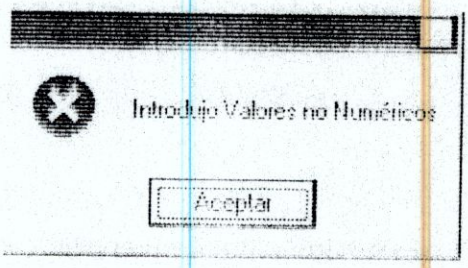


Figura_3.24. Cuadro de Mensaje VIII.

→ Representa como lo indica el mensaje, en donde se trato de incluir valores no numéricos en el Fp, o más de un punto decimal.

Una vez que se haya insertado el factor de potencia correctamente quedara seleccionado automáticamente el Botón "Menú Principal" (Figura_3.17) él cual al activarlo, se carga la Ventana Principal.

3.4.1.2 Abrir Proyecto.

Al hacer clic en el comando con el nombre *"Abri Proyecto"* aparecerá el siguiente Cuadro de Dialogo II (Figura_3.25) con una extensión (*.pvd).

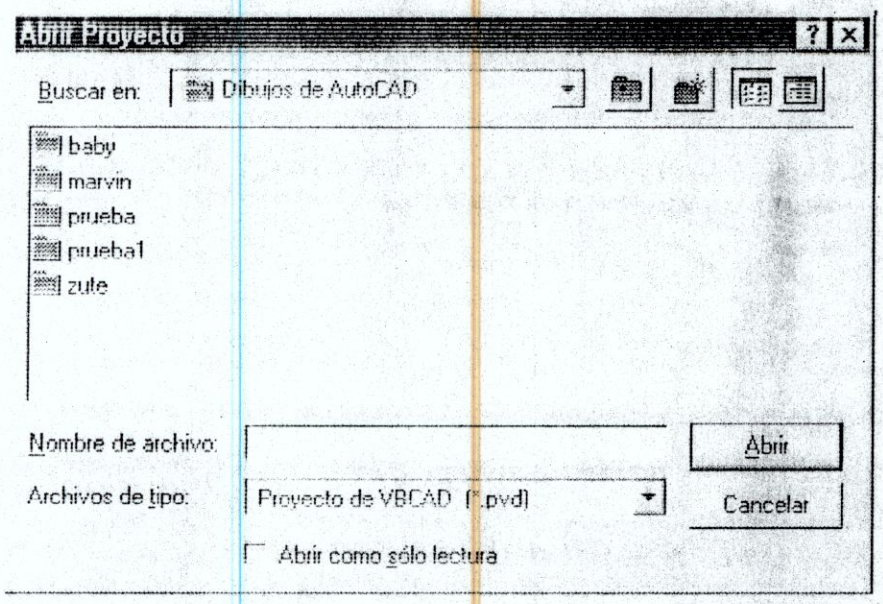


Figura_3.25. Cuadro de Dialogo II.

Una vez seleccionado el proyecto, se activara la Ventana Menú Principal con el proyecto previamente seleccionado.

3.4.1.3 Guardar Proyecto.

Al hacer clic en el comando con el nombre *"Guardar Proyecto"* aparecerán tres Cuadros de Dialogos, en donde el primero representado por la Figura_3.26, como su nombre lo indica es para crear una carpeta con un nombre designado por el usuario, el segundo representado por la Figura_3.27, utilizado para guardar el proyecto al cual se le asignara la extensión ".pvd" y por ultimo el tercer Cuadro de Dialogo representado por la Figura_3.28 , utilizado para guardar el sector al cual se le asigna por defecto t1 y se le da la extensión ".mzc" a medida que los

sectores se van incrementando, el valor del contador "t" se incrementara en la misma proporción.

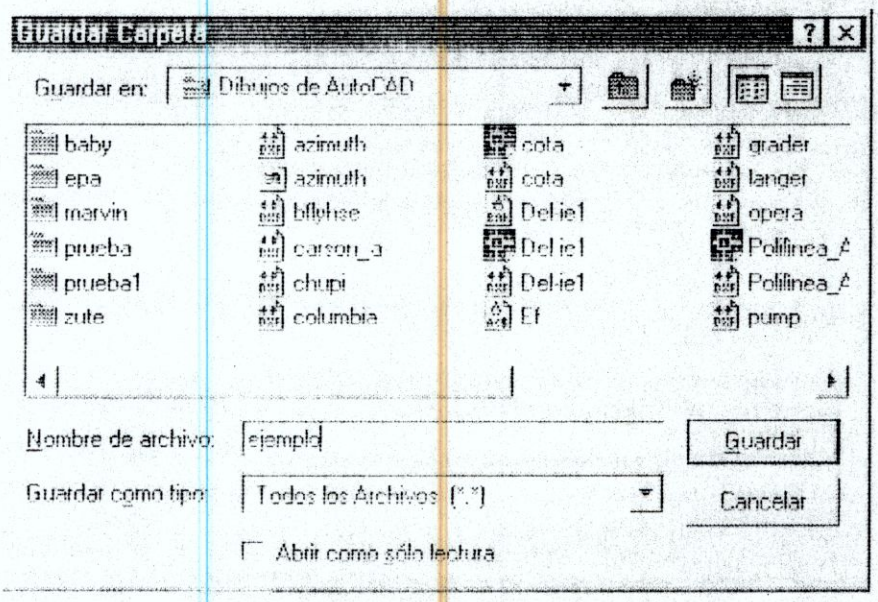


Figura_3.26. Cuadro de Dialogo III.

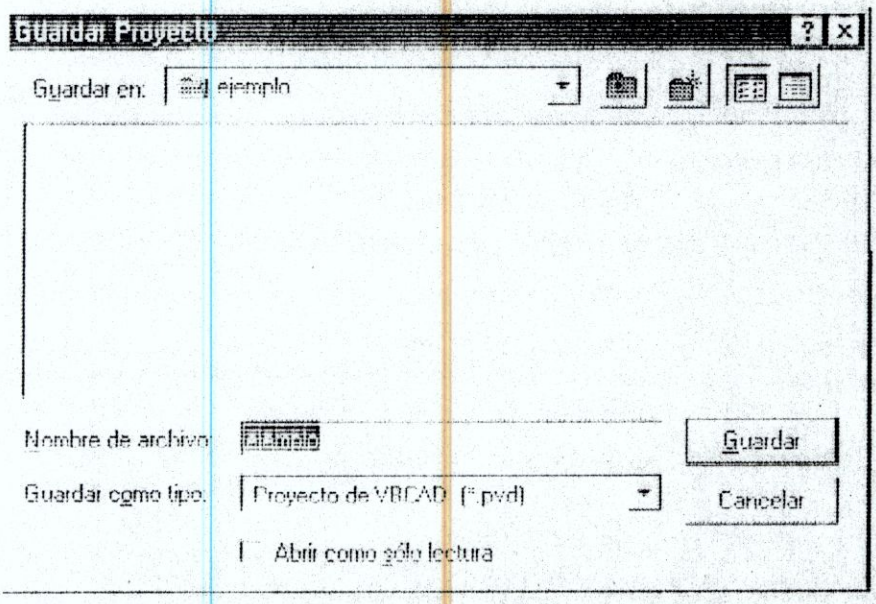


Figura_3.27. Cuadro de Dialogo IV.

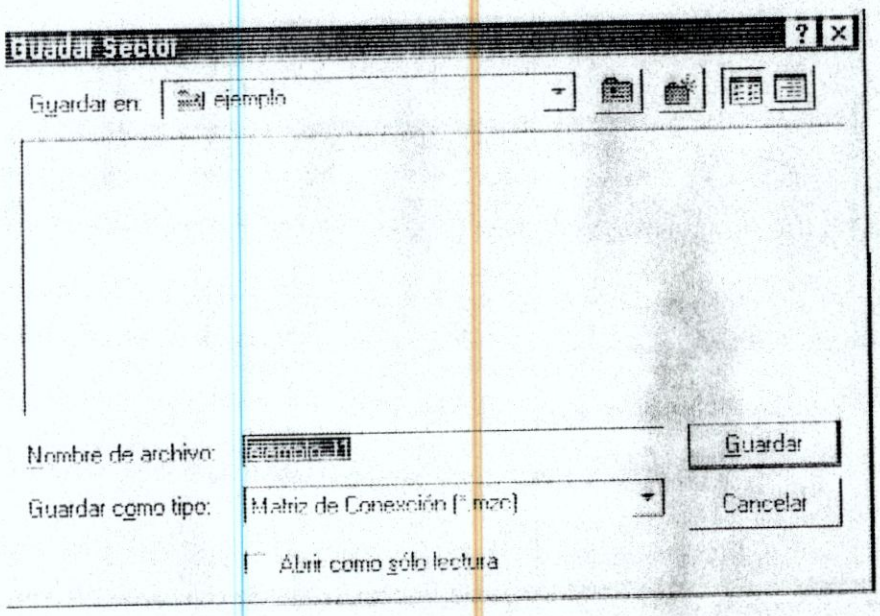


Figura_3.28. Cuadro de Dialogo V.

3.4.1.4 Eliminar Sector del Proyecto.

Al hacer clic en el comando con el nombre *"Eliminar Sector del Proyecto"* aparecerá el siguiente Cuadro de Dialogo VI (Figura_3.29) con la extensión (*.mzc) asignada anteriormente, posteriormente aparecerá un Cuadro de Mensaje IX representado por la Figura_3.30 ofreciéndole una alternativa al usuario de revocar la decisión tomada con anterioridad.

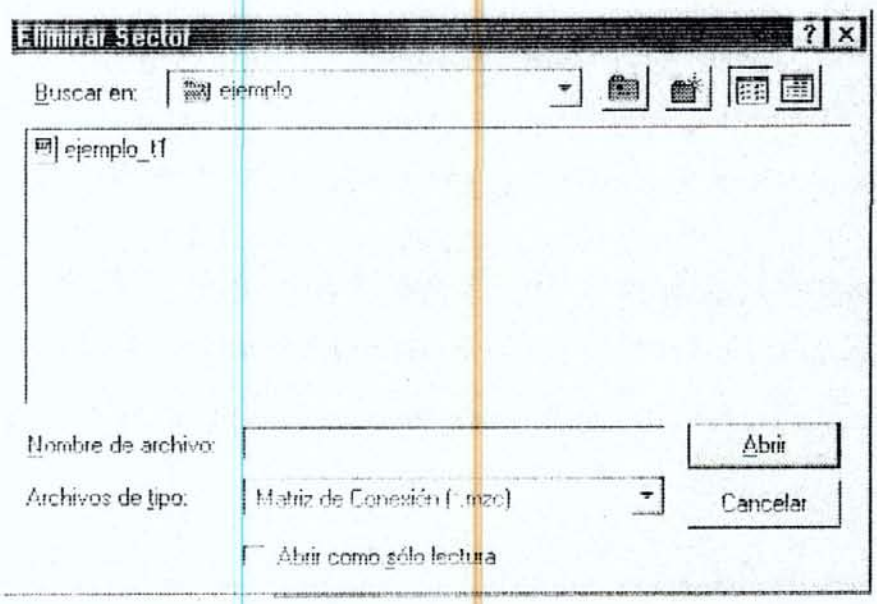


Figura_3.29. Cuadro de Dialogo VI.

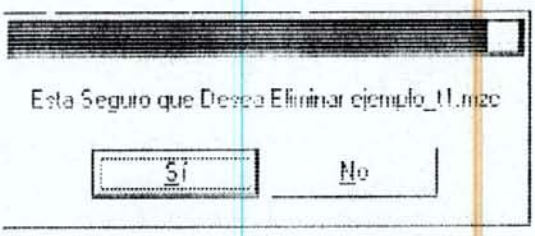


Figura_3.30. Cuadro de Mensaje IX.

3.4.1.5 Abrir Sector.

Al hacer clic en el comando con el nombre *"Abrir Sector"* aparecerá el siguiente Cuadro de Dialogo VII (Figura 3.31) con la extensión (*.mzc) asignada anteriormente, una vez seleccionado el sector, el mismo aparecerá en la ventana Menú Principal (Figura 3.2).

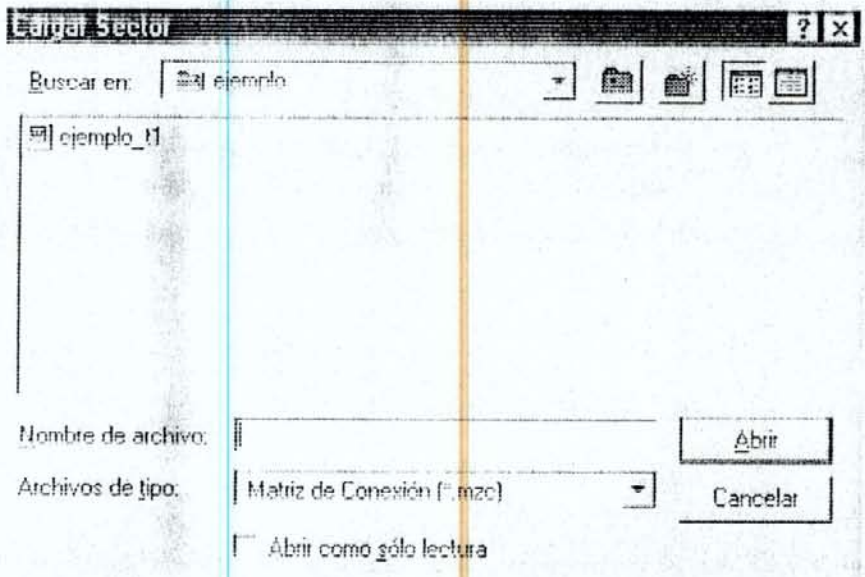


Figura_3.31. Cuadro de Dialogo VII.

3.4.1.6 Diseñar Nuevo Sector.

Al hacer clic en el comando con el nombre *"Diseñar Nuevo Sector"* aparecerá el proyecto cargado en la Ventana Diseño (Figura_3.8).

3.4.1.7 Imprimir.

Al hacer clic en el comando con el nombre *"Imprimir"* aparecerá la Ventana Imprimir I, representado por la Figura_3.32, en el cual se provee alternativas para imprimir informes completos es decir datos, resultados y grafico, de los sectores elegidos con anterioridad y cuya selección aparece por defecto, además si se desea imprimir solamente los datos o solamente los resultados o inclusive las dos opciones se debe hacer clic encima de los cuadritos seleccionados por el programa. También existe un botón denominado "Control de Impresora" presente en la Ventana Imprimir

I, el cual al activarlo carga la Ventana Imprimir II (Figura_3.33) en la cual se presentan caracteristicas de la impresión como son, calidad de impresión, tamaño de la hoja de impresion entre otras, cualquiera de estas alternativas son muy conocidas al usuario que haya trabajado con alguna de las herramientas de Windows.

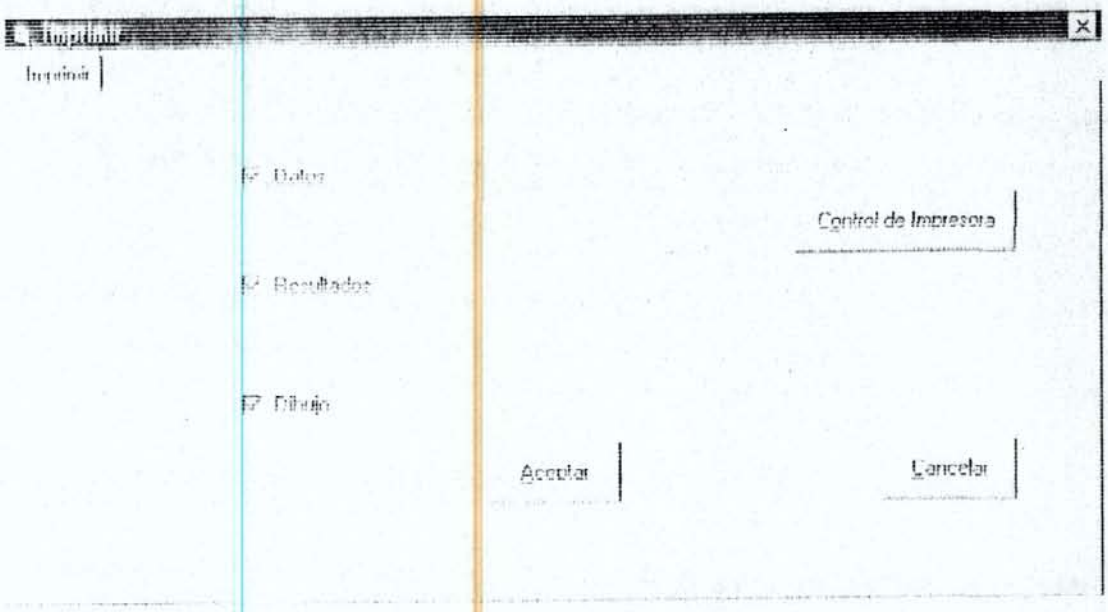


Figura_3.32. Ventana Imprimir I.

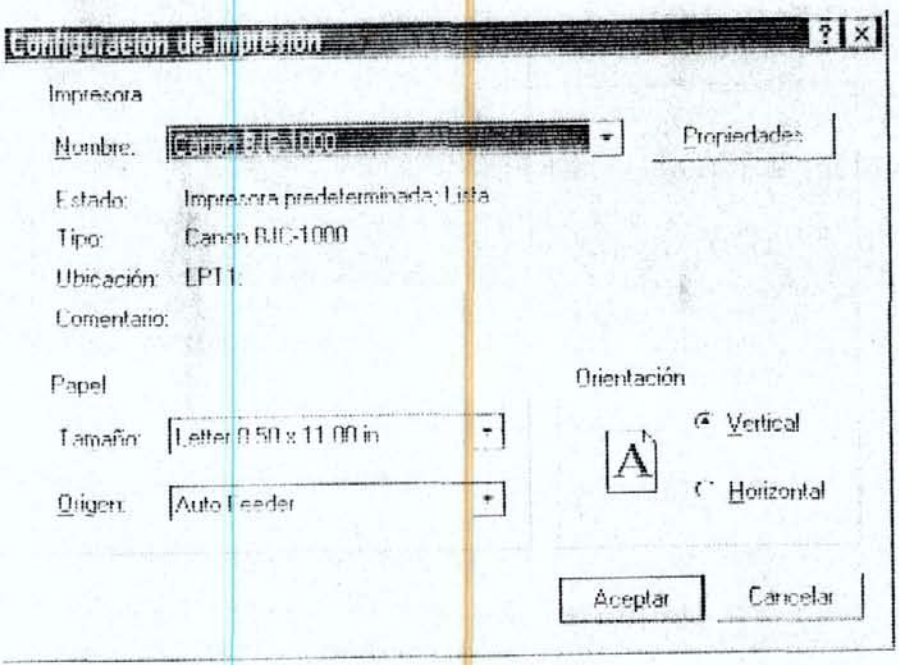


Figura_3.33. Ventana Imprimir II.

3.4.1.8 Salir.

Al hacer clic en el comando con el nombre *"Salir"* se saldrá del programa.

3.4.2 Opciones.

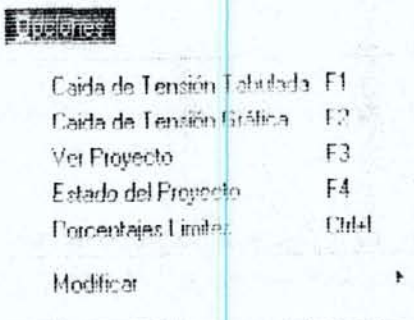


Figura_3.34. Menú Opciones.

Al hacer clic en el menú "Opciones" (Figura3.34) aparecerán los comandos presentes en éste menú

Caída de Tensión Tabulada o F1 ——► Representa los resultados de caída de tensión de todos los nodos de la red en estudio expresados en una tabla tabulada.

Caída de Tensión Gráfica o F2 ——► Representa los resultados de caída de tensión de todos los nodos de la red en estudio expresados mediante un código de colores.

Ver Proyecto o F3 ——► Representa la ubicación de todas las redes que se encuentran en el sector que correspondan.

Estado del Proyecto o F4 ——► Representa la ubicación y evaluación de todas las redes que se encuentran en el sector que correspondan.

Porcentajes límites o Ctrl+I ——► Representa la alternativa de cambiar el rango de valores estimados en la evaluación de la red.

Modificar ——► Representa alternativas de modificación para la mejora de la red en estudio.

"Se debe tener en cuenta que todos los comandos anteriormente nombrados trabajan bajo el ambiente de la ventana Menú Principal exceptuando el comando "*Modificar*" que ejecuta acciones en el ámbito de otras ventanas anteriormente conocidas, en cuanto al código de colores se utiliza como medio de evaluación de la red y consta de tres colores los cuales son verde, amarillo y rojo y que aparecerán en los tramos de la red, dependiendo de los valores de caída de tensión en los nodos de la red, además el

comando *"Porcentajes límites"* representa una alternativa en el caso de que el usuario desee cambiar el rango de evaluación de la red es decir cambiar los valores que estiman una red buena, regular o mala. Por ultimo se debe mencionar que en el comando *"Estado del Proyecto"* se utiliza este código de colores."

3.4.2.1 Porcentajes límites.

Al hacer clic en el comando *Porcentajes límites* se activa el Cuadro de Mensaje X (Figura 3.35) en donde el usuario escoge el valor tope que cataloga una red como buena, luego se posiciona el enfoque en el comando *aceptar* ubicado en el Cuadro de Mensaje X, el cual al seleccionarlo activa el Cuadro de Mensaje XI (Figura 3.36) en donde se escoge el valor frontera que cataloga una red como regular y mala es decir:

(0 – 3)% ⟶ red buena.

(3 – 5)% ⟶ red regular

(> 5)% ⟶ red mala.

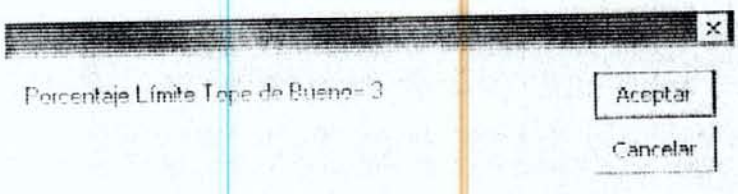


Figura 3.35. Cuadro de Mensaje X.

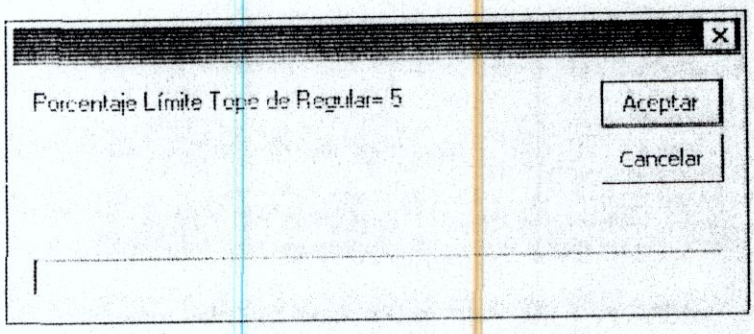


Figura_3.36. Cuadro de Mensaje XI.

3.4.2.2 Modificar.

Al hacer clic en el comando *Modificar* Figura_3.37 aparecerán tres ambientes de trabajo es decir alternativas de modificaciones en el diseño o características de la red.

Al seleccionar el comando *Topología de la Red* aparecerá la red en la Ventana Diseño (Figura_3.8) en donde se podrán realizar cambios en el ámbito de la topología de la red como pueden ser adicionar o borrar tramos, formar anillos etc.

Al seleccionar el comando *Conductor* aparecerá la ventana Conductor (Figura_3.10) en donde se podrá escoger cualquier otro conductor de la base de datos y visualizar los cambios que ocurren en la red.

Al seleccionar el comando *Cargas* aparecerá la ventana Cargas (Figura_3.16) en donde se podrán hacer cambios en las magnitudes de las cargas de los nodos que conforma la red y visualizar los cambios que ocurren en la misma.

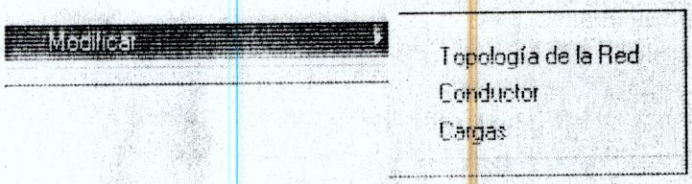


Figura 3.37. Modificar.

RECOMENDACIONES

Ya que el algoritmo utilizado para el cálculo de la caída de tensión se hace con números complejos este se puede utilizar, con una base de datos adecuada de los conductores para que calcule redes de distribución de baja tensión subterráneas.

Con modificaciones relativamente pequeñas el código fuente de VBCAD se puede cambiar para crear un programa que calcule redes de distribución de media tensión.

El software VBCAD se puede complementar con un programa que permita la coordinación de protecciones en redes de distribución.

Sintetizando las recomendaciones anteriores las cuales se pueden resumir en la creación de un conjunto de programas de distribución para mejorar significativamente la manera como se diseña en esta área.

CONCLUSIONES.

Para Discernir si el diseño de una red es bueno o no primero hay que saber para que compañía de distribución se esté trabajando, ya que cada una de estas acepta como límite de bueno un porcentaje de caída de tensión que no necesariamente es igual al de las otras.

Hoy en día el diseño asistido por computadora es común en todas las áreas de la ingeniería. La informática no está solamente en el futuro sino que ya forma parte del presente por lo tanto es necesario empaparse de este conocimiento en el mayor grado posible y el desarrollo de programas es una buena forma de lograrlo.

La utilización del plano urbanístico en el diseño de la red combinado con la exactitud de resultados y la facilidad de manejo convierten a VBCAD en una excelente herramienta de trabajo para estudiantes y profesionales, quienes muchas veces no tienen acceso a este tipo de programas debido a lo elevado que resulta la adquisición de los mismos.

En el trabajo de grado y en el CD de instalación se incluyó el código fuente de VBCAD para aquellas personas que deseen utilizarlo.

APÉNDICE

"A"

APENDICE A

TABLAS DE CONDUCTORES

A.1 Tabla para conductores de Aleación de Aluminio 6201 (ARVIDAL).

Calibre del Conductor	R (Ω / Km)	X (Ω / Km)
2	0.8709	0.3517
1/ 0	0.5477	0.3238
2/ 0	0.4343	0.3119
4/ 0	0.2731	0.2945

Nota: "(R)" Resistencia de Corriente Continua calculada a 20°C.

A.2 Tabla de conductores de Cobre Desnudo:

Calibre del Conductor	R (Ω / Km)	X (Ω / Km)
8	1.3436	0.3886
6	1.0661	0.3798
4	0.8451	0.3712
2	0.5313	0.3517
1/ 0	0.3341	0.3238
2/ 0	0.2649	0.3119
4/ 0	0.1666	0.2945

Nota: "(R)" Resistencia de Corriente Continua calculada a 20°C.

A.3 Cálculo del diámetro medio geométrico (ds):

ds: 0.363 * dm Para el #2 y el 1/ 0.

ds: 0.378 * dm Para el 2/ 0 al 4/ 0.

En donde:

dm: Es el diámetro del conductor.

A.4 Cálculo de la Reactancia (X):

$$X = 0,0754 * Ln\left(\frac{DMG}{ds}\right)$$ a 60Hz.

Donde :

DMG: es la distancia media geométrica.

ds: radio medio geométrico.

$$DMG = \sqrt[3]{D_1 * D_2 * D_3}$$

APÉNDICE

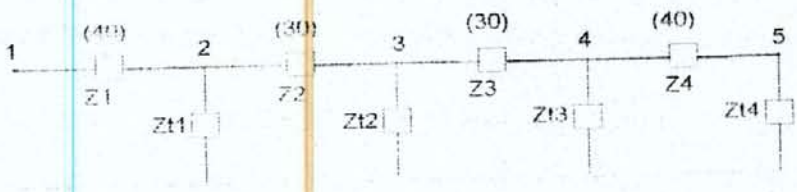
"B"

APÉNDICE B

ESTUDIO DE CARGAS

El programa VBCAD no toma en cuenta el valor de la impedancia de carga presente en cada nodo de la red a estudiar, la razón es debido a que la impedancia de carga es mucho mayor en magnitud que la impedancia de línea, y como ambas se encuentran en paralelo, la impedancia equivalente como es de esperarse es menor al valor de cualquiera de ellas pero muy parecido al valor de impedancia de línea, razón por la que VBCAD omite el valor de impedancia de carga.

Todo lo expresado anteriormente podrá observarse claramente en el ejemplo explicado a continuación, el cual se resolverá luego con el programa VBCAD sin incluir la impedancia de carga presente en cada uno de los nodos y se podrá observar comparando ambos resultados que no existe ninguna diferencia significativa en los valores de tensión V1, V2, V3, V4.



Figura_B1.Estudio de Cargas.

Conductor:

Cu – 4 ; R=0,8451 Ω / Km ; X=0,3693 Ω / Km.

Tensión en los Nodos:

V1 = (120 + J 0) volts (tensión de referencia monofásica).

V2 = ? ; V3 = ? ; V4 = ? ; V5 = ?.

Impedancias de los Tramos:

Z1 = 0,040 * (0,8451 + J 0,3693) Ω / Km = (0,0338 + J 0,0148) Ω.

Z2 = 0,030 * (0,8451 + J 0,3693) Ω / Km = (0,0253 + J 0,0111) Ω.

Z3 = 0,030 * (0,8451 + J 0,3693) Ω / Km = (0,0253 + J 0,0111) Ω.

Z4 = 0,040 * (0,8451 + J 0,3693) Ω / Km = (0,0338 + J 0,0148) Ω.

Impedancias de las Cargas:

Suponiendo una capacidad de 5 KVA trifásicos igual para todas las cargas y un factor de potencia de 0,90 atrasado.

Conociendo: $V = Z * I$; $S = V * I$; $S_{3\Phi} = 3 * S_{1\Phi}$

$$Z1 = \frac{120}{I}; \quad \text{en donde: } I = \frac{(\frac{5}{3})}{(\sqrt{3} * 0{,}208)}$$

Por lo que: Z1 = (7,776 + J 3,766) Ω.

Con la salvedad de que: Z1 = Z2 = Z3 = Z4 y además V = V1.

Impedancia Equivalente (Zth):

Buscando un thevenin desde la fuente se tiene:

$Za = Z4 + Zt4 = (7,8098 + J\,3,7808)\,\Omega.$

$Zb = Zt3 \,//\, Za = (3,8964 + J\,1,8867)\,\Omega$

$Zc = Z3 + Zb = (3,9217 + J\,1,8978)\,\Omega$

$Zd = Zt2 \,//\, Zc = (2,6069 + J\,1,2618)\,\Omega.$

$Ze = Z2 + Zd = (2,6322 + J\,1,2729)\,\Omega.$

$Zf = Zt1 \,//\, Ze = (1,9665 + J\,0,9513)\,\Omega.$

$Zth = Z1 + Zf = (2,0 + J\,0,9661)\,\Omega$

Cálculos:

Con el valor de Zth se procede a calcular la corriente en el primer tramo (I1).

$V2 = Zth * I1$, de donde: $I1 = \dfrac{(120 + J0)}{(2,0 + J0,9661)} = (48,6484 + J\,23,4996)\,A.$

$V2 = \sqrt{3} * (V1 - I1 * Z1) = \sqrt{3} * (118,00 + J0,0742)$ volts. $\longrightarrow$ **%V2 = 1,74.**

$V3 = Ze * I2$, de donde: $I2 = \dfrac{(118,00 - J0,0742)}{(2,6322 + J1,2729)} = (36,3437 - J\,17,5472)\,A.$

$V3 = \sqrt{3} * (V2 - I2 * Z2) = \sqrt{3} * (116,8858 + J\,0,1147)$ volts. $\longrightarrow$ **%V3 = 2,67.**

De la misma forma se calcula I3, I4 y por tanto V4 y V5 dando como resultado.

$V4 = \sqrt{3} * (116,1451 - J\,0,1451)$ volts. $\longrightarrow$ **%V4 = 3,28.**

$V5 = \sqrt{3} * (115,6516 - J\,0,1257)$ volts. $\longrightarrow$ **%V5 = 3,69.**

CORRIDA DEL EJEMPLO EN EL PROGRAMA

Redes del Proyecto

Red	Nodo de Referencia	Conductor	Factor de Potencia
1	1	CU-4	0,95

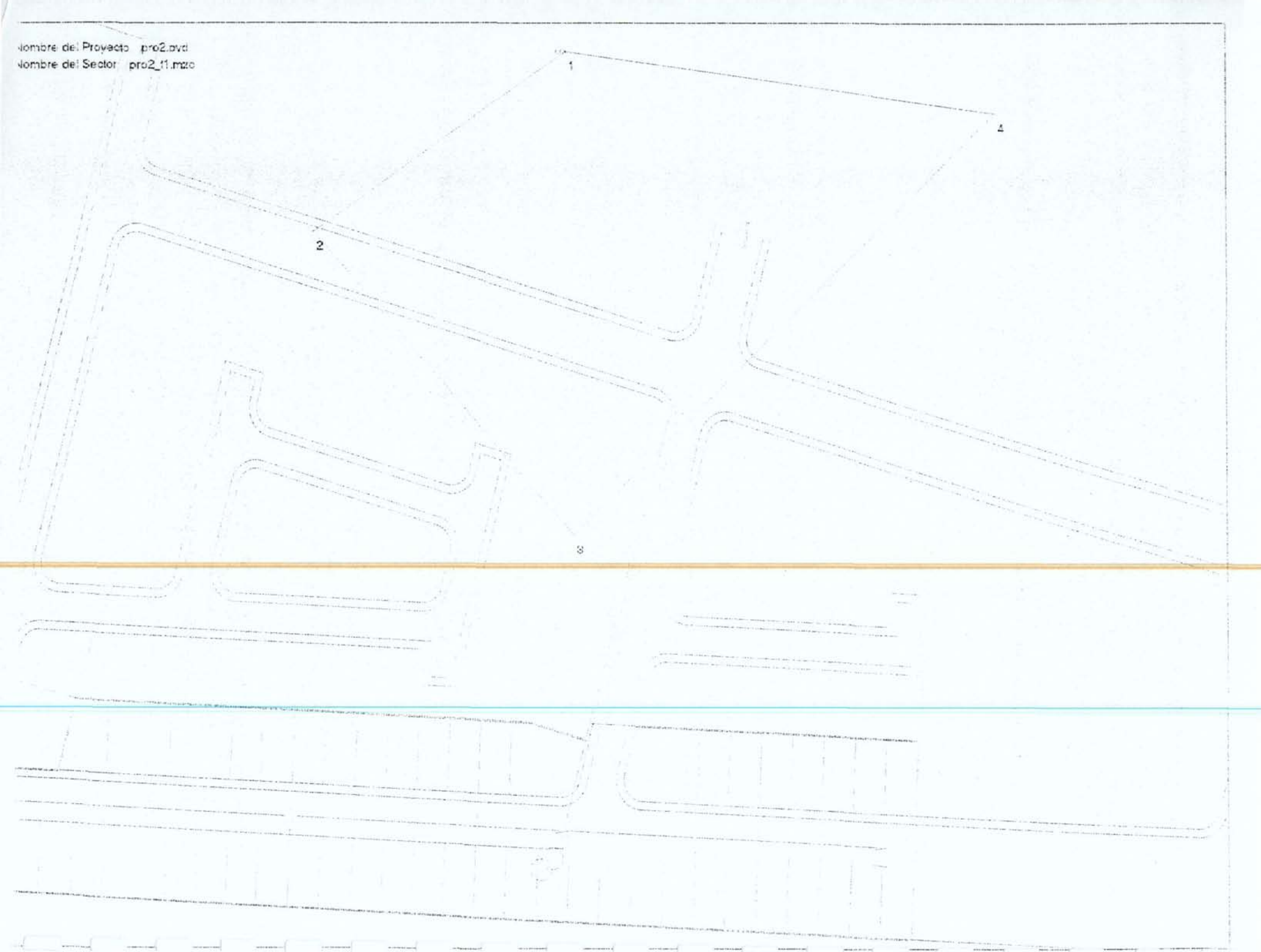
Caidas de Voltaje

Nodo	Caida de Voltaje(%)
1	0
2	1.4214
3	2.1376
4	1.6204

Estudio de la red

Red	Perdidas(w)	Máxima Caida de Voltaje(%)
1	238.98	2.14

Nombre del Proyecto pro2.pvd
Nombre del Sector pro2_f1.mzo



BIBLIOGRAFIA

- Cornel Gary : " **VISUAL BASIC 6.0** ", Mc. Graw Hill, 1999.

- Brown Steve : " **VISUAL BASIC 5.0** ", Anaya Multimedia, 1998.

- Halvorsen Michael: " **APRENDA VISUAL BASIC YA 5.0** ", Mc. Graw Hill, 1998.

- Grainger John y Stevenson William : **"ANALISIS DE SISTEMAS DE POTENCIA"**, Mc. Graw Hill, 1996.

- Canabal Carlos : **"PLANIFICACION DE DISTRIBUCION DE ENERGIA ELECTRICA"**

- Kolman Bernard: " **ALGEBRA LINEAL** ", Fondo Educativo Interamericano, 1981.

- Tajadura J.A. -Manso B. : " **PROGRAMACIÓN CON AUTOCAD** ", Mc Graw Hill, 1999.

- Domínguez José : " **AUTOCAD 14** ", Mc Graw Hill, 1998.

- Mc Kinney Bruce: " **PROGRAMACIÓN AVANZADA CON VISUAL BASIC** ", Mc Graw Hill, 1998.