Capítulo

3

PROGRAMACIÓN C++ EN AMBIENTE WINDOWS

Uno de los ambientes computacionales más usado actualmente es Microsoft Windows. Hasta hace poco tiempo para desarrollar una aplicación para Windows se debía aprender el "Microsoft Windows Software Development Kit" (SDK) y cientos de funciones y mensajes. Por esta razón Borland gracias a las ventajas que ofrece la Orientación a Objetos ofrece ahora la biblioteca o jerarquía de objetos "*ObjectWindows Library*®" (OWL), incorporada con C++.

ObjectWindows es un conjunto de clases que contienen las funciones y mensajes del "Microsoft Windows Software Development Kit", además tiene características importantes como las siguientes:

- Una curva fácil de aprendizaje.
- Una plataforma simple para el desarrollo de aplicaciones.
- Código reutilizable.
- Facilidades para la programación de ventanas, barras de íconos (button bars), línea de estado (status line), cajas de diálogo, controles, entre otros.

Por lo anterior este capítulo está orientado a presentar los elementos básicos de programación C++ en ambiente Windows mediante la reutilización de código con la jerarquía de objetos *Object-Windows*® 2.0 de Borland. Se incluyen temas como: el uso de menús, cajas de diálogo, aceleradores, íconos, mapas de bits (bitmaps), entre otros.

3.1 El programa mínimo

En esta sección presentamos el código mínimo que debe tener una aplicación Windows implementada con OWL, además se explica cómo se deben compilar y ejecutar tales aplicaciones.

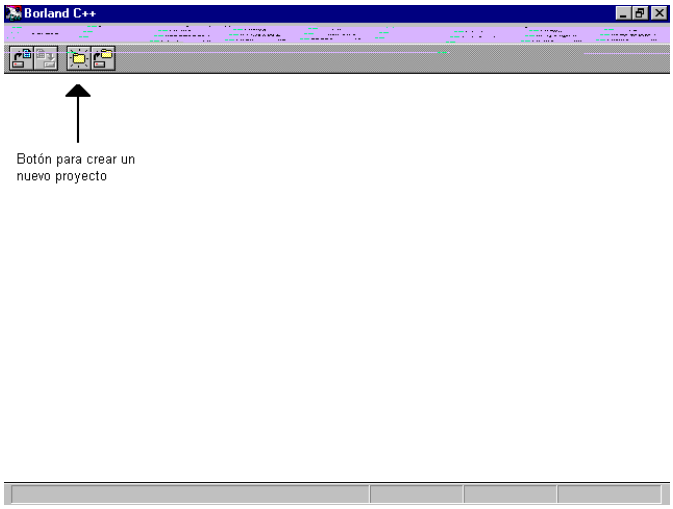


FIGURA 3.1. Icono para crear un nuevo proyecto en IDE.

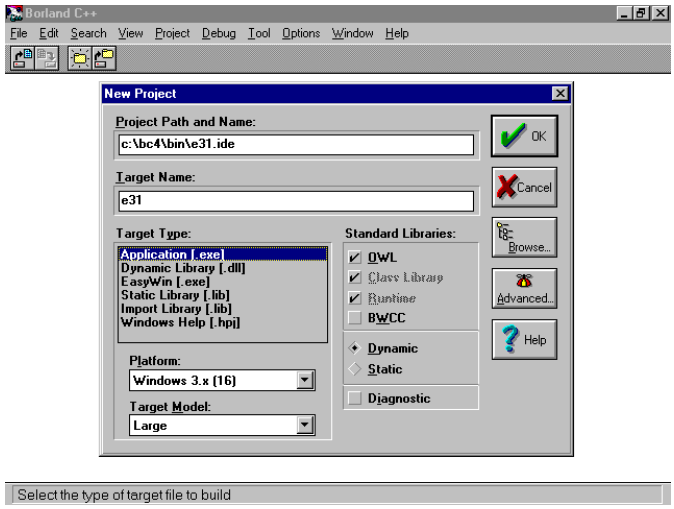


FIGURA 3.2. Caja de diálogo para crear un nuevo proyecto.

Para crear una aplicación con OWL siga los siguientes pasos:

1. Cargue Borland C++.
2. Seleccione **P**royect | **N**ew Proyect, o bien presione el ícono para crear proyectos nuevos, tal como se presenta en la Figura 3.1.

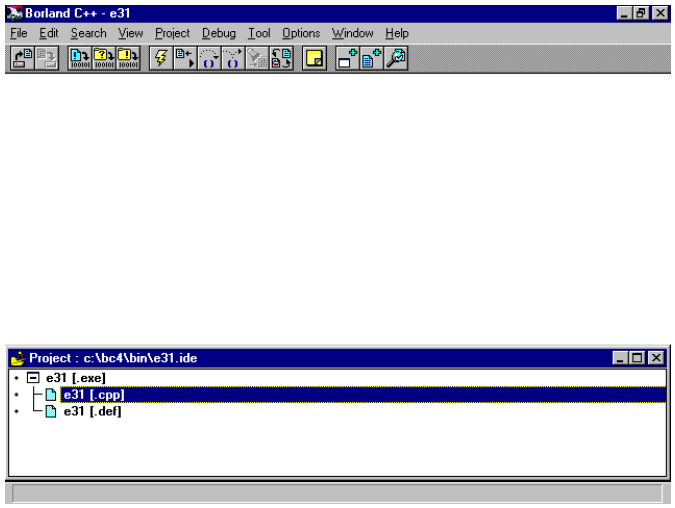


FIGURA 3.4. Ventana de proyectos en IDE.

1.

3. Luego Borland presenta la caja de diálogo de la Figura 3.2.

4. Mediante el botón **B**rouse... seleccione el directorio y el nombre adecuado para el proyecto, luego en el "Target T**y**pe" seleccione Application [exe] y en "Standard Libraries" seleccione los botones, tal como se muestra en la Figura 3.2.

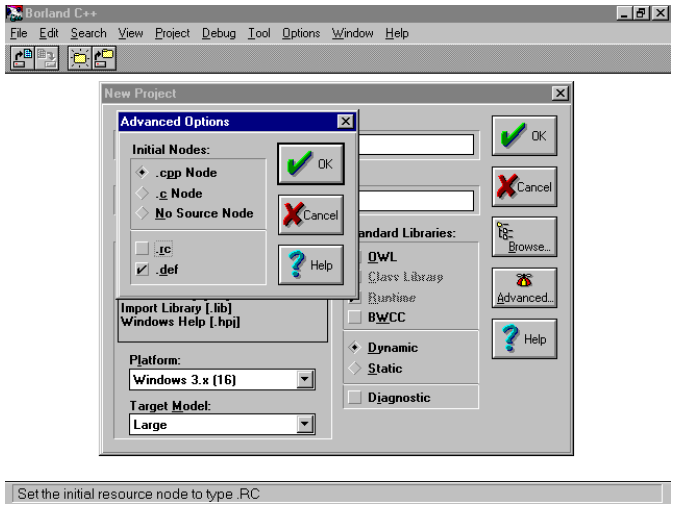


*FIGURA 3.3. Caja de diálogo de la opción **A**dvanced... en proyectos nuevos.*

1.

5. En esta misma caja de diálogo seleccione el botón **A**dvanced, le aparecerá la caja de diálogo de la Figura 3.3, seleccione .c**p**p Node y .**d**ef como se muestra en la Figura 3.3. Por tratarse de la aplicación mínima, este ejemplo no llevará archivo de recursos .rc, aunque prácticamente siempre deberá seleccionarse también.
6. Dé click en el botón OK, entonces aparecerá la ventana de la Figura 3.4. Luego dé doble click sobre el nodo e31.cpp como se muestra en la Figura 3.4 para que aparezca la ventana donde se editará el programa principal.
7. El programa se podrá compilar y ejecutar tal como se explicó en la sección 2.2.

El programa mínimo en OWL se presenta en el ejemplo 3.1. Este programa debe tener por lo menos dos partes: la clase ***TPrincipal*** derivada de TApplication y la función principal OwlMain que sustituye a la función main() de los programas tradicionales tipo C. La clase ***TPrincipal*** es la clase que nosotros hemos definido, por esto la destacamos en negrita e itálica; mientras que la clase TApplication forma parte de OWL.

🖫 Ejemplo 3.1. Programa mínimo en OWL:

```
// E31.CPP
#include <owl\owlpch.h>
#include <owl\applicat.h>
#include <owl\framewin.h>

class TPrincipal : public TApplication {
  public:
          TPrincipal() : TApplication()  {}  // Constructor
          void InitMainWindow();
};
// Implementación del método InitMainWindow
void TPrincipal::InitMainWindow() {
         SetMainWindow(new TFrameWindow(0, "Programa mínimo en
                                                Object-Windows"));
}

// Programa Principal en Object-Windows
```

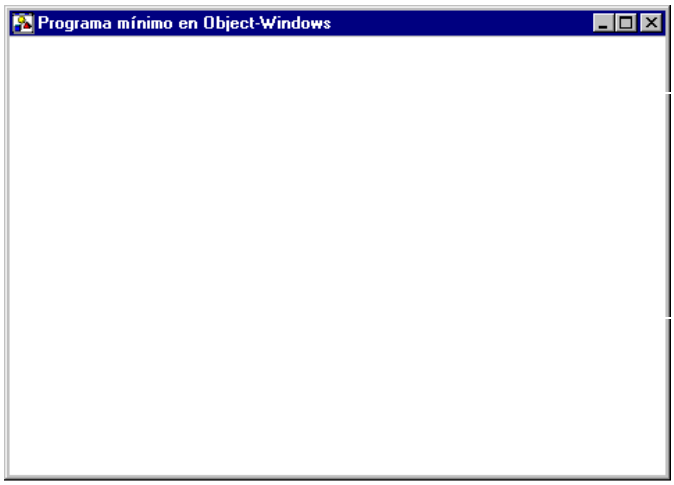


FIGURA 3.5. Salida del programa mínimo en OWL.

```
int OwlMain(int  argc, char* argv[] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

```
// E31.DEF  <------ No digite esta línea en el archivo pues produce error
NAME           E31
DESCRIPTION 'Aplicación mínima en OWL'
EXETYPE        WINDOWS
CODE           PRELOAD MOVEABLE DISCARDABLE
DATA           PRELOAD MOVEABLE MULTIPLE
HEAPSIZE       4096
STACKSIZE      8192
```

La salida que produce este programa es únicamente una instancia de TFrameWindow creada por el método InitMainWindow, es decir la ventana principal del programa. Esta ventana podrá ser minimizada, maximizada, movida y podrá tener controles, tales como el menú. La clase TFrameWindow forma parte OWL y se explica con detalle más adelante. Esta salida se presenta en la Figura 3.5.

La primera parte del programa mínimo es la clase que administra la aplicación, esta clase debe ser derivada de la clase TApplication de OWL, la cual provee las funcionalidades necesarias para inicializar la aplicación, crear la ventana principal, ejecutar el ciclo o paso de mensajes ("message loop" de Windows) para ejecutar la aplicación. La definición de esta clase se muestra en el siguiente fragmento de código:

```
class TPrincipal : public TApplication {
  public:
        TPrincipal() : TApplication()  {}  // Constructor
        void InitMainWindow();
};
```

Nótese que esta clase tiene dos métodos: el constructor que tiene código "inline" vacío y el método InitMainWindow() que redefine al método InitMainWindow() de la clase TApplication. Este método se encarga de crear la ventana principal y de ponerle un título. El código de este método es el siguiente:

```
void TPrincipal::InitMainWindow() {
        SetMainWindow(new TFrameWindow(0, "Programa mínimo en
```

```
                                        Object-Windows"));
}
```

El fragmento de código **new TFrameWindow(0, "Programa mínimo en Object-Windows")** lo que hace es invocar al constructor de la clase TFrameWindow (que forma parte de OWL) para crear la ventana principal.

Las relaciones existentes hasta ahora se muestran en la Figura 3.6. Las clases que forman parte de OWL se presentan sombreadas. De este diagrama se puede notar que el programa mínimo de OWL es sumamente pequeño; pero existe gran reutilización del código. Para ver un

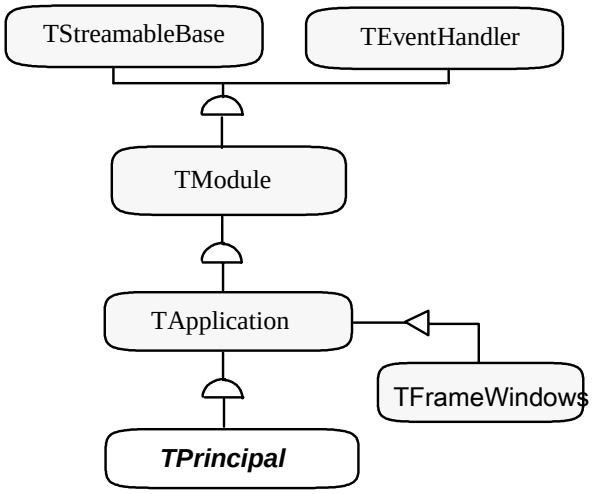


FIGURA 3.6. Relación de herencia en el programa mínimo de OWL.

diagrama con todo detalle, una vez que el proyecto esté cargado, seleccione la opción **V**iew | **C**lasses y Borland C++ desplegará un diagrama con las relaciones existentes entre las clases.

La segunda parte del programa mínimo en OWL es el programa principal OwlMain(. . .) el cual reemplaza la función main() de un programa usual tipo C o C++. El código de esta función es el siguiente:

```
int OwlMain(int  argc, char*  argv [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

Este programa principal consta de la declaración de una instancia o variable de tipo TPrincipal (clase que nosotros definimos) denominada en este caso Ap, esto mediante la instrucción:

```
TPrincipal Ap;
```

luego se ejecuta el método Run() de la clase TPrincipal, este método fue heredado de la clase TApplication, éste se encarga de correr la aplicación y de reconstruir la pantalla una vez que el programa concluya su ejecución, lo anterior se logra con la instrucción:

```
return Ap.Run();
```

El programa principal OwlMain(int argc, char* argv []) tiene como puede verse, dos argumentos: el primero de tipo entero argc y el segundo de tipo hilera argv[]. Los nombres de estos argumentos son muchas veces colocados como comentarios; pues generalmente no se utilizan, con lo cual se evitan dos mensajes de alerta (Warning). De modo que el código de la función OwlMain, usualmente es el siguiente:

```
// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

Para terminar esta sección diremos que todo programa Windows debe tener un archivo definición, cuya extensión es .DEF. En este archi-

vo se documenta el programa para futuras referencias y se controlan las características del mismo. Si el archivo .DEF no aparece en el proyecto Borland C++ tomará un archivo de definición estándar y enviará un mensaje de Warning. Para la aplicación mínima de OWL el archivo de definición es el siguiente:

```
NAME           E31
DESCRIPTION 'Aplicación mínima en OWL'
EXETYPE        WINDOWS
CODE           PRELOAD MOVEABLE DISCARDABLE
DATA           PRELOAD MOVEABLE MULTIPLE
HEAPSIZE       4096
STACKSIZE      8192
```

3.2 Ventanas bajo Object-Windows

Todas las aplicaciones que corren bajo Windows usan áreas rectangulares de la pantalla denominadas ventanas, con el fin de que la aplicación se comunique con el usuario. OWL ayuda a crear, desplegar, destruir ventanas, además ayuda a crear gráficos en las ventanas y sobre todo permite administrar toda la aplicación a través de ventanas.

En la sección anterior se mostró que la clase TApplication permite crear la ventana principal de la aplicación. En esta sección mostraremos cómo definir una nueva clase derivada de TFrameWindow para darle a esta ventana las características que deseamos para nuestra aplicación.

Toda aplicación OWL tiene una ventana principal. Las ventanas en OWL son todas derivadas de la clase TWindow. En el siguiente ejemplo se agrega al ejemplo 3.1 una nueva clase derivada de TFrameWindow la cual a su vez es derivada de TWindow, esto porque como se verá adelante TFrameWindow ofrece algunas ventajas adicionales.

🖫 Ejemplo 3.2. Ventana principal con OWL:

```
// E32.CPP
#include <owl\owlpch.h>
#include <owl\applicat.h>
#include <owl\framewin.h>
```

```
class TPrincipal : public TApplication {
  public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};

class TVentana : public TFrameWindow {
public:
          TVentana(TWindow *Padre, const char far *titulo) :
                                        TFrameWindow(Padre, titulo) { };
};
void TPrincipal::InitMainWindow() {
          SetMainWindow(new TVentana(0, "Programa mínimo en
                                                Object-Windows"));
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run()
;
}
```

```
// E32.DEF  <------ No digite esta línea en el archivo pues produce error
NAME            E32
DESCRIPTION 'Aplicación mínima en OWL'
EXETYPE         WINDOWS
CODE            PRELOAD MOVEABLE DISCARDABLE
DATA            PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

En este ejemplo se ha definido la clase TVentana derivada de la clase TFrameWindow, esto con el siguiente código:

```
class TVentana : public TFrameWindow {
public:
          TVentana(TWindow *Padre, const char far *titulo) :
                                        TFrameWindow(Padre, titulo) { };
```

```
};
```

Aunque en este ejemplo no se hace aún, la idea de definir la clase TVentana es agregar características adicionales a la ventana principal del programa, como veremos adelante. Por ahora esta clase tiene solamente el método constructor. Este método tiene código "inline" vacío y un llamado al constructor de la clase base.

Ahora bien la ventana principal es realmente creada cuando el método InitMainWindow() de TPrincipal es invocado, esta invocación se da en forma automática, veamos el código de este método:

```
void TPrincipal::InitMainWindow() {
        SetMainWindow(new TVentana(0, "Programa mínimo en
                                                Object-Windows"));
}
```

Nótese que en este método se hace un llamado al constructor de la clase TVentana que se encarga de crear y desplegar la ventana principal mediante el llamado al constructor de la clase TFrameWindow.

Cuando el constructor de TVentana es invocado, la ventana principal aparece casi en forma mágica. Qué sucedió realmente cuándo se declaró la ventana principal durante la ejecución de InitMainWindow()? Lo que sucedió es que el método InitMainWindow() llamó en dos métodos de la clase TFrameWindow, estos son Create y Show. El método Create crea e inicializa los elementos o variables de la interfaz y el método Show despliega la ventana en la pantalla.

Algunas otras funciones o métodos importantes de la clase TFrameWindow se presentan en la siguiente tabla:

Nombre de la Función	Descripción de la función
CanClose	Determina si una ventana existe, para luego cerrarla.

Nombre de la Función	Descripción de la función
Create	Crea los elementos de interfaz asociados con la ventana.
GetClientRect	Retorna el tamaño del área cliente de la ventana.
GetParent	Retorna un puntero a la ventana padre.
Invalidate	Invalida el área cliente para ser redibujada.
InvalidateRect	Invalida una porción del área cliente para ser redibujada.
Paint	Sobrecarga esta función para desplegar información en la pantalla.
SetCaption	Cambia el título de la ventana.
SetCursor	Cambia el tipo de cursor que está siendo desplegado.
Show	Despliega u oculta la ventana en la pantalla.
UpDateWindow	Fuerza el área cliente de la ventana a ser redibujada inmediatamente.

En la ayuda del Borland C++ puede obtenerse información bastante completa sobre estos métodos de la clase TFrameWindow. Más adelante trabajaremos con mayor detalle algunas de éstos.

Ventanas hijas (Child Windows)

Una ventana hija ("*Child Window*") pertenece a la ventana madre ("*Parent Window*"), ésta se puede crear inmediatamente después de la creación de la ventana madre o bien posteriormente. Una ventana hija es destruida automáticamente si la ventana madre es destruida. La relación madre-hija es en realidad una relación Componente-Compuesto, pues se debe tener un puntero hacia algún tipo de ventana dentro de la clase donde está definida la ventana madre, luego a través de este puntero se puede crear la ventana, enviar los mensajes a la ventana hija, o bien cerrar la ventana. En el ejemplo 3.3 se agrega al código del ejemplo 3.2 una ventana hija.

🖫 Ejemplo 3.3. Ventanas hijas con OWL:

```
// E33.CPP
#include <owl\owlpch.h>
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\edit.h>

class TPrincipal : public TApplication {
  public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};
class TVentana : public TFrameWindow {
   TEdit *VentanaHija;
public:
          TVentana(TWindow *Padre, const char far *titulo);
};

void TPrincipal::InitMainWindow() {
          SetMainWindow(new TVentana(0, "Ejemplo de ventana hija en
                                                          Object-Windows"));
}
TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                          : TFrameWindow(Padre, titulo) {
  VentanaHija = new TEdit(this, 101, "Ejemplo de ventana hija",
                                                  10,10,300,30,0,FALSE);
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E33.DEF <------ No digite esta línea en el archivo pues produce error

```
NAME            E33
DESCRIPTION 'Aplicación mínima en OWL'
EXETYPE         WINDOWS
CODE            PRELOAD MOVEABLE DISCARDABLE
DATA            PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

Como puede verse en la definición de la clase TVentana, y como ya se ha dicho, la relación madre-hija es una relación Componente- Compuesto, pues en esta clase se declaró una variable VentanaHija tipo puntero a TEdit, esta variable permitirá crear la ventana hija. La clase TEdit pertenece a Object-Window, de hecho la variable VentanaHija puede declararse como un puntero a cualquier clase derivada de TWindow o bien a la misma TWindow de Object-Windows, según las necesidades del programa, como se muestra en el siguiente fragmento de código.

```
class TVentana : public TFrameWindow {
  TEdit *VentanaHija;   // Puntero para la ventana hija
public:
        TVentana(TWindow *Padre, const char far *titulo);
};
```

En este ejemplo el constructor de la clase TVentana ya no tiene el código "inline" pues este es un poco más amplio; ya que se debe inicializar y crear la ventana hija, como se muestra en el siguiente fragmento de código.

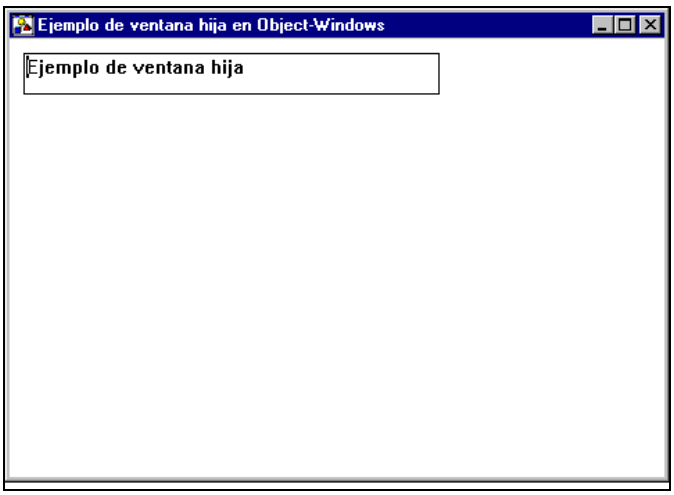
FIGURA 3.7. *Ejemplo de venta-*

```
TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                        : TFrameWindow(Padre, titulo) {
 VentanaHija = new TEdit(this, 101, "Ejemplo de ventana hija",
                                                  10,10,300,30,0,FALSE);
}
```

Como se desprende de este código la variable VentanaHija se inicializa invocando a constructor de TEdit con sus respectivos parámetros. El detalle sobre los constructores de cada clase de Object-Windows se puede consultar mediante la ayuda de Borland C++ *FIGURA* 3.7.

```
void TPrincipal::InitMainWindow() {
        SetMainWindow(new TVentana(0, "Ejemplo de ventana hija en
                                                        Object-Windows"));
}
```

La salida del programa del ejemplo 3.3 se muestra en la Figura 3.7.



3.3 La Tabla de Respuestas a Eventos

La "***Tabla de Respuestas a Eventos***" (Event Response Table) o simplemente la "*Tabla de Respuestas*" es el mecanismo central que provee OWL para la administración de eventos a través de menús, cajas de diálogo, barra de herramientas (tools bars), entre otros. en una aplicación Windows, de ahí su importancia. Dedicamos toda esta sección a explicar este mecanismo tan importante en OWL 2.0.

Justamente la Tabla de Respuestas aparece como una de las principales mejoras de OWL 2.0 respecto a OWL 1.0, ya que permite que el código OWL sea transportable a otros compiladores de C++, debido a que gracias a esta tabla se suprimen algunas extensiones al lenguaje C++ que aparecían en OWL 1.0. Es decir OWL 2.0 logra realmente hacer "compatibles" la Programación Orientada a Objetos con el uso de recursos de interfaz de Windows.

Todas las aplicaciones Windows están basadas en el manejo de eventos. Un evento es un mensaje que Windows envía a su aplicación para que algo importante ocurra. Por ejemplo, un mensaje podría indicar que el usuario seleccionó una opción del menú, presionó algún botón, minimizó la ventana, entre otros. Así Windows puede generar cientos de diferentes eventos. OWL provee un mecanismo llamado la *Tabla de Respuestas* para facilitarle al programador el manejo de los eventos que está interesado en procesar. El procesamiento de un evento se hace a través de un "método-evento" el cual es definido en alguna clase de su aplicación.

La tabla de respuestas es un mecanismo que brinda OWL de C++ para ligar las clases (propias de la Programación Orientada a Objetos) con los recursos de interfaz provistos por Windows, es decir, es una forma de ligar los métodos de las clases con los recursos de interfaz (menús y cajas de diálogo) que se constru

yen con cualquier generador de recursos de interfaz para Windows Resource WorkShop.

La *Tabla de Respuestas* es un miembro de la clase Window de OWL, por lo que la tabla de respuestas podrá ser usada en cualquier clase derivada de TWindow. Por ejemplo, podría ser usada en la clase TVentana que se presentó en el ejemplo 3.2; ya que TVentana es una clase deriv

ada de TFrameWindow, la cual a su vez deriva de la clase Window.

En el ejemplo 3.4 se presenta una Tabla de Respuestas asociada a la clase TVentana. Esta tabla de respuestas le permite al programa detectar si el usuario presionó el botón derecho del mouse o bien si presionó el botón izquierdo.

Ejemplo 3.4. Tabla de respuestas en OWL:

```
// E34.CPP
#include <owl\owlpch.h>
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\edit.h>
```

```
class TPrincipal : public TApplication {
  public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};

class TVentana : public TFrameWindow {
public:
          TVentana(TWindow *Padre, const char far *titulo);
          // Funciones que responden a la tabla de Respuestas
          void EvLButtonDown(UINT, TPoint&);
          void EvRButtonDown(UINT, TPoint&);
          DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
          SetMainWindow(new TVentana(0, "Ejemplo de manejo de eventos
                    Object-Windows"));
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                    : TFrameWindow(Padre, titulo) {
}

void TVentana::EvLButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón izquierdo del mouse", "Manejo de
          Eventos", MB_OK);
}

void TVentana::EvRButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón derecho del mouse", "Manejo de
          Eventos", MB_OK);
}

// Programa Principal en Object-Windows
```

```
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}

// E34.DEF  <------ No digite esta línea en el archivo pues produce error
NAME           E34
DESCRIPTION 'Manejo de eventos en OWL'
EXETYPE        WINDOWS
CODE           PRELOAD MOVEABLE DISCARDABLE
DATA           PRELOAD MOVEABLE MULTIPLE
HEAPSIZE       4096
STACKSIZE      8192
```

La tabla de respuestas es un miembro más de la clase TVentana, es decir, la tabla de respuestas debe ser *declarada* dentro de la definición de clase TVentana, tal como se muestra en el siguiente fragmento de código:

```
class TVentana : public TFrameWindow {
public:
        .....................
        DECLARE_RESPONSE_TABLE(TVentana);
};
```

OWL tiene una serie de macros que ayudan a declarar y definir la tabla de respuestas. Después de que la tabla de respuestas es declarada debe ser *definida*, esto se hace fuera de la definición de la clase. La definición de la tabla de respuestas para la clase TVentana se muestra en el siguiente fragmento de código.

```
DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
END_RESPONSE_TABLE;
```

La declaración y definición de la tabla de respuestas es muy simple y debe seguir las siguientes reglas:

1. La *declaración* de la tabla de respuestas debe estar dentro de la clase.
2. La primera línea de la *definición* de la tabla de respuestas debe ser siempre el macro DEFINE_RESPONSE_TABLEX. El valor de X depende de cuántas clases hereda la clase en donde se declaró la tabla de respuestas, es decir el número inmediato de clases base. En el ejemplo anterior la primera línea de la definición de la tabla de respuestas es DEFINE_RESPONSE_TABLE1(. . .), aquí X tomó el valor de 1; ya que la clase TVentana es derivada en forma inmediata únicamente de la clase TFrameWindow.
3. La última línea de la tabla de respuestas es siempre el macro END_RESPONSE_TABLE, el cual finaliza la definición de la tabla de respuestas.
4. Entre el macro DEFINE_RESPONSE_TABLEX y el macro END_RESPONSE_TABLE están otros macros asociados a eventos particulares para el manejo de las funciones propias del programador.
5. El macro recibe X+1 argumentos; el primero es el nombre de clase en donde está definida la tabla de respuestas, el resto es, los nombres de las clases respecto de las cuales se está recibiendo herencia directamente. En el ejemplo anterior la primera línea del macro es: DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow) de donde se observa que los parámetros son TVentana nombre de la clase a la que pertenece el macro y TFrameWindow nombre de la clase base de TVentana.

OWL define macros para todos los mensajes de Windows, simplemente agregue el macro apropiado a la tabla de respuestas si usted desea que su programa responda a algún mensaje en particular. Los macros de OWL para mensajes de Windows siempre comienzan con EV_ y concluyen con el nombre completo del mensaje en Windows. En la tabla de respuestas del ejemplo anterior se usó el macro EV_WM_LBUTTONDOWN de OWL, esto indica que se desea procesar el mensaje WM_LBUTTONDOWN de Windows. Como ya hemos mencionado Windows tiene cientos de mensajes similares al anterior; para conocerlos se debe consultar cualquier manual de *programación* en Win-

dows, es decir se debe conocer "*Windows Application Programmer Interface (API)*".

Además para responder a los mensajes de Windows, se deben crear las funciones de respuesta dentro de la clase C++, el nombre de una función que responde a un mensaje Windows debe comenzar con Ev seguido por el nombre del mensaje Windows, excepto la inicial WM_. Recuerde que el nombre debe comenzar con letra mayúscula y cada palabra completa debe comenzar con letra mayúscula. En el ejemplo anterior la función de la clase TVentana que responde al mensaje Windows WM_LBUTTONDOWN es la función EvLButtonDown. Estas funciones que responden a los mensajes de OWL se presentan en el manual de referencia "*OWL Reference Guide*" que viene con el paquete Borland C++.

En el ejemplo anterior usamos dos mensajes Windows WM_LBUTTONDOW y WM_RBUTTONDOW, que indican si el botón izquierdo o el botón derecho del mouse han sido presionados, respectivamente. Para estos dos mensajes de Windows, OWL provee dos macros EV_WM_LBUTTONDOWN y EV_WM_RBUTTONDOWN que *FIGURA* 3.8.

FIGURA 3.8.
Uso de la función

```
void TVentana::EvLButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón izquierdo del mouse", "Manejo de
                Eventos", MB_OK);
}

void TVentana::EvRButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón derecho del mouse", "Manejo de
                Eventos", MB_OK);
}
```

Estas funciones despliegan por pantalla una caja de diálogo en la que se indica qué botón fue presionado, esto se muestra en la Figura 3.8.

La caja de diálogo fue desplegada mediante la utilización de la función de la MessageBox de la clase TWindow de OWL. Esta función recibe tres parámetros. El primero es el mensaje que despliega, el segundo es el título de la caja de diálogo y el tercero es el "tipo" de

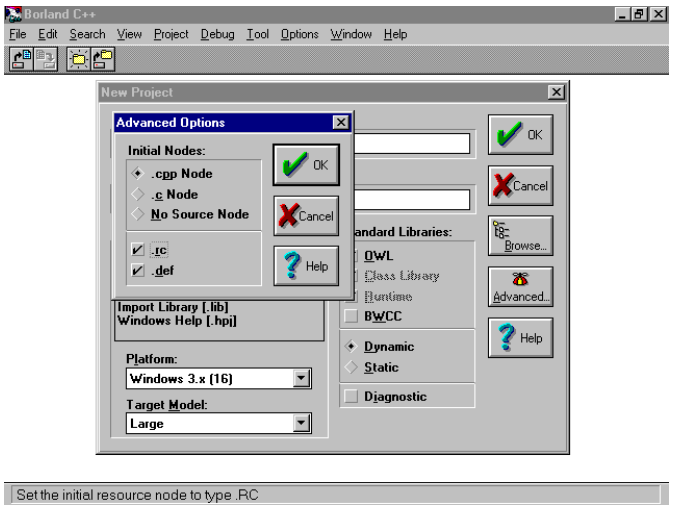


*FIGURA 3.9. Caja de diálogo de la opción **Advanced...** en proyectos nuevos.*

3.4 Menús

Casi todas las aplicaciones de Windows tienen un menú del cual se pueden escoger "comandos" para ser ejecutados. OWL provee mecanismos bastantes simples para agregar un menú a una ventana, esto mediante la clase TMenu. Esta clase también tiene métodos que permiten modificar los menús en tiempo de ejecución, como veremos en la sección 3.5.

Para agregar un menú a una aplicación se deben seguir los siguientes nueve pasos:

1. Cree un proyecto nuevo, tal como se explica en los pasos 1,2 y 3 de la sección 3.1.

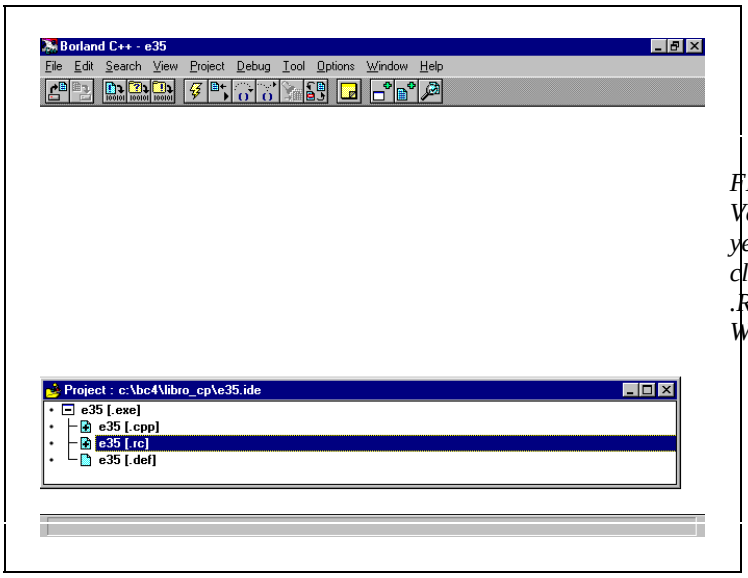


FIGURA 3.10. Ventana de proyectos, dé doble click en el nodo .RC para cargar Workshop.

2. En la caja de diálogo que se muestra en la Figura 3.2 (sección 3.1) seleccione el botón **A**dvanced, le aparecerá la caja de diálogo de la

Figura 3.9, seleccione .CPP Node, .DEF y .RC tal como se muestra en esta Figura.

3. Dé doble click en el nodo .RC de la ventana de proyectos, como se muestra en la Figura 3.10. En este archivo se deben almacenar todos los recursos de interfaz tales como: menús, cajas de diálo-

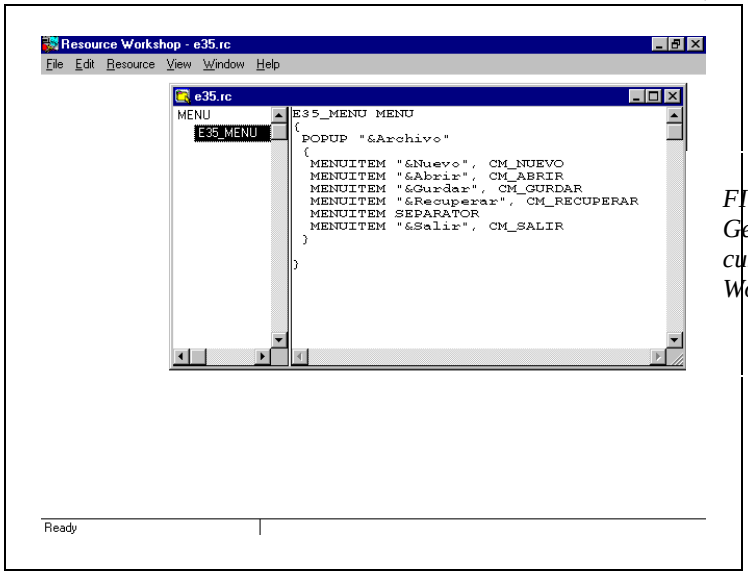


FIGURA 3.11. Generador de recursos de interfaz Workshop.

go, tablas de hileras (StringTables),entre otras. Una vez que se da doble click en esta opción, Borland C++ carga automáticamente el programa ***Resource Workshop,*** el WorkShop se muestra en la Figura 3.11. W

orkshop permite crear en forma gráfica los recursos de interfaz de una aplicación, generando en forma simultánea el código Windows de estos. El uso de Workshop es bastante simple y su forma de utilización escapa a los objetivos de este libro; para una

4. Una vez cargado el Workshop genere el código para el menú. (este código puede generarse en forma de texto en el editor de C++ si se conoce bien el código Windows para menús). Por ejemplo, supóngase que desea agregar el menú que se muestra en la Figura 3.12 al ejemplo 3.4 de la sección anterior, entonces el código Windows para este menú generado por Workshop es el siguiente: (E35_MENU es el nombre que se le dío al menú y con este nombre será referenciado posteriormente)

```
E35_MENU MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Gurdar", CM_GURDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
}
```

FIGURA 3.12.
Ejemplo de un menú.

5. Como puede verse en este código, cada opción del menú tiene asociada una constante entera, por ejemplo CM_NUEVO, a la cual en Workshop le asignó el número entero 101. Por esta razón en este paso se debe crear un archivo en donde se definan estas constantes, usualmente este archivo es extensión .H. Por ejemplo el archivo de constantes del menú de la Figura 3.12 se llama E35.H y su código es el siguiente: (se debe tener cuidado de que los números no sean repetidos y que sean los mismos que asignó Workshop)

```
// E35.H
#define CM_NUEVO        101
#define CM_ABRIR        102
#define CM_GUARDAR      103
#define CM_RECUPERAR    104
#define CM_SALIR        1
```

6. Este archivo debe ser conocido tanto en el programa C++ como en el archivo de recursos, esto se logra mediante un #include en el programa principal .CPP y otro en el archivo de recursos .RC, por ejemplo el archivo E35.RC queda como sigue:

```
#include "e35.h"
E35_MENU MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Gurdar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
}
```

7. Se debe asociar el menú a la ventana principal del programa C++, esto se logra modificando el constructor de la clase TPrincipal co-

mo sigue: (nótese que E35_MENU fue el nombre que se le dio al menú en Workshop)

```
void TPrincipal::InitMainWindow() {
        SetMainWindow(new TVentana(0, "Ejemplo de menús en
                        Object-Windows"));
        GetMainWindow()->AssignMenu("E35_MENU");
}
```

8. Se debe agregar un método en la clase TVentana para cada una de las opciones del menú, como se muestra en el siguiente fragmento de código (en este caso lo único que hacen estos métodos es presentar un mensaje indicando que aún no han sido implementados):

```
class TVentana : public TFrameWindow {
public:
        TVentana(TWindow *Padre, const char far *titulo);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Nuevo();
        void Abrir();
        void Guardar();
        void Recuperar();
        DECLARE_RESPONSE_TABLE(TVentana);
};
. . . . .

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
```

```
void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
```

9. Finalmente se debe ligar el número que se asignó a cada opción del menú en el archivo .H y en el Workshop con el método correspondiente. Esto se hace a través de la *Tabla de Respuestas* mediante el macro EV_COMMAND(Constante,NombreMetodo), por ejemplo, el macro EV_COMMAND(CM_NUEVO,Nuevo) causa que el método Nuevo() sea invocado cuando el ítem del menú correspondiente a la constante CM_NUEVO es escogido por el usuario. De tal modo que la tabla de respuestas para el menú de la Figura 3.12 es el siguiente: (el método CmExit() es heredado de la clase TWindow de OWL)

```
DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
  EV_COMMAND(CM_SALIR,CmExit),
END_RESPONSE_TABLE;
```

Para una mejor comprensión a continuación presentamos de nuevo el código completo del ejemplo 3.4, pero se le ha agregado el menú de la Figura 3.12.

🖫 Ejemplo 3.5. Ejemplo de un menú con OWL:

```
// E35.CPP
#include <owl\owlpch.h>
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include "e35.h"

class TPrincipal : public TApplication {
```

```
 public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};

class TVentana : public TFrameWindow {
public:
          TVentana(TWindow *Padre, const char far *titulo);
          // Functiones que responden a la tabla de Respuestas
          void EvLButtonDown(UINT, TPoint&);
          void EvRButtonDown(UINT, TPoint&);
          // Métodos Asociados al Menú
          void Nuevo();
          void Abrir();
          void Guardar();
          void Recuperar();
          DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
 EV_WM_LBUTTONDOWN,
 EV_WM_RBUTTONDOWN,
 EV_COMMAND(CM_NUEVO,Nuevo),
 EV_COMMAND(CM_ABRIR,Abrir),
 EV_COMMAND(CM_GUARDAR,Guardar),
 EV_COMMAND(CM_RECUPERAR,Recuperar),
 EV_COMMAND(CM_SALIR,CmExit), // CmExit lo hereda de TWindow
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
          SetMainWindow(new TVentana(0, "Ejemplo de menús en
                              Object-Windows"));
          GetMainWindow()->AssignMenu("E35_MENU");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                                          : TFrameWindow(
}

void TVentana::EvLButtonDown(UINT, TPoint&) {
 MessageBox("Presionó el botón izquierdo del mouse", "Manejo de
```

```
                    Eventos", MB_OK);
}

void TVentana::EvRButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón derecho del mouse", "Manejo de
                    Eventos", MB_OK);
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E35.H

```
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR       104
#define CM_SALIR     1
```

// E35.RC

```
#include "e35.h"
E35_MENU MENU
{
 POPUP "&Archivo"
```

```
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Gurdar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
}
```

```
// E35.DEF  <------ No digite esta línea en el archivo pues produce error
NAME            E35
DESCRIPTION 'Ejemplo de menus en OWL'
EXETYPE         WINDOWS
CODE            PRELOAD MOVEABLE DISCARDABLE
DATA            PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

3.5 Manipulando menús

En esta sección se presentan algunas de las facilidades que ofrece OWL para manipular menús en "tiempo de ejecución". Con este propósito OWL provee la clase TMenu que incluye las siguientes acciones:

1. Agregar y remover marcas tipo "check" ✓ a una opción del menú.
2. Habilitar y deshabilitar un ítem del menú.
3. Agregar ítemes al menú.
4. Remover ítemes del menú.

Estas acciones se ilustran en el ejemplo 3.6, cuyo código se presenta seguidamente. Analice con detalle las partes de código resaltadas en letra negrita e itálica, pues éstas se explican con detalle al final de la sección.

🖫 Ejemplo 3.6. Manipulando un menú con OWL:

```
// E36.CPP
#include <owl\owlpch.h>
```

```
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
#include "e36.h"

class TPrincipal : public TApplication {
  public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};

class TVentana : public TFrameWindow {
          BOOL MensajeHabilitado; // Tipo booleano en Windows
public:
          TVentana(TWindow *Padre, const char far *titulo);
          void EvLButtonDown(UINT, TPoint&);
          void EvRButtonDown(UINT, TPoint&);
          void Nuevo();
          void Abrir();
          void Guardar();
          void Recuperar();
          void Marcar();
          void HabilitarDeshabilitar();
          void Mensaje();
          void Modificar(TCommandEnabler &Aux);
          void Borrar();
          void Agregar();
          DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
  EV_COMMAND(CM_SALIR,CmExit),
  EV_COMMAND(CM_MARCAR,Marcar),
  EV_COMMAND(CM_HABILITAR_DESHABILITAR,HabilitarDeshabilitar),
```

```
  EV_COMMAND(CM_MENSAJE,Mensaje),
  EV_COMMAND_ENABLE(CM_MENSAJE,Modificar),
  EV_COMMAND(CM_BORRAR,Borrar),
  EV_COMMAND(CM_AGREGAR,Agregar),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
        SetMainWindow(new TVentana(0, "Ejemplo de menús en
                                                Object-Windows"));
        GetMainWindow()->AssignMenu("E35_MENU");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)                    : TFrameV
        MensajeHabilitado = FALSE;
}

void TVentana::EvLButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón izquierdo del mouse", "Manejo de
                                                Eventos", MB_OK);
}

void TVentana::EvRButtonDown(UINT, TPoint&) {
  MessageBox("Presionó el botón derecho del mouse", "Manejo de
                                                Eventos", MB_OK);
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
```

```
void TVentana::Marcar() {
        TMenu VarMenu(GetMenu());
        int EstadoActual=
                VarMenu.GetMenuState(CM_MARCAR,MF_CHECKED);
        VarMenu.CheckMenuItem(CM_MARCAR,EstadoActual ?
                                MF_UNCHECKED : MF_CHECKED);
        DrawMenuBar();
}

void TVentana::HabilitarDeshabilitar() {
  MensajeHabilitado=!MensajeHabilitado;
}
void TVentana::Mensaje() {
  MessageBox("Seleccionó en ítem habilitado", "Opciones del menú",
                                                        MB_OK);
}

void TVentana::Modificar(TCommandEnabler &Aux) {
  Aux.Enable(MensajeHabilitado);
}

void TVentana::Borrar() {
        TMenu VarMenu(GetMenu());
        VarMenu.DeleteMenu(CM_ABRIR,MF_BYCOMMAND);
        DrawMenuBar();
}

void TVentana::Agregar() {
        TMenu VarMenu(GetMenu());
        VarMenu.InsertMenu(CM_GUARDAR,MF_BYCOMMAND,
                                        CM_ABRIR,"&Abrir");
        DrawMenuBar();
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E36.H

```
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR       104
#define CM_SALIR     1
#define CM_MARCAR   105
#define CM_HABILITAR_DESHABILITAR          106
#define CM_MENSAJE  107
#define CM_BORRAR   108
#define CM_AGREGAR 109
```

// E36.RC

```
#include "e36.h"
E35_MENU MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
 POPUP "&Manipulando-ítems"
 {
  MENUITEM "&Ítem con/sin Marcar", CM_MARCAR
  MENUITEM "&Habilitar/Deshabilitar el Mensaje de Prueba",
                                          CM_HABILITAR_DESHABILITAR
  MENUITEM "&Mensaje de Prueba", CM_MENSAJE, GRAYED
  MENUITEM "&Borrar el ítem Abrir", CM_BORRAR
  MENUITEM "&Agregar el ítem Abrir", CM_AGREGAR
 }
}
```

// E36.DEF

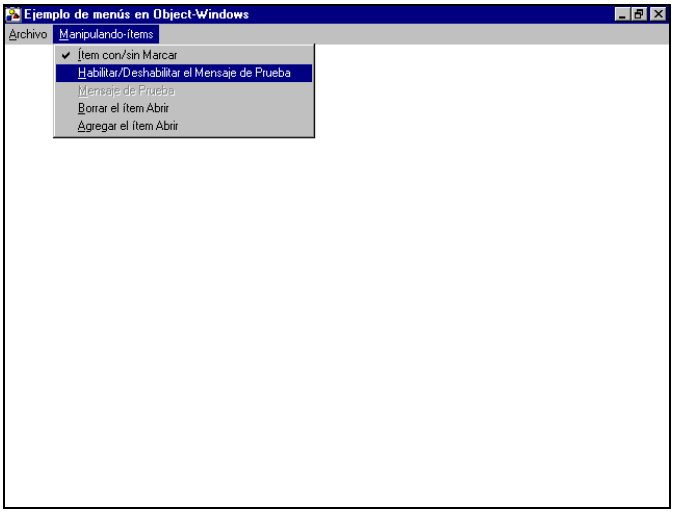
```
NAME                    E36
DESCRIPTION  'Modificando Menus'
EXETYPE                 WINDOWS
CODE PRELOAD MOVEABLE DISCARDABLE
DATA PRELOAD MOVEABLE MULTIPLE
```

```
HEAPSIZE        4096
STACKSIZE       8192
```

La primera opción del submenú Manipulando-ítems es Ítem con/sin Marcar, la cual permite ejecutar el método Marcar() de la clase TVentana. Este método coloca o elimina un ✓ (check-mark) a la opción de menú Ítem con/sin Marcar, tal como se muestra en la Figura 3.13 .

FIGURA 3.13. Uso del método

El código del método Marcar() se presenta en el siguiente fragmento de código:




```
void TVentana::Marcar() {
        TMenu VarMenu(GetMenu());
        int EstadoActual=
                VarMenu.GetMenuState(CM_MARCAR,MF_CHECKED);
        VarMenu.CheckMenuItem(CM_MARCAR,EstadoActual ?
                                MF_UNCHECKED : MF_CHECKED);
        DrawMenuBar();
}
```

Lo primero que se hace en este método es declarar la variable VarMenu de tipo TMenu, mediante la sentencia TMenu VarMenu(GetMenu()); luego obtiene el estado actual del menú usando el método GetMenuState(. . .) de la clase TMenu. Mediante el método CheckMenuItem(. . .) se coloca (o se elimina si ya la tenía) la marca ✓ a la opción. Finalmente se redibuja el menú a través del método DrawMenuBar(); para que el cambio se haga efectivo. El método DrawMenuBar pertenece a la clase TWindow y por lo tanto a la clase TVentana mediante el mecanismo de la herencia.

Para habilitar y deshabilitar una opción de menú OWL provee un mecanismo conocido como "*command enabler event*", el cual se tiene en forma automática para cada opción de menú. Si un EV_COMMAND aparece en la tabla de respuestas asociado a una opción de menú, entonces el ítem del menú aparece automáticamente habilitado. Si se desea la posibilidad de habilitar y deshabilitar una opción de menú, se debe agregar a la Tabla de Respuestas una nueva entrada, en este ejemplo es la siguiente:

```
EV_COMMAND_ENABLE(CM_MENSAJE,Modificar),
```

El método asociado a esta nueva entrada de la tabla de respuestas es el siguiente:

```
void TVentana::Modificar(TCommandEnabler &Aux) {
  Aux.Enable(MensajeHabilitado);
}
```

Nótese que este método ejecuta la función Enable(. . .) que recibe una variable booleana (BOOL de Windows), en este caso el atributo MensajeHabilitado de la clase TVentana. Este atributo se inicializó en FALSE en el constructor de la clase TVentana. Luego el método HabilitarDeshabilitar() cambia su valor cada vez que es invocado, esto mediante el siguiente código:

```
void TVentana::HabilitarDeshabilitar() {
  MensajeHabilitado=!MensajeHabilitado;
```

```
}
```

Ahora, el método anterior habilita y deshabilita el ítem Mensaje de Prueba del menú, tal como se muestra en la Figura 3.13. El código del método asociado a este ítem es el siguiente:

```
void TVentana::Mensaje() {
  MessageBox("Seleccionó en ítem habilitado", "Opciones del menú",
                                                          MB_OK);
}
```

Pero surge una pregunta. Cómo sabe C++ que los métodos Modificar() y HabilitarDeshabiltar() están asociados con el método Mensaje()? La respuesta es simple, C++ conoce este hecho ya que el número asociado con el método Modificar() y el método Mensaje() es el mismo, como puede verse en la Tabla de Respuestas en el siguiente fragmento de código:

```
DEFINE_RESPONSE_TABLE1(TVentana, TFrameWindow)
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
  EV_COMMAND(CM_SALIR,CmExit),
  EV_COMMAND(CM_MARCAR,Marcar),
  EV_COMMAND(CM_HABILITAR_DESHABILITAR,HabilitarDeshabilitar),
  EV_COMMAND(CM_MENSAJE,Mensaje),
  EV_COMMAND_ENABLE(CM_MENSAJE,Modificar),
  EV_COMMAND(CM_BORRAR,Borrar),
  EV_COMMAND(CM_AGREGAR,Agregar),
END_RESPONSE_TABLE;
```

Para eliminar un ítem del menú se hace mediante el método Borrar() de la clase TVentana; el código de este método es el siguiente:

```
void TVentana::Borrar() {
        TMenu VarMenu(GetMenu());
        VarMenu.DeleteMenu(CM_ABRIR,MF_BYCOMMAND);
```

```
            DrawMenuBar();
        }
```

Lo primero que se hace en este método es declarar la variable VarMenu tipo TMenu, luego se elimina el ítem mediante el método DeleteMenu(. . .) de la clase TMenu, este método recibe dos parámetros, en este ejemplo el primero es CM_ABRIR que indica que el ítem que será eliminado es Abrir, el segundo es MF_BYCOMMAND que indica justamente que el primer parámetro debe ser interpretado como la constante asociada al ítem. Finalmente se invoca el método DrawMenuBar() para redibujar el menú.

Para agregar un ítem a un menú se usa una lógica similar, en este ejemplo se hace mediante el método Agregar() cuyo código es el siguiente:

```
void TVentana::Agregar() {
        TMenu VarMenu(GetMenu());
        VarMenu.InsertMenu(CM_GUARDAR,MF_BYCOMMAND,
                                                CM_ABRIR,"&Abrir");
        DrawMenuBar();
}
```

Para agregar el ítem se utiliza el método InsertMenu(. . .) de la clase TMenu. Esta función recibe cuatro parámetros, en este ejemplo el primer parámetro indica que el nuevo ítem será agregado *antes* del ítem Guardar, el segundo parámetro, en este caso es MF_BYCOMMAND que indica justamente que el primer parámetro debe ser interpretado como la constante asociada al ítem, el tercer parámetro es la constante que será asociada con el nuevo item insertado (en este ejemplo CM_ABRIR), por último el cuarto parámetro es una hilera que será el ítem que desplegará, en este caso Abrir.

3.6 Ventanas con accesorios

Usando clases derivadas de TFrameWindow de OWL se pueden agregar *accesorios* ("*gadgets*") a las ventanas, tales como: barras de

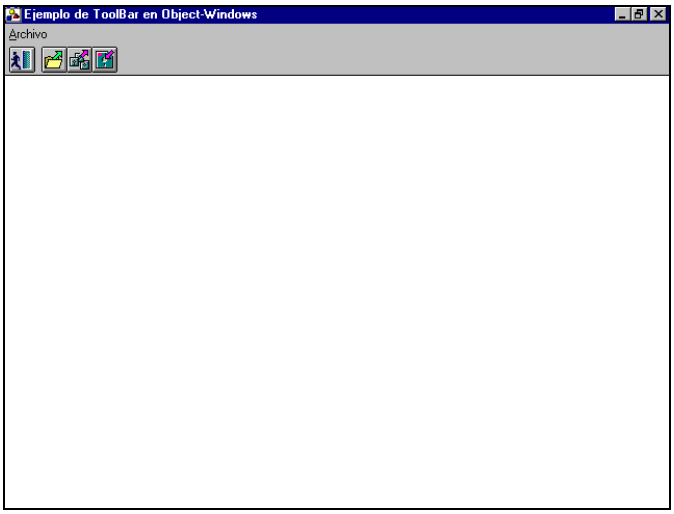
íconos ("toolbars"), barra de estado ("status line") y barras de herramientas ("toolbox"), entre otras. En esta sección se presentan tres ejemplos que ilustran estas decoraciones.

Barra de íconos

Una barra de íconos, conocida en Borland como el "toolbar", usualmente se despliega en la parte superior de la ventana, justo debajo

FIGURA 3.14. Ejemplo de una Barra de Íconos

del menú principal. El objetivo central de la barra de íconos es brindarle al usuario una forma rápida de ejecutar los procesos más importantes y comunes de la aplicación. En la Figura 3.14 se presenta un ejemplo de una barra de íconos. El código de este programa se presenta en ejemplo 3.7.



🖫 Ejemplo 3.7. Barra de íconos con OWL:

```
// E37.CPP
#include <owl\
owlpch.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
#include <owl\statusba.h>
#include "e37.h"

class TPrincipal : public TApplication {
        TControlBar* Barraiconos;
 public:
        TPrincipal() : TApplication() {}
        void InitMainWindow();
};

class TVentana : public TWindow {
public:
        TVentana(TWindow *Padre, const char far *titulo);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Nuevo();
        void Abrir();
        void Guardar();
        void Recuperar();
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
```

```
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
  TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
          de ToolBar en Object-Windows", new TVentana(0,NULL),FALSE);
// Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new TButtonGadget(IDB_GUARDAR,CM_GUARDAR));
// Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E37_MENU");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                          : TWindow(Padre, titulo) {
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
   CloseWindow();
```

```
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E37.H

```
#define CM_NUEVO            101
#define CM_ABRIR            102
#define CM_GUARDAR          103
#define CM_RECUPERAR        104
#define CM_SALIR            1
```

// Identificadores de los íconos en la Barra de íconos

```
#define IDB_SALIR           1001
#define IDB_NUEVO           1101
#define IDB_ABRIR           1102
#define IDB_GUARDAR1103
```

// E37.RC

```
#include "e37.h"
E37_MENU MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
}
```

IDB_SALIR BITMAP {

```
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
```

```
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
 '00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
 '87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
 '46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
 '00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
 '40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
 '46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
 '00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
 '40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
 '46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
 '00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
 '78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
 '46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
 '00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
 '88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_NUEVO BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '00 00 80 00 00 00 00 00 00 00 88 88 00 00 80 0F'
 'BF BF BF BF BF B0 88 88 00 00 80 0B FB FB FB FB'
 'FB F0 88 88 00 00 80 70 BF BF BF BF BF BF 08 88'
 '00 00 80 B0 FB FB FB FB FB FB 08 88 00 00 80 70'
 'BF BF BF BF BF BF 08 88 00 00 80 B7 0B FB FB FB'
 'FB FB F0 88 00 00 80 7B 0F BF BF BF BF BF B0 88'
 '00 00 80 B7 00 00 00 00 00 00 00 88 00 00 80 7B'
 '7B 7B 0A EA 0B 08 88 88 00 00 80 00 B7 B0 00 AE'
 'A0 08 80 88 00 00 88 88 00 08 88 0A EA 08 00 88'
 '00 00 88 88 88 88 88 80 AE A0 A0 88 00 00 88 88'
 '88 88 88 88 0A EA E0 88 00 00 88 88 88 88 88 88'
 '80 AE A0 88 00 00 88 88 88 88 88 0A EA E0 88'
```

```
 '00 00 88 88 88 88 88 80 00 00 00 88 00 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_ABRIR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 00 08 23 10 88 0F'
```

```
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

// E37.DEF
NAME                    E37
DESCRIPTION  'Modificando Menus'
EXETYPE                 WINDOWS
CODE PRELOAD MOVEABLE DISCARDABLE
DATA PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

Para agregar una barra de íconos se deben hacer los siguientes cambios respecto al ejemplo 3.6, presentado en la sección anterior:

- Se debe cambiar la clase base de TVentana de TFrameWindow a TWindow, tal como se muestra en el siguiente fragmento de código.

```
class TVentana : public TWindow {
public:
        TVentana(TWindow *Padre, const char far *titulo);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Nuevo();
        void Abrir();
        void Guardar();
        void Recuperar();
```

```
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};
```

 Sin embargo, para usar accesorios en TVentana, la ventana principal de la clase debe ser creada como una ventana tipo TDecoratedFrame de OWL (o de alguna clase derivada). Esto se hace en el método InitMainWindow(...) de TPrincipal, como se muestra en el siguiente fragmento de código:

```
void TPrincipal::InitMainWindow() {
  TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
        de ToolBar en Object-Windows", new TVentana(0,NULL),FALSE);
   . . . . . .
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E37_MENU");
}
```

 Nótese que el constructor de TDecorateWindow recibe cuatro parámetros, esto son en orden:

1. Un puntero a la ventana madre.
2. Un puntero al título de la ventana.
3. Un puntero a la ventana cliente, en este caso TVentana.
4. Un valor booleano que le indica al compilador si la barra de íconos tiene asociado una barra de estado. En este caso es FALSE ya que todavía en este ejemplo no se ha incluido la barra de estado.

 Incluir por lo menos el archivo controllb.h mediante la instrucción #include <owl\controlb.h>.
 Agregar una variable tipo puntero a TControlBar en la clase TPrincipal, tal como se muestra en el siguiente fragmento de código.

```
class TPrincipal : public TApplication {
        TControlBar* BarraIconos;
 public:
        TPrincipal() : TApplication() {}
        void InitMainWindow();
```

```
};
```

- En el método InitMainWindow(. . .) se crea la barra de íconos mediante el siguiente código:

```
void TPrincipal::InitMainWindow() {
  TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
          de ToolBar en Object-Windows", new TVentana(0,NULL),FALSE);
  // Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
 // Introduce un espacio entre los dos íconos
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new
                          TButtonGadget(IDB_GUARDAR,CM_GUARDAR));
  // Inserta la BarraIconos en la parte superior de la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E37_MENU");
}
```

Cada ícono es en realidad un BitMap, en este ejemplo de tamaño 20 por 20 que debe estar definido en u archivo de recursos .RC, en este caso E37.RC. Cada ícono se inserta a la barra de herramientas mediante el método Insert(...) de la clase TControlBar, por ejemplo:

```
BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
```

Este método recibe dos parámetros, el primero es el identificador del bitmap para ícono en el archivo de recursos, este identificador es una constante entera que debe definirse en el archivo de constantes .H. El segundo parámetro es el identificador del método que será invocado cuando se presione el ícono.

- Debe crearse un bitmap para cada ícono en el archivo de recursos usando el Resource WorkShop, por ejemplo el código generado por WorkShop para el bitmap asociado al ícono de *salir* es siguiente:

```
IDB_SALIR BITMAP    {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
 '00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
 '87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
 '46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
 '00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
 '40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
 '46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
 '00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
 '40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
 '46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
 '00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
 '78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
 '46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
 '00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
 '88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}
```

 Por último se deben definir en el archivo de constantes los nombres asociados a cada bitmap para cada uno de los íconos, así el archivo E37.H queda como sigue:

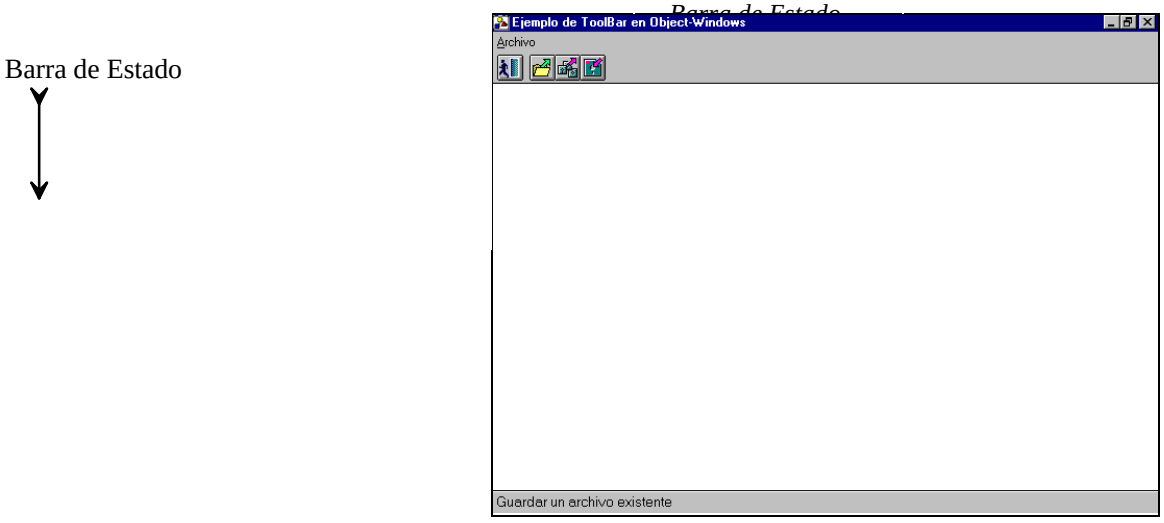
```
// E37.H
#define CM_NUEVO        101
#define CM_ABRIR        102
#define CM_GUARDAR      103
#define CM_RECUPERAR    104
#define CM_SALIR        1
// Identificadores de los íconos en la Barra de íconos
#define IDB_SALIR        1001
#define IDB_NUEVO        1101
#define IDB_ABRIR        1102
```

```
#define IDB_GUARDAR          1103
```

Barra de estado

La barra de estado, conocida en Borland C++ como el "status bar", generalmente se coloca en la parte inferior de la pantalla, se usa para enviar mensajes que aclaran el uso de cajas de diálogo, de los íconos, entre otros. También se puede desplegar el estado de las teclas *Caps Lock*, *Num Lock*, *ScrLock* y de la tecla de *Ins*. En la Figura 3.15 se muestra la barra de estado, cuyo código de presenta en el ejemplo 3.8.

FIGURA 3.15. Ejemplo de una Barra de Estado



Para agregar una barra de estado al ejemplo 3.7 se deben hacer los siguientes cambios:

 Incluya en el programa principal los siguientes archivos:

```
#include <owl\messageb.h>
#include <owl\statusba.h>
```

 Declare una variable tipo puntero a TStatusBar dentro de la clase TPrincipal, como se muestra en el siguiente fragmento de código:

```
class TPrincipal : public TApplication {
        TControlBar* BarraIconos;
        TStatusBar*  BarraEstado;
  public:
        TPrincipal() : TApplication() {}
        void InitMainWindow();
};
```

 Incluya la barra de estado en el método InitMainWindow(...), como se muestra en el siguiente fragmento de código:

```
void TPrincipal::InitMainWindow() {
  TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
       de ToolBar en Object-Windows",new TVentana(0,NULL),TRUE);
 // Construye una BarraIconos
 BarraIconos = new TControlBar(NVentana);
 BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
 BarraIconos->Insert(*new TSeparatorGadget(6));
 BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
 BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
 BarraIconos->Insert(*new
                    TButtonGadget(IDB_GUARDAR,CM_GUARDAR));
// Habilita la  barra de estado a enviar mensajes asociados a los íconos
 BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
 // Inserta la BarraIconos en la ventana NVentana
 NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);

 BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
               TStatusBar::CapsLock | TStatusBar::NumLock |
               TStatusBar::ScrollLock | TStatusBar::Overtype);
 NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
```

```
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E38_MENU");
}
```

- En el código anterior no se debe olvidar colocar TRUE en el cuarto parámetro, cuando se llama el constructor de TDecoratedFrame.
- Otra instrucción importante en el fragmento anterior es:

```
BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
```

pues permite que la barra de estado envíe mensajes asociados a los íconos de la barra de íconos.

- La barra de estado de este ejemplo informa del estado de las teclas *Caps Lock*, *Num Lock*, *ScrLock* y de la tecla de *Ins*, esto se logra mediante la instrucción:

```
BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
                  TStatusBar::CapsLock | TStatusBar::NumLock |
                  TStatusBar::ScrollLock | TStatusBar::Overtype);
```

- Finalmente se debe agregar un "string-table" al archivo de recursos, esto se puede hacer usando el Resource WorkShop o bien digitando el código directamente en el archivo .RC, desde el editor de Borland C++. En el ejemplo de la Figura 3.15 el "string-table" es el siguiente:

```
STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE
{
  CM_SALIR, "Selecciona salir del programa"
  CM_NUEVO, "Crear un archivo nuevo"
  CM_ABRIR, "Abrir un archivo existente"
  CM_GUARDAR, "Guardar un archivo existente"
}
```

Nótese que se tiene un mensaje ligado a cada una de las constantes asociadas a cada uno de los íconos de la barra de íconos. El código completo del ejemplo de la Figura 3.15 es el siguiente:

🖫 Ejemplo 3.8. Barra de estado con OWL:

***//* E38.CPP**

```
#include <owl\owlpch.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
#include <owl\statusba.h>
#include "e38.h"

class TPrincipal : public TApplication {
  TControlBar* BarraIconos;
  TStatusBar*  BarraEstado;
  public:
          TPrincipal() : TApplication() {}
          void InitMainWindow();
};

class TVentana : public TWindow {
public:
          TVentana(TWindow *Padre, const char far *titulo);
          void EvLButtonDown(UINT, TPoint&);
          void EvRButtonDown(UINT, TPoint&);
          void Nuevo();
          void Abrir();
          void Guardar();
          void Recuperar();
          void Salir();
          DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
```

```
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
   TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
           de ToolBar en Object-Windows",new TVentana(0,NULL),TRUE);
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new TButtonGad-
get(IDB_GUARDAR,CM_GUARDAR));
 // Habilita la  barra de estado a enviar mensajes asociados a los íconos
  BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
  // Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
                    TStatusBar::CapsLock | TStatusBar::NumLock |
                    TStatusBar::ScrollLock | TStatusBar::Overtype);
  NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E38_MENU");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo) :                    TWindow(P
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recuperar() {
```

```
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
   CloseWindow();
}

int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E38.RC

```
#include "e38.h"
E38_MENU MENU  {
 POPUP "&Archivo"  {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
}

IDB_SALIR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
 '00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
 '87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
 '46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
 '00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
 '40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
 '46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
 '00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
```

```
 '40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
 '46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
 '00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
 '78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
 '46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
 '00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
 '88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_NUEVO BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '00 00 80 00 00 00 00 00 00 00 88 88 00 00 80 0F'
 'BF BF BF BF BF B0 88 88 00 00 80 0B FB FB FB FB'
 'FB F0 88 88 00 00 80 70 BF BF BF BF BF BF 08 88'
 '00 00 80 B0 FB FB FB FB FB FB 08 88 00 00 80 70'
 'BF BF BF BF BF BF 08 88 00 00 80 B7 0B FB FB FB'
 'FB FB F0 88 00 00 80 7B 0F BF BF BF BF BF B0 88'
 '00 00 80 B7 00 00 00 00 00 00 00 88 00 00 80 7B'
 '7B 7B 0A EA 0B 08 88 88 00 00 80 00 B7 B0 00 AE'
 'A0 08 80 88 00 00 88 88 00 08 88 0A EA 08 00 88'
 '00 00 88 88 88 88 88 80 AE A0 A0 88 00 00 88 88'
 '88 88 88 88 0A EA E0 88 00 00 88 88 88 88 88 88'
 '80 AE A0 88 00 00 88 88 88 88 88 88 0A EA E0 88'
 '00 00 88 88 88 88 88 80 00 00 00 88 00 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_ABRIR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
```

```
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 00 08 23 10 88 0F'
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
```

```
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE {
  CM_SALIR, "Selecciona salir del programa"
  CM_NUEVO, "Crear un archivo nuevo"
  CM_ABRIR, "Abrir un archivo existente"
  CM_GUARDAR, "Guardar un archivo existente"
}
```

// E38.H

```
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR        104
#define CM_SALIR     1
#define IDB_SALIR     1001
#define IDB_NUEVO 1101
#define IDB_ABRIR     1102
#define IDB_GUARDAR1103
```

// E38.DEF

```
NAME                    E38
DESCRIPTION  'Modificando Menús'
EXETYPE                 WINDOWS
CODE PRELOAD MOVEABLE DISCARDABLE
DATA PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

Barra de herramientas

Una barra de herramientas se despliega en una matriz multicolumnar, usualmente se coloca en el borde derecho o izquierdo de la ventana principal. Las barras de herramientas son muy comunes en programas para graficar, presentando íconos para círculos, líneas y otra serie de herramientas de dibujo. En la Figura 3.16 se presenta un ejemplo.

Barra de herramientas ➘

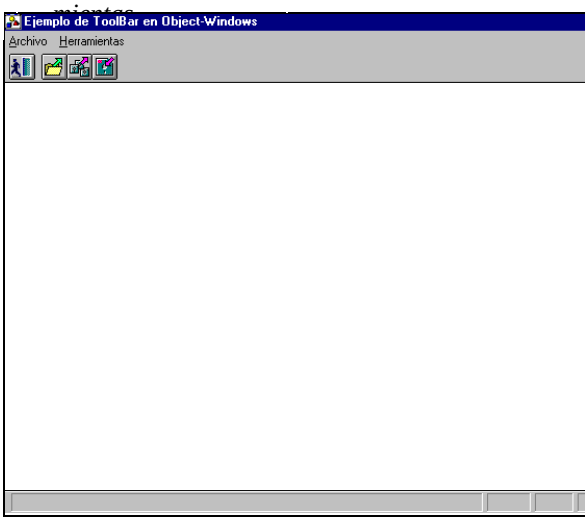


FIGURA 3.16. Ejemplo de una Barra de herramientas

Para agregar una barra de herramientas, a partir del ejemplo 3.8, se deben seguir los siguientes pasos:

- Incluya en el programa principal el archivo:

```
#include <owl\toolbox.h>
```

- Agregue una variable tipo puntero a TToolBox a la clase TPrincipal, tal como se muestra en el siguiente fragmento de código:

```
class TPrincipal : public TApplication {
        TControlBar* BarraIconos;
```

```
            TStatusBar*  BarraEstado;
            TToolBox*    BarraHerramientas;
      public:
            TPrincipal() : TApplication() {}
            void InitMainWindow();
      };
```

- En el método InitMainWindow(...) de la clase TPrincipal cree la barra de herramientas mediante las siguientes instrucciones:

```
void TPrincipal::InitMainWindow() {
 TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo de
          ToolBar en Object-Windows",new TVentana(0,NULL),TRUE);
  . . . . . .
 // Construye una Barra de herramientas
 BarraHerramientas = new TToolBox(NVentana, 1);  // una columna
 BarraHerramientas->Insert(*new TButtonGadget(IDB_CIRCULO,
                                                CM_CIRCULO));
 BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTANGULO,
                                              CM_RECTANGULO));
 BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTA,
                                                CM_RECTA));
 BarraHerramientas->Insert(*new TButtonGadget(IDB_COLOR,CM_COLOR));
 BarraHerramientas->Insert(*new TButtonGadget(IDB_LUPA, CM_LUPA));
 BarraHerramientas->Insert(*new TButtonGadget(IDB_LAPIZ, CM_LAPIZ));
 NVentana->Insert(*BarraHerramientas, TDecoratedFrame::Right);
  . . . . . .
}
```

- La barra de herramientas se inserta en la parte derecha de la ventana principal mediante la siguiente instrucción del método InitMainWindow(...):

```
NVentana->Insert(*BarraHerramientas, TDecoratedFrame::Right);
```

- Se debe crear un bitmap asociado a cada ícono, usando para esto el Resource WorkShop. Usualmente el bitmap es tamaño 20 por 20.
- Por último se deben definir en el archivo de constantes los nombres asociados a cada bitmap, esto para cada uno de los íconos, así el archivo de constantes debe quedar como sigue:

```
// E39.H
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR       104
#define CM_SALIR     1
#define CM_CIRCULO   110
#define CM_RECTANGULO 111
#define CM_RECTA  112
#define CM_COLOR  113
#define CM_LUPA      114
#define CM_LAPIZ     115
// Identificadores de los íconos en la Barra de Íconos
#define IDB_SALIR      1001
#define IDB_NUEVO 1101
#define IDB_ABRIR    1102
#define IDB_GUARDAR1103
// Identificadores de los íconos en la Barra de Herramientas
#define IDB_CIRCULO   1110
#define IDB_RECTANGULO 1111
#define IDB_RECTA  1112
#define IDB_COLOR  1113
#define IDB_LUPA     1114
#define IDB_LAPIZ    1115
```

 El código completo de la barra de herramientas de la Figura 3.16 se presenta en el ejemplo 3.8.

Ejemplo 3.9. Barra de herramientas con OWL:

```
// E39.CPP
#include <owl\owlpch.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
```

```
#include <owl\statusba.h>
#include <owl\toolbox.h>
#include "e39.h"

class TPrincipal : public TApplication {
        TControlBar* BarraIconos;
        TStatusBar*  BarraEstado;
        TToolBox*    BarraHerramientas;
public:
        TPrincipal() : TApplication() {}
        void InitMainWindow();
};

class TVentana : public TWindow {
public:
        TVentana(TWindow *Padre, const char far *titulo);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Nuevo();
        void Abrir();
        void Guardar();
        void Recuperar();
        void Circulo();
        void Rectangulo();
        void Recta();
        void Color();
        void Lupa();
        void Lapiz();
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
  EV_COMMAND(CM_CIRCULO,Circulo),
  EV_COMMAND(CM_RECTANGULO,Rectangulo),
  EV_COMMAND(CM_RECTA,Recta),
  EV_COMMAND(CM_COLOR,Color),
```

```
  EV_COMMAND(CM_LUPA,Lupa),
  EV_COMMAND(CM_LAPIZ,Lapiz),
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
        TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
de ToolBar en Object-Windows",
                                                          new TVent

// Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new TButtonGad-
get(IDB_GUARDAR,CM_GUARDAR));
  // Habilita la  barra de estado a enviar mensajes asociados a los íconos
  BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
  // Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);

  BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
                    TStatusBar::CapsLock | TStatusBar::NumLock |
  TStatusBar::ScrollLock | TStatusBar::Overtype);
  NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
  // Construye una Barra de herramientas
  BarraHerramientas = new TToolBox(NVentana, 1);  // una columna
  BarraHerramientas->Insert(*new TButtonGadget(IDB_CIRCULO,
                                                  CM_CIRCULO));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTANGULO,
                                              CM_RECTANGULO));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTA,
                                                  CM_RECTA));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_COLOR,
                                                  CM_COLOR));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_LUPA, CM_LUPA));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_LAPIZ, CM_LAPIZ));
  NVentana->Insert(*BarraHerramientas, TDecoratedFrame::Right);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E39_MENU");
```

```
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                                    : TWindow(Padre, titulo) {
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Circulo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Rectangulo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recta() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Color() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Lupa() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
```

```
void TVentana::Lapiz() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
   CloseWindow();
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}

// E39.RC
#include "e39.h"
E39_MENU MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
 POPUP "&Herramientas"
 {
  MENUITEM "&Círculo", CM_CIRCULO
  MENUITEM "&Rectángulo", CM_RECTANGULO
  MENUITEM "R&ecta", CM_RECTA
  MENUITEM "C&olor", CM_COLOR
  MENUITEM "&Lupa", CM_LUPA
  MENUITEM "L&ápiz", CM_LAPIZ
 }
}

IDB_SALIR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
```

```
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
 '00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
 '87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
 '46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
 '00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
 '40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
 '46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
 '00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
 '40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
 '46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
 '00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
 '78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
 '46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
 '00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
 '88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_NUEVO BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88'
 '00 00 80 00 00 00 00 00 00 88 88 00 00 80 0F'
 'BF BF BF BF BF B0 88 88 00 00 80 0B FB FB FB FB'
 'FB F0 88 88 00 00 80 70 BF BF BF BF BF BF 08 88'
 '00 00 80 B0 FB FB FB FB FB FB 08 88 00 00 80 70'
 'BF BF BF BF BF BF 08 88 00 00 80 B7 0B FB FB FB'
 'FB FB F0 88 00 00 80 7B 0F BF BF BF BF BF B0 88'
 '00 00 80 B7 00 00 00 00 00 00 00 88 00 00 80 7B'
 '7B 7B 0A EA 0B 08 88 88 00 00 80 00 B7 B0 00 AE'
```

```
 'A0 08 80 88 00 00 88 88 00 08 88 0A EA 08 00 88'
 '00 00 88 88 88 88 88 80 AE A0 A0 88 00 00 88 88'
 '88 88 88 88 0A EA E0 88 00 00 88 88 88 88 88 88'
 '80 AE A0 88 00 00 88 88 88 88 88 88 0A EA E0 88'
 '00 00 88 88 88 88 88 80 00 00 00 88 00 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_ABRIR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
```

```
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 00 08 23 10 88 0F'
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE
{
  CM_SALIR, "Selecciona salir del programa"
  CM_NUEVO, "Crear un archivo nuevo"
  CM_ABRIR, "Abrir un archivo existente"
  CM_GUARDAR, "Guardar un archivo existente"
}

IDB_CIRCULO BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 88 88 87 00 00 78 88 88 88 23 10 88 88'
 '87 00 07 70 00 78 88 88 88 C8 88 88 70 08 88 88'
 '80 07 88 88 00 31 88 87 07 88 88 88 88 70 78 88'
 '10 25 88 80 08 88 88 88 87 80 08 88 99 55 88 70'
```

```
 '88 88 88 88 88 88 07 88 10 25 88 07 88 88 88 88'
 '88 88 70 88 9D 44 88 07 88 88 88 88 88 88 70 88'
 '22 26 88 07 88 88 88 88 88 88 70 88 00 01 88 07'
 '88 88 88 88 88 88 70 88 19 A2 88 07 88 88 88 88'
 '88 88 70 88 20 02 88 70 88 88 88 88 88 88 07 88'
 '44 4C 88 80 08 88 88 88 88 80 08 88 01 03 88 87'
 '07 88 88 88 88 70 78 88 22 44 88 88 70 07 88 88'
 '70 07 88 88 41 26 88 88 87 00 07 70 00 78 88 88'
 '21 91 88 88 88 87 00 00 78 88 88 88 02 54 88 88'
 '88 88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}


IDB_RECTANGULO BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 88 88 88 88 88 88 88 88 88 23 10 87 77'
 '77 77 77 77 77 77 77 78 88 C8 87 00 00 00 00 00'
 '00 00 01 78 00 31 87 90 00 00 00 00 00 00 00 78'
 '10 25 87 00 77 77 77 77 77 77 00 78 99 55 87 90'
 '78 88 88 88 88 87 00 78 10 25 87 00 78 88 88 88'
 '88 87 00 78 9D 44 87 90 78 88 88 88 88 87 00 78'
 '22 26 87 00 78 88 88 88 88 87 00 78 00 01 87 90'
 '78 88 88 88 88 87 00 78 19 A2 87 00 78 88 88 88'
 '88 87 00 78 20 02 87 90 78 88 88 88 88 87 00 78'
 '44 4C 87 00 77 77 77 77 77 77 00 78 01 03 87 90'
 '00 00 00 00 00 00 00 78 22 44 87 19 09 09 09 09'
 '09 09 09 78 41 26 87 77 77 77 77 77 77 77 77 78'
 '21 91 88 88 88 88 88 88 88 88 88 88 02 54 88 88'
 '88 88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

IDB_RECTA BITMAP  {
```

```
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 88 88 88 88 88 88 88 88 88 65 F0 88 88'
 '88 88 88 88 88 88 88 88 65 F0 88 88 88 88 88 88'
 '88 88 88 88 65 F0 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 88 88 88 88 88 88 88 88 88 65 F0 88 88'
 '88 88 88 88 88 88 88 88 A2 C0 88 88 88 88 88 88'
 '88 88 88 88 00 00 88 88 88 88 88 88 88 88 88 88'
 '88 88 88 88 88 88 88 88 88 88 88 88 04 6D 80 00'
 '00 00 00 00 00 00 00 08 08 80 88 88 88 88 88 88'
 '88 88 88 88 01 09 88 88 88 88 88 88 88 88 88 88'
 '00 88 88 88 88 88 88 88 88 88 88 88 00 E0 88 88'
 '88 88 88 88 88 88 88 88 2A 20 88 88 88 88 88 88'
 '88 88 88 88 22 A4 88 88 88 88 88 88 88 88 88 88'
 '04 08 88 88 88 88 88 88 88 88 88 88 99 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_COLOR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '00 00 88 88 88 88 88 88 88 88 88 88 00 00 8F FF'
 'FF FF FF FF FF FF FF F8 00 00 80 00 00 00 00 00'
 '00 00 00 F8 00 00 80 BB BB BB BB BB BB BB BB F8'
 '00 00 80 BB BB BB BB BB BB BB BB F8 00 00 80 AA'
 'AA AA AA AA AA AA AA F8 00 00 80 AA AA AA AA AA'
 'AA AA AA F8 00 00 80 AA AA AA AA AA AA AA AA F8'
 '00 00 80 CC CC CC CC CC CC CC CC F8 00 00 80 CC'
```

```
 'CC CC CC CC CC CC CC F8 00 00 80 CC CC CC CC CC'
 'CC CC CC F8 00 00 80 CC CC CC CC CC CC CC CC F8'
 '00 00 80 99 99 99 99 99 99 99 99 F8 00 00 80 99'
 '99 99 99 99 99 99 99 F8 00 00 80 99 99 99 99 99'
 '99 99 99 F8 00 00 80 99 99 99 99 99 99 99 99 F8'
 '00 00 80 99 99 99 99 99 99 99 99 F8 00 00 80 00'
 '00 00 00 00 00 00 00 F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_LUPA BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 B8 77 88 88 88 88 70 F0'
 '00 00 88 88 8B 37 00 00 00 87 07 07 00 00 88 88'
 '88 00 77 87 87 00 70 78 00 00 88 88 80 77 88 8F'
 '88 87 07 88 00 00 88 88 07 7F 83 77 FF F8 70 88'
 '00 00 88 88 07 F8 BB 37 78 FF 80 88 00 00 88 80'
 '78 3B BB B3 77 8F F7 08 00 00 88 80 73 88 BB 33'
 '38 8F F8 08 00 00 88 80 78 38 8B B3 77 88 F8 08'
 '00 00 88 80 73 88 88 BB 37 78 F8 08 00 00 88 80'
 '73 3B BB BB B7 77 F8 08 00 00 88 80 7F 83 BB B3'
 '77 88 F7 08 00 00 88 88 0F F8 3B BB 37 78 80 88'
 '67 14 88 88 07 FF 33 BB B3 77 70 88 8A 44 88 88'
 '00 78 88 3B 88 37 07 88 CD EB 88 88 88 00 77 77'
 '77 00 77 78 41 98 88 88 88 88 00 00 00 33 B7 77'
 'E9 F3 88 88 88 88 88 88 8B BB BB 88 93 30 88 88'
 '88 88 88 88 88 BB BB B8 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_LAPIZ BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
```

```
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 80 44 47 77 78 78 78 78 88 88'
 '00 00 87 00 78 88 88 88 88 88 48 88 00 00 88 00'
 '00 88 88 88 88 88 87 88 00 00 88 70 07 07 88 88'
 '88 88 48 88 00 00 88 80 77 70 08 88 88 84 88 88'
 '00 00 88 88 08 37 70 78 88 87 88 88 00 00 88 88'
 '74 83 77 00 88 88 87 87 00 00 88 88 80 38 37 74'
 '07 88 88 88 00 00 88 88 88 08 83 77 44 08 88 88'
 '00 00 88 88 88 74 88 37 74 70 88 88 00 00 88 88'
 '88 80 88 87 44 47 08 88 00 00 88 88 88 88 08 B7'
 '78 34 70 88 00 00 88 88 88 88 70 07 B3 33 47 08'
 '67 14 88 88 88 88 80 3B FB 33 34 70 8A 44 88 88'
 '88 88 88 03 BB B3 33 47 CD EB 88 88 88 88 88 80'
 '3B FB 33 34 41 98 88 88 88 88 88 08 03 BF B3 33'
 'E9 F3 88 88 88 88 88 88 80 3B BB 33 93 30 88 88'
 '88 88 88 88 88 03 BF B3 26 69 88 88 88 88 88 88'
 '88 80 3B FB 32 26'
}
```

// E39.H

```
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR      104
#define CM_SALIR     1
#define CM_CIRCULO   110
#define CM_RECTANGULO 111
#define CM_RECTA  112
#define CM_COLOR  113
#define CM_LUPA      114
#define CM_LAPIZ     115
// Identificadores de los íconos en la Barra de Íconos
#define IDB_SALIR      1001
#define IDB_NUEVO 1101
#define IDB_ABRIR    1102
#define IDB_GUARDAR1103
#define IDB_CIRCULO   1110
#define IDB_RECTANGULO 1111
#define IDB_RECTA  1112
```

```
#define IDB_COLOR  1113
#define IDB_LUPA       1114
#define IDB_LAPIZ      1115

// E39.DEF
NAME                    E39
DESCRIPTION  'Modificando Menus'
EXETYPE                 WINDOWS
CODE PRELOAD MOVEABLE DISCARDABLE
DATA PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

3.7 Cajas de diálogo

Existen dos clases de cajas de diálogo; las provee Windows y las que el programador puede crear. Para "administrar" estas cajas de diálogo OWL provee una clase denominada TDialog. Como las cajas de diálogo son un tipo especial de ventanas, naturalmente TDialog es una clase derivada de TWindow.

El aspecto más complicado y hasta cierto punto "misterioso" es cómo enviar y recuperar (leer) información a través de las cajas de diálogo. En este aspecto OWL 2.0 provee una mejora significativa respecto a OWL 1.0 al introducir un mecanismo llamado "*mecanismo para la transferencia de datos*" (*data transfer mechanism*), el cual simplifica enormemente el mecanismo para enviar y recibir datos de la caja de diálogo.

Las cajas de diálogo son de dos tipos: "***Modal Dialogs***" los cuales se caracterizan porque se requiere que el usuario cierre la caja de diálogo antes de que pueda seleccionar cualquier otro comando del programa, y "***Modaless Dialogs***" que se caracterizan porque el usuario puede seleccionar otros comandos del programa sin necesidad de cerrar previamente la caja de diálogo. En esta sección presentamos únicamente cajas de diálogo tipo Modal.

Para agregar una caja de diálogo a una aplicación se debe seguir los siguientes pasos:

 El primer paso es diseñar la caja de diálogo mediante el Resource Workshop, tal como se muestra en la Figura 3.17. El Resource Workshop permite diseñar cajas de diálogo tipo Windows Estándar o bien tipo Borland, como la que se presenta en la Figura 3.17. Consulte el manual de Resource Workshop para más detalles.

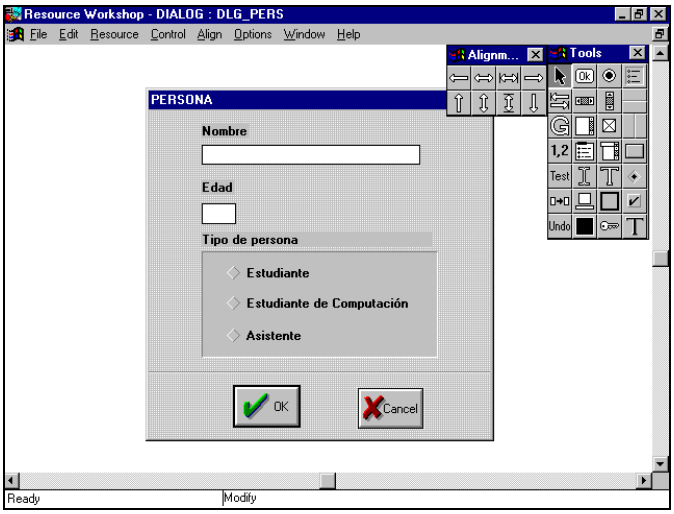
FIGURA 3.17. Creación de una caja de logo en Workshop.

 Luego se debe crear una clase derivada de TDialog (de OWL), tal como se muestra en el siguiente fragmento de código:

```
class TDialogoPersona: public TDialog {
public:
   TDialogoPersona(TWindow* Madre, TResId IdenRecur);
};
```

FIGURA 3.17. Creación de una caja de diálogo en Workshop.

o bien tipo



 Después de crear la clase derivada de TDialog se debe declarar una instancia. En el siguiente fragme

 nto de código, mediante el constructor, se hace la conexión entre la clase y el nombre asociado a la caja de diálogo en el archivo de recursos. Luego se ejecuta la caja de diálogo mediante el método Execute() de la clase TDialog.

```
TDialogoPersona *TDPer = new TDialogoPersona(this,"DLG_PERS");
if(TDPer->Execute()==IDOK) {
     . . . . .
}
```

 OWL 2.0 provee una forma muy simple de transferencia automática de datos y de los controles de la caja de diálogo. Para esto se debe definir una estructura que incluya un campo para cada control en la caja de diálogo. Por ejemplo para la caja de diálogo que aparece en la Figura 3.17 se definió la siguiente estructura:

```
struct TransDatosPersona {
        char TNombre[31];
        char TEdad[4];
        WORD TEsEstudiante;
```

```
        WORD TEsEstudianteCompu;
        WORD TEsAsistente;    // WORD es un tipo de Windows
} TPersona;
```

- Finalmente en el constructor de la clase derivada de TDialog se debe asociar cada control con una constante definida en el archivo .H de constantes, además se deben inicializar los controles de la caja de diálogo, como se muestra en el siguiente fragmento de código:

```
TDialogoPersona::TDialogoPersona(TWindow* Madre,
                    TResId IdenRecur)  : TDialog(Madre, IdenRecur) {
  new TEdit(this, IDC_NOMBRE,31);
  new TEdit(this, IDC_EDAD,4);
  new TRadioButton(this,IDC_ESTUDIANTE);
  new TRadioButton(this,IDC_ESTUDIANTECOMPU);
  new TRadioButton(this,IDC_ASISTENTE);
  strcpy(TPersona.TNombre,"Digite el nombre");
  strcpy(TPersona.TEdad,"0");
  TPersona.TEsEstudiante=TRUE;
  TPersona.TEsEstudianteCompu=FALSE;
  TPersona.TEsAsistente=FALSE;
  SetTransferBuffer(&TPersona);
}
```

La última instrucción del anterior fragmento de código SetTransferBuffer(&TPersona) se encarga de transferir la información almacenada en la estructura TPersona a los campos de la caja de diálogo. Además establece una liga entre la caja de diálogo y la estructura TPersona, la cual permite que, cuando la caja de diálogo es cerrada, los datos que se digitaron en la caja de diálogo queden almacenados en TPersona.

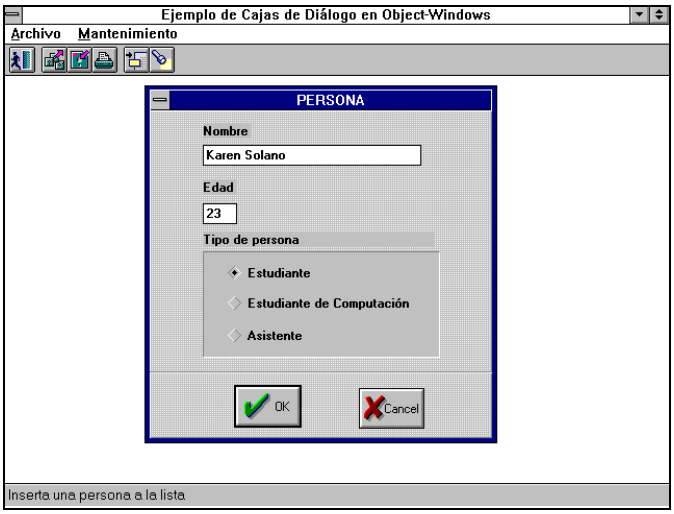
Debido a lo anterior es que en el programa se manipulan los datos de la caja de diálogo a través de la Estructura TPersona, como se presenta en el siguiente fragmento de código:

```
MessageBox(TPersona.TEdad, "Edad", MB_OK);
if(TPersona.TEsEstudiante==TRUE) . . . . . .
        . . . . . .
```

En los siguientes ejemplos se pretende establecer una interfaz gráfica para el ejemplo 2.12.

En el ejemplo 3.10 se presenta una caja de diálogo que permite insertar los datos para la clase Persona y luego determinar si esta Persona es Estudiante, EstuCompu o Asistente, estas clases son las presentadas en el ejemplo 2.7. Tal como se muestra en la Figura 3.18. Sin embargo, en este ejemplo no se insertan aún los datos a la lista polimórfica de Personas (esto se hace en el ejemplo 3.11). Lo único que se hace, mediante la opción Insertar del menú principal, es leer los datos de una Persona en una caja de diálogo, para luego, cuando la caja de diálogo es cerrada, desplegar su nombre, su edad y el tipo de Persona que es (Estudiante, EstuCompu o Asistente).

FIGURA 3.18. Ejemplo de una caja de diálogo.



Ejemplo 3.10. Cajas de diálogo con OWL:

```
// E3_10.CPP
#include <owl\owlpch.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
```

```
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
#include <owl\statusba.h>
#include <owl\dialog.h>
#include <owl\radiobut.h>
#include "e3_10.h"

class TPrincipal : public TApplication {
        TControlBar* BarraIconos;
        TStatusBar*  BarraEstado;
public:
        TPrincipal() : TApplication() {}
        void InitMainWindow();
};

class TVentana : public TWindow {
public:
        TVentana(TWindow *Padre, const char far *titulo);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Abrir();
        void Guardar();
        void Imprimir();
        void Insertar();
        void Buscar();
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_IMPRIMIR,Imprimir),
  EV_COMMAND(CM_INSERTAR,Insertar),
  EV_COMMAND(CM_BUSCAR,Buscar),
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;
```

```
struct TransDatosPersona {
        char TNombre[31];
        char TEdad[4];
        WORD TEsEstudiante;
        WORD TEsEstudianteCompu;
        WORD TEsAsistente;
} TPersona;

class TDialogoPersona: public TDialog {
public:
  TDialogoPersona(TWindow* Madre, TResId IdenRecur);
};

void TPrincipal::InitMainWindow() {
  TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
                                  de Cajas de Diálogo en Object-Windows",
                                                          new TVentana(0,NULL),T
// Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new
                          TButtonGadget(IDB_GUARDAR,CM_GUARDAR));
  BarraIconos->Insert(*new
                           TButtonGadget(IDB_IMPRIMIR,CM_IMPRIMIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));

  BarraIconos->Insert(*new
                     TButtonGadget(IDB_INSERTAR,CM_INSERTAR));
  BarraIconos->Insert(*new TButtonGadget(IDB_BUSCAR,CM_BUSCAR));
  // Habilita la  barra de estado a enviar mensajes asociados a los íconos
  BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
  // Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
           TStatusBar::CapsLock | TStatusBar::NumLock |
           TStatusBar::ScrollLock | TStatusBar::Overtype);
  NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
  SetMainWindow(NVentana);
```

```
  GetMainWindow()->AssignMenu("MENU_PRIN");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                              : TWindow(Padre, titulo) {
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Imprimir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Insertar() {
 TDialogoPersona *TDPer = new TDialogoPersona(this,"DLG_PERS");
 if(TDPer->Execute()==IDOK) {
   MessageBox("Salió presionando OK", "Resultados de la Caja de
                                                    Diálogo", MB_OK);
  MessageBox(TPersona.TNombre, "Nombre", MB_OK);
  MessageBox(TPersona.TEdad, "Edad", MB_OK);
  if(TPersona.TEsEstudiante==TRUE)
          MessageBox("Es un Estudiante", "Resultados de la Caja de
                                                    Diálogo", MB_OK);
  if(TPersona.TEsEstudianteCompu==TRUE)
           MessageBox("Es un Estudiante de Computación",
                          "Resultados de la Caja de Diálogo", MB_OK);
  if(TPersona.TEsAsistente==TRUE)
           MessageBox("Es un Asistente", "Resultados de la Caja de
                    Diálogo", MB_OK);
 }
 else
   MessageBox("Salió presionando CANCEL", "Resultados de la
                                              Caja de Diálogo", MB_OK);
}
```

```
void TVentana::Buscar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
        CloseWindow();
}

TDialogoPersona::TDialogoPersona(TWindow* Madre, TResId IdenRecur)
                                        : Dialog(Madre, IdenRecur) {
  new TEdit(this, IDC_NOMBRE,31);
  new TEdit(this, IDC_EDAD,4);
  new TRadioButton(this,IDC_ESTUDIANTE);
  new TRadioButton(this,IDC_ESTUDIANTECOMPU);
  new TRadioButton(this,IDC_ASISTENTE);
  strcpy(TPersona.TNombre,"Digite el nombre");
  strcpy(TPersona.TEdad,"0");
  TPersona.TEsEstudiante=TRUE;
  TPersona.TEsEstudianteCompu=FALSE;
  TPersona.TEsAsistente=FALSE;
  SetTransferBuffer(&TPersona);
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}

// E3_10.H
#define CM_ABRIR       101
#define CM_GUARDAR 102
#define CM_IMPRIMIR 103
#define CM_INSERTAR 201
#define CM_BUSCAR   202
#define CM_SALIR       10
#define IDB_SALIR       1001
#define IDB_ABRIR      1101
#define IDB_GUARDAR 1102
#define IDB_IMPRIMIR  1103
```

```
#define IDB_BUSCAR   1104
#define IDB_INSERTAR          1105
#define PERSONA       2001
#define IDC_NOMBRE  101
#define IDC_EDAD       102
#define IDC_ESTUDIANTE       103
#define IDC_ESTUDIANTECOMPU       104
#define IDC_ASISTENTE         105

// E3_10.RC
#include "e3_10.h"
MENU_PRIN MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Abrir...", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM SEPARATOR
  MENUITEM "&Imprimir...", CM_IMPRIMIR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
 POPUP "&Mantenimiento"
 {
  MENUITEM "&Insertar...", CM_INSERTAR
  MENUITEM "&Buscar...", CM_BUSCAR
 }
}

IDB_SALIR BITMAP  {
'42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
'00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
'00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
'00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
'00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
'00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
'00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
'00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
'00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
'87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
'46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
```

```
'00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
'40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
'46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
'00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
'40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
'46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
'00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
'78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
'46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
'00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
'88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
'88 88 88 88 00 00'
}

IDB_ABRIR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP {
```

```
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 08 23 10 88 0F'
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE
{
  CM_SALIR, "Selecciona salir del programa"
  CM_ABRIR, "Abrir un archivo de personas"
  CM_GUARDAR, "Guardar el archivo personas"
  CM_IMPRIMIR, "Imprime el archivo de personas"
  CM_INSERTAR, "Inserta una persona a la lista"
  CM_BUSCAR, "Busca una persona de la lsita"
}

IDB_IMPRIMIR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
```

```
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 70 00 00 00 00 00 00 07 88 65 F0 88 06'
 '76 76 76 76 76 76 70 88 65 F0 80 00 00 00 00 00'
 '00 00 00 08 65 F0 80 66 66 66 66 66 66 66 66 08'
 '65 F0 80 66 06 06 06 66 66 00 66 08 65 F0 80 66'
 '66 66 66 66 66 66 66 08 A2 C0 80 88 F8 F8 F8 F8'
 'F8 F8 F8 08 00 00 88 00 00 00 00 00 00 00 00 88'
 '88 88 88 80 60 FF FF FF FF 06 08 88 04 6D 88 80'
 '00 F0 07 07 0F 00 08 88 88 80 88 88 80 FF FF FF'
 'FF 08 88 88 01 09 88 88 80 F0 70 88 88 08 88 88'
 '00 88 88 88 80 FF FF F0 00 08 88 88 00 E0 88 88'
 '80 F0 70 F0 F0 88 88 88 2A 20 88 88 80 FF FF F0'
 '08 88 88 88 22 A4 88 88 80 00 00 00 88 88 88 88'
 '04 08 88 88 88 88 88 88 88 88 88 88 99 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

DLG_PERS DIALOG 80, 39, 188, 193
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION                    | MENU
CLASS "bordlg"
CAPTION "PERSONA"
FONT 8, "MS Sans Serif"
{
 EDITTEXT IDC_NOMBRE, 29, 20, 120, 12
 EDITTEXT IDC_EDAD, 29, 55, 19, 12
 CONTROL "Estudiante", IDC_ESTUDIANTE, "BorRadio",
BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 43,
91, 46, 10
 CONTROL "Estudiante de Computación", IDC_ESTUDIANTECOMPU, "Bo-
rRadio", BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP, 43, 109, 101, 10
 CONTROL "Asistente", IDC_ASISTENTE, "BorRadio",
BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 43,
128, 41, 10
 CONTROL "OK", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 46, 162, 37, 25
 CONTROL "Cancelar", IDCANCEL, "BorBtn", BS_PUSHBUTTON |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 114, 163, 38, 25
```

```
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD |
WS_VISIBLE, -1, 154, 189, 3
 LTEXT "Nombre", -1, 29, 8, 26, 8
 LTEXT "Edad", -1, 29, 41, 18, 8
 CONTROL "", -1, "BorShade", BSS_GROUP | BSS_CAPTION | BSS_LEFT |
WS_CHILD | WS_VISIBLE | WS_GROUP, 29, 83, 130, 63
 LTEXT "Tipo de persona", -1, 29, 72, 127, 8, NOT WS_GROUP
}

IDB_INSERTAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 4F 8F 8F 8F 8F 8F'
 '00 00 88 88 88 88 48 88 88 88 78 78 00 00 88 88'
 '77 77 77 77 77 78 8F 8F 00 00 88 84 44 44 44 44'
 '44 78 78 78 00 00 88 84 FB FB FB FB F4 78 8F 8F'
 '00 00 88 84 BF BF BF BF B4 78 78 78 00 00 88 84'
 'FB FB FB FB F4 78 8F 8F 00 00 88 84 BF BF BF BF'
 'B4 78 78 78 00 00 88 84 FB FB FB FB F4 78 8F 8F'
 '00 00 88 84 44 44 44 44 44 88 78 78 00 00 88 88'
 '88 88 4F 8F 8F 8F 8F 8F 00 00 88 80 88 88 48 F8'
 'F8 78 78 78 00 00 88 80 08 88 4F 8F 8F 8F 8F 8F'
 '67 14 00 00 00 88 48 F8 F8 F8 F8 F8 8A 44 00 00'
 '00 88 4F 8F 8F 8F 8F 8F CD EB 88 80 08 88 48 F8'
 'F8 F8 F8 F8 41 98 88 80 88 88 44 44 44 44 44 44'
 'E9 F3 88 88 88 88 88 88 88 88 88 88 93 30 88 88'
 '88 88 88 88 88 88 88 88 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_BUSCAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
```

```
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 FB FB FF FF FB'
 '44 4C 88 88 88 88 8F BF BF FF BF FF 23 10 88 88'
 '88 88 88 FF FF BF FF FF 88 C8 88 88 88 88 8F BF'
 'BF FF BF FB 00 31 88 88 88 88 88 FB FB FF FF FF'
 '10 25 88 88 88 88 84 44 7F BF FF FB 99 55 88 88'
 '88 88 44 73 44 FB FB FF 10 25 88 88 88 84 47 8F'
 '34 7F BF BB 9D 44 88 88 88 44 74 8F F7 4B FF FF'
 '22 26 88 88 88 47 47 4F F7 4F BF BF 00 01 88 88'
 '88 44 7F 84 44 78 F8 F8 19 A2 88 88 84 47 FF F8'
 '74 88 88 88 20 02 88 88 44 77 48 87 48 88 88 88'
 '44 4C 88 84 47 F8 74 44 88 88 88 88 01 03 88 44'
 '7F 88 48 88 88 88 88 88 22 44 84 47 F8 84 88 88'
 '88 88 88 88 41 26 84 7F 88 48 88 88 88 88 88 88'
 '21 91 84 78 84 88 88 88 88 88 88 88 02 54 88 44'
 '48 88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

NAME                    E3_10
DESCRIPTION  'Modificando Menus'
EXETYPE                 WINDOWS
CODE PRELOAD MOVEABLE DISCARDABLE
DATA PRELOAD MOVEABLE MULTIPLE
HEAPSIZE        4096
STACKSIZE       8192
```

Tabla de Eventos asociada a una caja de diálogo

En el ejemplo 3.11 se ilustra la forma en que una caja de diálogo puede tener asociada una Tabla de Respuestas a Eventos. En este ejemplo se usan cajas de diálogo para insertar Estudiantes, EstCompu y Asistentes a la lista polimórfica del ejemplo 2.12. También se usan cajas de diálogo para desplegar el contenido de la lista polimórfica en la pantalla.

La clase más importante que administra este proceso es TDialogoDespliega, cuyo código se presenta seguidamente:

```
class TDialogoDespliega : public TDialog {
public:
  TDialogoDespliega(TWindow* Madre, TResId IdenRecur);
  void Siguiente();
  void Mas();
  DECLARE_RESPONSE_TABLE(TDialogoDespliega);
};

DEFINE_RESPONSE_TABLE1(TDialogoDespliega, TDialog)
  EV_COMMAND(IDC_SIGUIENTE, Siguiente),
  EV_COMMAND(IDC_MAS, Mas),
END_RESPONSE_TABLE;
```

Nótese que la clase TDialogoDespliega tiene una *Tabla de Respuestas* asociada, la cual se encarga de ligar botones de la caja de diálogo con métodos propios de la clase. Por ejemplo se asocia un botón al método Siguiente(), este método permite desplegar la siguiente "Persona" que está en la lista en una caja de diálogo. También se asocia un botón con el método Mas(), el cual permite desplegar información adicional de la Persona, dependiendo si ésta es Estudiante, EstuCompu o Asistente. El funcionamiento de este ejemplo se ilustra en la Figura 3.19.

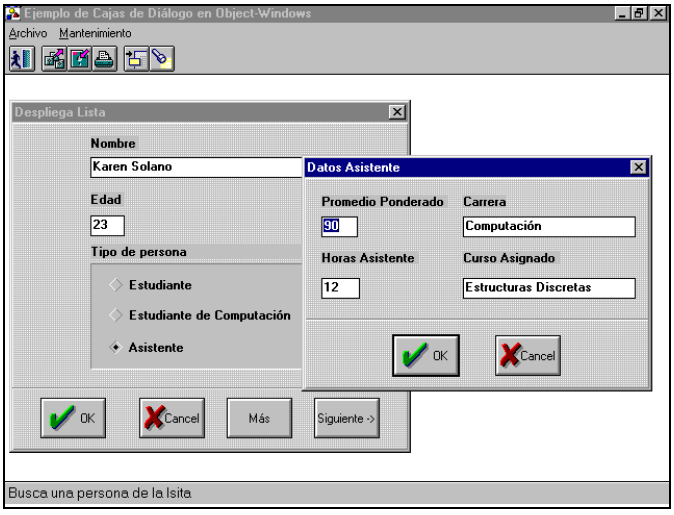


FIGURA 3.19. Ejemplo de una caja de diálogo con Tabla de Eventos asociada.

En este ejemplo también se introduce el uso de un par de cajas de diálogo de tipo Windows-Estándar, las cuales ya están pre-definidas en Windows. Estas cajas son de suma importancia en muchas aplicaciones, pues permiten seleccionar mediante una caja de diálogo el nombre de un archivo. En el ejemplo 3.11 se utilizan para escoger el nombre de un archivo en el que se guardará o recuperará de disco la lista polimórfica.

En la figura 3.20 se presenta la caja de diálogo que aparece cuando se escoge la opción Guardar en el ejemplo 3.11. Como se puede ver ésta es la caja de diálogo estándar que utiliza Windows para buscar nombres

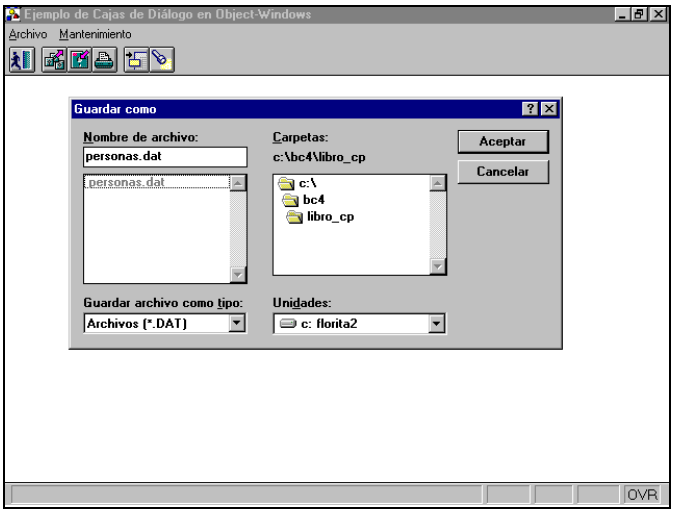


FIGURA 3.20. Caja de diálogo para seleccionar el nombre de un archivo.

de archivos.

La caja de diálogo de la Figura 3.20 se ejecuta con el método Cargar de la clase TVentana, cuyo código es el siguiente:

```
void TVentana::Guardar() {
        if ((TFileSaveDialog(this, *DatosArchivo)).Execute() == IDOK)  {
                UnaLista.Guardar(Arch,DatosArchivo->FileName);
        }
}
```

Para que este código compile correctamente es necesario declarar la variable DatosArchivo en la clase TVentana como se muestra en el siguiente fragmento de código:

```
class TVentana : public TWindow {
        TOpenSaveDialog::TData *DatosArchivo;
public:
        . . . . . .
        . . . . . .
};
```

El método Abrir de la clase TVentana es muy similar al método Guardar; analice el siguiente fragmento de código:

```
void TVentana::Abrir() {
  if ((TFileOpenDialog(this, *DatosArchivo)).Execute() == IDOK) {
     UnaLista.Recuperar(Arch,DatosArchivo->FileName);
  }
}
```

El código completo se presenta en el ejemplo 3.11. Nótese que el código de las clases Persona, Estudiante, EstuCompu, Asistente, Objeto y ListaObjetos han sufrido algunas modificaciones, esto con el fin de adaptarlas al uso de cajas de diálogo y de eliminar el código DOS que ahora es innecesario.

🖫 Ejemplo 3.11. Cajas de diálogo con Tabla de Respuestas:

```
 // E3_11.CPP
#include <owl\owlpch.h>
#include <owl\framewin.h>
#include <owl\edit.h>
#include <owl\menu.h>
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
```

```
#include <owl\statusba.h>
#include <owl\dialog.h>
#include <owl\radiobut.h>
#include <owl\opensave.h>
#include <stdlib.h>
#include "e3_11.h"
#include "pers311.hpp"
#include "lpers311.hpp"

// Variables Globales
Objeto *PObjeto;
Persona *PPersona;
Asistente *PAsistente;
Estudiante *PEstudiante;
EstCompu *PEstCompu;
ListaObjetos UnaLista;
FILE *Arch;

class TPrincipal : public TApplication {
            TControlBar* BarraIconos;
            TStatusBar*  BarraEstado;
public:
            TPrincipal() : TApplication() {}
            void InitMainWindow();
};

class TVentana : public TWindow {
          TOpenSaveDialog::TData *DatosArchivo;
public:
          TVentana(TWindow *Padre, const char far *titulo);
          ~TVentana() {delete DatosArchivo;}
          void EvLButtonDown(UINT, TPoint&);
          void EvRButtonDown(UINT, TPoint&);
          void Abrir();
          void Guardar();
          void Imprimir();
          void Insertar();
          void Buscar();
          void Salir();
          DECLARE_RESPONSE_TABLE(TVentana);
};
```

```
DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_IMPRIMIR,Imprimir),
  EV_COMMAND(CM_INSERTAR,Insertar),
  EV_COMMAND(CM_BUSCAR,Buscar),
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;

struct TransDatosPersona {
        char TNombre[30];
        char TEdad[4];
        WORD TEsEstudiante;
        WORD TEsEstudianteCompu;
        WORD TEsAsistente;
} TPersona;

class TDialogoPersona: public TDialog {
public:
  TDialogoPersona(TWindow* Madre, TResId IdenRecur);
};

struct TransDatosEstudiante {
        char TNota1[4];
        char TNota2[4];
} TEstudiante;

class TDialogoEstudiante: public TDialog {
public:
  TDialogoEstudiante(TWindow* Madre, TResId IdenRecur);
};

struct TransDatosEstCompu {
        char TNota1[4];
        char TNota2[4];
        char TNota3[4];
} TEstCompu;

class TDialogoEstCompu: public TDialog {
public:
```

```
  TDialogoEstCompu(TWindow* Madre, TResId IdenRecur);
};

struct TransDatosAsistente {
        char TPromedio[4];
        char THorasAsis[4];
        char TCarrera[50];
        char TCurso[50];
} TAsistente;

class TDialogoAsistente : public TDialog {
public:
  TDialogoAsistente(TWindow* Madre, TResId IdenRecur);
};

class TDialogoDespliega : public TDialog {
public:
  TDialogoDespliega(TWindow* Madre, TResId IdenRecur);
  void Siguiente();
  void Mas();
  DECLARE_RESPONSE_TABLE(TDialogoDespliega);
};

DEFINE_RESPONSE_TABLE1(TDialogoDespliega, TDialog)
  EV_COMMAND(IDC_SIGUIENTE, Siguiente),
  EV_COMMAND(IDC_MAS, Mas),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
 TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
 de Cajas de Diálogo en Object-Windows",new TVentana(0,NULL),TRUE);
// Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new                                    TButtonGa
  BarraIconos->Insert(*new                                    TButtonGa
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new                                    TButtonGa
  BarraIconos->Insert(*new TButtonGadget(IDB_BUSCAR,CM_BUSCAR));
```

```
  // Habilta la  barra de estado a enviar mensajes asociados a los íconos
  BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
  // Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
            TStatusBar::CapsLock | TStatusBar::NumLock |
            TStatusBar::ScrollLock | TStatusBar::Overtype);
  NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("MENU_PRIN");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                                    : TWindow(Padre, titulo) {
  DatosArchivo = new TOpenSaveDialog::TData(OFN_HIDEREADONLY
          |OFN_FILEMUSTEXIST, "Archivos (*.DAT)|*.dat|", 0, "","DAT");
}

void TVentana::Abrir() {
  if ((TFileOpenDialog(this, *DatosArchivo)).Execute() == IDOK) {
     UnaLista.Recuperar(Arch,DatosArchivo->FileName);
  }
}

void TVentana::Guardar() {
          if ((TFileSaveDialog(this, *DatosArchivo)).Execute() == IDOK)  {
                    UnaLista.Guardar(Arch,DatosArchivo->FileName);
          }
}

void TVentana::Imprimir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Insertar() {
          TDialogoPersona *TDPer = new                                    TDialogoP
          if(TDPer->Execute()==IDOK) {
            if(TPersona.TEsEstudiante==TRUE) {
                    TDialogoEstudiante *TDEst = new                       TDialogoEs
                    TDEst->Execute();
                    int Ed = atoi(TPersona.TEdad);
```

```
            float E1 = atof(TEstudiante.TNota1);
            float E2 = atof(TEstudiante.TNota2);
            if((Ed==0) || (E1==0) || (E2==0))
                  MessageBox("Error en Datos", "Mensaje de
                              Error", MB_OK|MB_ICONHAND);
            else {
                  PEstudiante=new Estudiante();
                  PEstudiante->CambiaNombre(
                                    TPersona.TNombre);
                  PEstudiante->CambiaEdad(Ed);
                  PEstudiante->CambiaNotas(E1,E2);
                  UnaLista.Inserta(PEstudiante);
            }
}
if(TPersona.TEsEstudianteCompu==TRUE)  {
            TDialogoEstCompu *TDEstC = new
                        TDialogoEstCompu(this,"DLG_ESTC");
            TDEstC->Execute();
            int Ed = atoi(TPersona.TEdad);
            float E1 = atof(TEstCompu.TNota1);
            float E2 = atof(TEstCompu.TNota2);
            float E3 = atof(TEstCompu.TNota3);
            if((Ed==0) || (E1==0) || (E2==0) || (E3==0))
                        MessageBox("Error en la edad", "Mensaje
                         de Error", MB_OK|MB_ICONHAND);
            else {
                  PEstCompu=new EstCompu();
                  EstCompu->CambiaNombre(TPersona.TNombre);
                  PEstCompu->CambiaEdad(Ed);
                  PEstCompu->CambiaNotas(E1,E2,E3);
                  UnaLista.Inserta(PEstCompu);
            }
}
if(TPersona.TEsAsistente==TRUE) {
            TDialogoAsistente *TDAsis = new
                        TDialogoAsistente(this,"DLG_ASIS");
            TDAsis->Execute();
            int Ed = atoi(TPersona.TEdad);
            float Prom = atof(TAsistente.TPromedio);
            int HA = atoi(TAsistente.THorasAsis);
            if((Ed==0) || (Prom==0) || (HA==0))
```

```
                        MessageBox("Error en Datos", "Mensaje de
                                        Error", MB_OK|MB_ICONHAND);
                    else {
                        PAsistente=new Asistente();
                        PAsistente->CambiaNombre(TPersona.TNombre);
                        PAsistente->CambiaEdad(Ed);
                        PAsistente->CambiaPromedio(Prom);
                        PAsistente->CambiaHorasAs(HA);
                        PAsistente->CambiaCurso(TAsistente.TCurso);
                        PAsistente->CambiaCarrera(TAsistente.TCarrera);
                        UnaLista.Inserta(PAsistente);
                    }
                }
              }
}

void TVentana::Buscar() {
  if(UnaLista.Cabeza() != NULL) {
        TDialogoDespliega *TDDesp = new
                        TDialogoDespliega(this,"DLG_DESPLIEGA");
        TDDesp->Execute();
  }
  else
        MessageBox("La lista está vacía", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
        CloseWindow();
}

TDialogoPersona::TDialogoPersona(TWindow* Madre, TResId IdenRecur)
                                        : TDialog(Madre, IdenRecur) {
  new TEdit(this, IDC_NOMBRE,30);
  new TEdit(this, IDC_EDAD,4);
  new TRadioButton(this,IDC_ESTUDIANTE);
  new TRadioButton(this,IDC_ESTUDIANTECOMPU);
  new TRadioButton(this,IDC_ASISTENTE);
  strcpy(TPersona.TNombre,"Digite el nombre");
  TPersona.TEsEstudiante=TRUE;
  TPersona.TEsEstudianteCompu=FALSE;
  TPersona.TEsAsistente=FALSE;
```

```
  SetTransferBuffer(&TPersona);
}

TDialogoEstudiante::TDialogoEstudiante(TWindow* Madre, TResId                    IdenRecur)
  new TEdit(this, IDC_NOTA1,4);
  new TEdit(this, IDC_NOTA2,4);
  SetTransferBuffer(&TEstudiante);
}

TDialogoEstCompu::TDialogoEstCompu(TWindow* Madre, TResId                        IdenRecur)
  new TEdit(this, IDC_NOTA1,4);
  new TEdit(this, IDC_NOTA2,4);
  new TEdit(this, IDC_NOTA3,4);
  SetTransferBuffer(&TEstCompu);
}

TDialogoAsistente::TDialogoAsistente(TWindow* Madre, TResId IdenRecur)
                                        : TDialog(Madre, IdenRecur) {
  new TEdit(this, IDC_PROM,4);
  new TEdit(this, IDC_HOR_ASIS,4);
  new TEdit(this, IDC_CARRERA,50);
  new TEdit(this, IDC_CURSO,50);
  SetTransferBuffer(&TAsistente);
}

TDialogoDespliega::TDialogoDespliega(TWindow* Madre, TResId                      IdenRecur)
  new TEdit(this, IDC_NOMBRE,30);
  new TEdit(this, IDC_EDAD,4);
  new TRadioButton(this,IDC_ESTUDIANTE);
  new TRadioButton(this,IDC_ESTUDIANTECOMPU);
  new TRadioButton(this,IDC_ASISTENTE);
  PObjeto = UnaLista.Primero();
  PObjeto->ObtNombre(TPersona.TNombre);
  int Aux=PObjeto->ObtEdad();
  int dec, sign, ndig = 0;
  char *str;
  str = fcvt(Aux, ndig, &dec, &sign);
  strcpy(TPersona.TEdad,str);
  int Tip=PObjeto->ObtTipo();
  if(Tip==1)
          TPersona.TEsEstudiante=TRUE;
```

```
  else
        TPersona.TEsEstudiante=FALSE;
  if(Tip==2)
        TPersona.TEsEstudianteCompu=TRUE;
  else
        TPersona.TEsEstudianteCompu=FALSE;
  if(Tip==3)
        TPersona.TEsAsistente=TRUE;
  else
        TPersona.TEsAsistente=FALSE;
  SetTransferBuffer(&TPersona);
}

void TDialogoDespliega::Siguiente() {
  PObjeto = UnaLista.Siguiente();
  PObjeto->ObtNombre(TPersona.TNombre);
  int Aux=PObjeto->ObtEdad();
  int dec, sign, ndig = 0;
  char *str;
  str = fcvt(Aux, ndig, &dec, &sign);
  strcpy(TPersona.TEdad,str);
  int Tip=PObjeto->ObtTipo();
  if(Tip==1)
        TPersona.TEsEstudiante=TRUE;
  else
        TPersona.TEsEstudiante=FALSE;
  if(Tip==2)
        TPersona.TEsEstudianteCompu=TRUE;
  else
        TPersona.TEsEstudianteCompu=FALSE;
  if(Tip==3)
        TPersona.TEsAsistente=TRUE;
  else
        TPersona.TEsAsistente=FALSE;
  TransferData(tdSetData);
              // Transfiere los datos a la caja, para actualizarla
}

void TDialogoDespliega::Mas() {
  PObjeto = UnaLista.NActual();
  int dec, sign, ndig = 0;
```

```
char *str;
int Tip=PObjeto->ObtTipo();
if(Tip==1) {
        float E1=PObjeto->ObtEx1();
        str = fcvt(E1, ndig, &dec, &sign);
        strcpy(TEstudiante.TNota1,str);
        float E2=PObjeto->ObtEx2();
        str = fcvt(E2, ndig, &dec, &sign);
        strcpy(TEstudiante.TNota2,str);
        TransferData(tdSetData);
        TDialogoEstudiante *TDEst = new
                        TDialogoEstudiante(this,"DLG_ESTU");
        TDEst->Execute();
}
if(Tip==2) {
        TPersona.TEsEstudianteCompu=TRUE;
        float E1=PObjeto->ObtEx1();
        str = fcvt(E1, ndig, &dec, &sign);
        strcpy(TEstCompu.TNota1,str);
        float E2=PObjeto->ObtEx2();
        str = fcvt(E2, ndig, &dec, &sign);
        strcpy(TEstCompu.TNota2,str);
        float E3=PObjeto->ObtEx2();
        str = fcvt(E3, ndig, &dec, &sign);
        strcpy(TEstCompu.TNota3,str);
        TransferData(tdSetData);
        TDialogoEstCompu *TDEstC = new
                        TDialogoEstCompu(this,"DLG_ESTC");
        TDEstC->Execute();
}
if(Tip==3) {
        TPersona.TEsAsistente=TRUE;
        float Prom=PObjeto->ObtPromedio();
        str = fcvt(Prom, ndig, &dec, &sign);
        strcpy(TAsistente.TPromedio,str);
        int HA=PObjeto->ObtHorasAs();
        str = fcvt(HA, ndig, &dec, &sign);
        strcpy(TAsistente.THorasAsis,str);
        PObjeto->ObtCurso(TAsistente.TCurso);
        PObjeto->ObtCarrera(TAsistente.TCarrera);
        TransferData(tdSetData);
```

```
            TDialogoAsistente *TDAsis = new
                                    TDialogoAsistente(this,"DLG_ASIS");
            TDAsis->Execute();
  }
}

// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// OBJ311.HPP

```
#if ! defined(_I_OBJ311_HPP_)
#define _I_OBJ311_HPP_

class Objeto {
public:
          Objeto() { };
          virtual void CambiaNombre(char NuevoNom[30]) { };
          virtual void CambiaEdad(int NuevaEdad) { };
          virtual void ObtNombre(char Nom[30]) { };
          virtual int  ObtEdad(void) {return 0;};
          virtual int  ObtTipo(void) {return 0;};
          virtual int  ObtHorasAs(void) {return 0;};
          virtual void ObtCurso(char Cur[50]) { };
          virtual void ObtCarrera(char Carr[50]) { };
          virtual void CambiaPromedio(float NuevoProm) { };
          virtual float ObtPromedio(void) {return 0;};
          virtual float ObtEx1(void) {return 0;};
          virtual float ObtEx2(void) {return 0;};
          virtual void Muestra(void) { };
          virtual void Captura(void) { };
          virtual void Guardar(FILE *archivo) { };
          virtual void Recuperar(FILE *archivo){ };
};
#endif
```

// PERS311.HPP

```
// protege para múltiple inclusión
#if ! defined(_I_PERS11_HPP_)
```

```
#define _I_PERS311_HPP_

#include <stdio.h>
#include "obj311.hpp"
#include "lpers311.hpp"

class Persona : public Objeto {
protected:                                  // Permite que estos atributos sean
        char Nombre[30];          // conocidos en las clases derivadas
        int  Edad;
public:
        Persona(char Nom[30], int Ed);
        Persona();
        virtual void CambiaNombre(char NuevoNom[30]);
        virtual void CambiaEdad(int NuevaEdad);
        virtual void ObtNombre(char Nom[30]);
        virtual int  ObtEdad(void);
        virtual int  ObtTipo(void);
        // Retorna 0=Persona, 1=Estudiante, 2=EstuCompu, 3=Asistente
        virtual void Guardar(FILE *archivo);
        virtual void Recuperar(FILE *archivo);
};

class Estudiante : public Persona {
protected:
        float Examen1;
        float Examen2;
        float Promedio;
public:
        Estudiante(char Nom[30], int Ed, float Ex1, float Ex2);
        Estudiante();
        virtual void  CalculaPromedio(void);
        void  CambiaNotas(float NuevaNota1, float NuevaNota2);
        virtual float ObtEx1(void);
        virtual float ObtEx2(void);
        virtual float ObtProm(void);
        virtual int  ObtTipo(void);
        virtual void Guardar(FILE *archivo);
        virtual void Recuperar(FILE *archivo);
};
```

```
class EstCompu : public Estudiante {
        float  Examen3;
public:
        EstCompu(char Nom[30],int Ed, float Ex1, float Ex2, float Ex3);
        EstCompu();
        void CambiaNotas(float NuevaNota1, float NuevaNota2, float Nueva-
Nota3);
        virtual void CalculaPromedio(void);
        virtual float ObtEx3(void);
        virtual int  ObtTipo(void);
        virtual void Guardar(FILE *archivo);
        virtual void Recuperar(FILE *archivo);
};


class Asistente : public Persona {
        float Promedio;
        int HorasAs;
        char Curso[50];
        char Carrera[50];
public:
        Asistente(char Nom[30],int Ed,float Prom,char Cur[50],int HA,char
Carr[50]);
        Asistente();
        virtual int  ObtHorasAs(void);
        virtual void ObtCurso(char Cur[50]);
        virtual void ObtCarrera(char Carr[50]);
        virtual void CambiaPromedio(float NuevoProm);
        virtual float ObtPromedio(void);
        virtual void CambiaHorasAs(int NuevaHo);
        virtual void CambiaCurso(char NuevoCurso[50]);
        virtual void CambiaCarrera(char NuevaCarrera[50]);
        virtual int  ObtTipo(void);
        virtual void Guardar(FILE *archivo);
        virtual void Recuperar(FILE *archivo);
};
#endif
```

// PERS311.CPP

```
#include "pers311.hpp"
#include <string.h>
```

```
#include <stdio.h>

/*-------------------------P E R S O N A-------------------------------*/
Persona::Persona(char Nom[30],int Ed) {
  strcpy(Nombre,Nom);
  Edad = Ed;
}

Persona::Persona() {
  strcpy(Nombre,"");
  Edad = 0;
}

void Persona::CambiaNombre(char NuevoNom[30]) {
  strcpy(Nombre,NuevoNom);
}

void Persona::CambiaEdad(int NuevaEdad) {
  Edad = NuevaEdad;
}

void Persona::ObtNombre(char Nom[30]) {
  strcpy(Nom,Nombre);
}

int Persona::ObtEdad(void) {
  return Edad;
}

int Persona::ObtTipo(void) {
  return 0;
}

void Persona::Guardar(FILE *archivo) {
  char aux='P';
  fwrite(&aux,sizeof(char),1,archivo);
  fwrite(&Edad,sizeof(int),1,archivo);
  fwrite(Nombre,30*sizeof(char),1,archivo);
}

void Persona::Recuperar(FILE *archivo) {
```

```
  fread(&Edad,sizeof(int),1,archivo);
  fread(Nombre,30*sizeof(char),1,archivo);
}

/*------------------------E S T U D I A N T E----------------------------*/
Estudiante::Estudiante(char Nom[30],int Ed,float Ex1,float Ex2) : Perso-
na(Nom,Ed) {
  Examen1 = Ex1;
  Examen2 = Ex2;
}

Estudiante::Estudiante() : Persona() {
  Examen1 = 0;
  Examen2 = 0;
}

void Estudiante::CalculaPromedio(void) {
  Promedio = (Examen1 + Examen2)/2;
}

void Estudiante::CambiaNotas(float NuevaNota1, float NuevaNota2) {
  Examen1 = NuevaNota1;
  Examen2 = NuevaNota2;
}

float Estudiante::ObtEx1(void) {
  return Examen1;
}

float Estudiante::ObtEx2(void) {
  return Examen1;
}

float Estudiante::ObtProm(void) {
  return Promedio;
}

int Estudiante::ObtTipo(void) {
  return 1;
}
```

```
void Estudiante::Guardar(FILE *archivo) {
  char aux='E';
  fwrite(&aux,sizeof(char),1,archivo);
  fwrite(&Edad,sizeof(int),1,archivo);
  fwrite(Nombre,30*sizeof(char),1,archivo);
  fwrite(&Examen1,sizeof(float),1,archivo);
  fwrite(&Examen2,sizeof(float),1,archivo);
  fwrite(&Promedio,sizeof(float),1,archivo);
}

void Estudiante::Recuperar(FILE *archivo) {
  Persona::Recuperar(archivo);
  fread(&Examen1,sizeof(float),1,archivo);
  fread(&Examen2,sizeof(float),1,archivo);
  fread(&Promedio,sizeof(float),1,archivo);
}

/*------------------------E S T - C O M P----------------------------*/
EstCompu::EstCompu(char Nom[30],int Ed,float Ex1,float Ex2,float Ex3) : Es-
tudiante(Nom,Ed,Ex1,Ex2) {
        Examen3 = Ex3;
}

EstCompu::EstCompu() : Estudiante() {
        Examen3 = 0;
}

void EstCompu::CambiaNotas(float NuevaNota1, float NuevaNota2, float Nue-
vaNota3) {
  Examen1 = NuevaNota1;
  Examen2 = NuevaNota2;
  Examen3 = NuevaNota3;
}

void EstCompu::CalculaPromedio(void) {
  Promedio = (Examen1 + Examen2 + Examen3)/3;
}

float EstCompu::ObtEx3(void) {
  return Examen3;
}
```

```
int EstCompu::ObtTipo(void) {
  return 2;
}

void EstCompu::Guardar(FILE *archivo) {
  char aux='C';
  fwrite(&aux,sizeof(char),1,archivo);
  fwrite(&Edad,sizeof(int),1,archivo);
  fwrite(Nombre,30*sizeof(char),1,archivo);
  fwrite(&Examen1,sizeof(float),1,archivo);
  fwrite(&Examen2,sizeof(float),1,archivo);
  fwrite(&Examen3,sizeof(float),1,archivo);
  fwrite(&Promedio,sizeof(float),1,archivo);
}

void EstCompu::Recuperar(FILE *archivo) {
  Persona::Recuperar(archivo);
  fread(&Examen1,sizeof(float),1,archivo);
  fread(&Examen2,sizeof(float),1,archivo);
  fread(&Examen3,sizeof(float),1,archivo);
  fread(&Promedio,sizeof(float),1,archivo);
}

/*------------------------A S I S T E N T E-----------------------------*/
Asistente::Asistente(char Nom[30],int Ed,float Prom,char Cur[50],
                              int HA,char Carr[50]) : Persona(Nom,Ed){
        Promedio = Prom;
        strcpy(Curso,Cur);
        HorasAs = HA;
        strcpy(Carrera,Carr);
}

Asistente::Asistente(){
        Promedio = 0;
        strcpy(Curso,"");
        HorasAs = 0;
        strcpy(Carrera,"");
}

float Asistente::ObtPromedio(void) {
```

```
  return Promedio;
}

int Asistente::ObtHorasAs(void) {
  return HorasAs;
}

void Asistente::ObtCurso(char Cur[50]) {
  strcpy(Cur,Curso);
}

void Asistente::ObtCarrera(char Carr[50]) {
  strcpy(Carr,Carrera);
}

void Asistente::CambiaPromedio(float NuevoProm) {
  Promedio = NuevoProm;
}

void Asistente::CambiaHorasAs(int NuevaHo) {
  HorasAs = NuevaHo;
}

void Asistente::CambiaCarrera(char NuevaCarrera[50]) {
  strcpy(Carrera,NuevaCarrera);
}

void Asistente::CambiaCurso(char NuevoCurso[50]) {
  strcpy(Curso,NuevoCurso);
}

int Asistente::ObtTipo(void) {
  return 3;
}

void Asistente::Guardar(FILE *archivo) {
  char aux='A';
  fwrite(&aux,sizeof(char),1,archivo);
  fwrite(&Edad,sizeof(int),1,archivo);
  fwrite(Nombre,30*sizeof(char),1,archivo);
  fwrite(&Promedio,sizeof(float),1,archivo);
```

```
  fwrite(&HorasAs,sizeof(int),1,archivo);
  fwrite(Curso,50*sizeof(char),1,archivo);
  fwrite(Carrera,50*sizeof(char),1,archivo);
}

void Asistente::Recuperar(FILE *archivo) {
  Persona::Recuperar(archivo);
  fread(&Promedio,sizeof(float),1,archivo);
  fread(&HorasAs,sizeof(int),1,archivo);
  fread(Curso,50*sizeof(char),1,archivo);
  fread(Carrera,50*sizeof(char),1,archivo);
}
```

// LPERS311.HPP

```
// protege para múltiple inclusión
#if ! defined(_I_LPERS311_HPP_)
#define _I_LPERS311_HPP_
#include "obj311.hpp"

struct Nodo {
        Objeto *PObjeto;
        Nodo   *Sig;
};

class ListaObjetos {
  Nodo *Primera;
  Nodo *Actual;
public:
  ListaObjetos();
  ~ListaObjetos();
  Nodo *Cabeza() {
                return Primera;
  };
  Objeto *Primero() {
                return Primera->PObjeto;
  };
  void Inserta(Objeto *NuevoObjeto);
  Objeto *Siguiente(void);
  Objeto *NActual(void);
  void Guardar(FILE *archivo,char NomArch[80]);
  void Recuperar(FILE *archivo,char NomArch[80]);
```

```
};
#endif

// LPERS311.CPP
#include <stdio.h>
#include "lpers311.hpp"
#include "pers311.hpp"

ListaObjetos::ListaObjetos() {
        Primera=NULL;
        Actual=Primera;
}

ListaObjetos::~ListaObjetos() {
  while(Primera != NULL) {
    Nodo *Tempo=Primera;
                Primera=Primera->Sig;
    delete Tempo->PObjeto;
                delete Tempo;
  }
}

void ListaObjetos::Inserta(Objeto *NuevaPersona) {
        Nodo *NodoTemp, *NuevoNodo;
        NuevoNodo = new Nodo;
        NuevoNodo->PObjeto = NuevaPersona;
        NuevoNodo->Sig = NULL;
        if(Primera==NULL) {
                Primera = NuevoNodo;
                Actual = Primera;                       // LINEA NUEVA
        }
        else {
                  NodoTemp=Primera;
                  while(NodoTemp->Sig != NULL)
                          NodoTemp=NodoTemp->Sig;
                  NodoTemp->Sig=NuevoNodo;
        }
}

Objeto *ListaObjetos::Siguiente() {
  if((Actual->Sig) != NULL)
```

```
            Actual = Actual->Sig;
  else
            Actual = Primera;
  return (Actual->PObjeto);
}

Objeto *ListaObjetos::NActual() {
  return (Actual->PObjeto);
}

void ListaObjetos::Guardar(FILE *archivo,char NomArch[80])  {
  Nodo *Actual=Primera;
  archivo = fopen(NomArch,"w");
  while(Actual != NULL) {
                    Actual->PObjeto->Guardar(archivo);
                    Actual = Actual->Sig;
  }
  fclose(archivo);
}

void ListaObjetos::Recuperar(FILE *archivo,char NomArch[80])  {
  char clave;
  ListaObjetos::~ListaObjetos();
  archivo = fopen(NomArch,"r");
  while(!feof(archivo)) {
            if(fread(&clave,sizeof(char),1,archivo)!=0) {
                    switch(clave) {
                      case 'P': { Persona *PObjeto=new Persona();
                                  PObjeto->Recuperar(archivo);
                                  Inserta(PObjeto); }
                                  break;
                      case 'E': { Estudiante *PEstudiante=new Estudiante();
                                  PEstudiante->Recuperar(archivo);
                                  Inserta(PEstudiante); }
                                  break;
                      case 'C': { EstCompu *PEstCompu=new EstCompu();
                                  PEstCompu->Recuperar(archivo);
                                  Inserta(PEstCompu); }
                                  break;
                      case 'A': { Asistente *PAsistente=new Asistente();
                                  PAsistente->Recuperar(archivo);
```

```
                                        Inserta(PAsistente); }
                                        break;
                    }
            }
  }
  fclose(archivo);
}
```

// E3_11.RC

```
#include "e3_11.h"
MENU_PRIN MENU
{
 POPUP "&Archivo"
 {
  MENUITEM "&Abrir...", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM SEPARATOR
  MENUITEM "&Imprimir...", CM_IMPRIMIR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
 POPUP "&Mantenimiento"
 {
  MENUITEM "&Insertar...", CM_INSERTAR
  MENUITEM "&Buscar...", CM_BUSCAR
 }
}

IDB_SALIR BITMAP   {
'42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
'00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
'00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
'00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
'00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
'00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
'00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
'00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
'00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
'87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
'46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
'00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
```

```
'40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
'46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
'00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
'40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
'46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
'00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
'78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
'46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
'00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
'88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
'88 88 88 88 00 00'

}

IDB_ABRIR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP {
```

```
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 00 08 23 10 88 0F'
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE {
  CM_SALIR, "Selecciona salir del programa"
  CM_ABRIR, "Abrir un archivo de personas"
  CM_GUARDAR, "Guardar el archivo personas"
  CM_IMPRIMIR, "Imprime el archivo de personas"
  CM_INSERTAR, "Inserta una persona a la lista"
  CM_BUSCAR, "Busca una persona de la lsita"
}

IDB_IMPRIMIR BITMAP   {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
```

```
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 70 00 00 00 00 00 00 07 88 65 F0 88 06'
 '76 76 76 76 76 76 70 88 65 F0 80 00 00 00 00 00'
 '00 00 00 08 65 F0 80 66 66 66 66 66 66 66 66 08'
 '65 F0 80 66 06 06 06 66 66 00 66 08 65 F0 80 66'
 '66 66 66 66 66 66 66 08 A2 C0 80 88 F8 F8 F8 F8'
 'F8 F8 F8 08 00 00 88 00 00 00 00 00 00 00 00 88'
 '88 88 88 80 60 FF FF FF FF 06 08 88 04 6D 88 80'
 '00 F0 07 07 0F 00 08 88 88 80 88 88 80 FF FF FF'
 'FF 08 88 88 01 09 88 88 80 F0 70 88 88 08 88 88'
 '00 88 88 88 80 FF FF F0 00 08 88 88 00 E0 88 88'
 '80 F0 70 F0 F0 88 88 88 2A 20 88 88 80 FF FF F0'
 '08 88 88 88 22 A4 88 88 80 00 00 00 88 88 88 88'
 '04 08 88 88 88 88 88 88 88 88 88 88 99 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_INSERTAR BITMAP  {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 4F 8F 8F 8F 8F 8F'
 '00 00 88 88 88 88 48 88 88 88 78 78 00 00 88 88'
 '77 77 77 77 77 78 8F 8F 00 00 88 84 44 44 44 44'
 '44 78 78 78 00 00 88 84 FB FB FB FB F4 78 8F 8F'
 '00 00 88 84 BF BF BF BF B4 78 78 78 00 00 88 84'
 'FB FB FB FB F4 78 8F 8F 00 00 88 84 BF BF BF BF'
 'B4 78 78 78 00 00 88 84 FB FB FB FB F4 78 8F 8F'
 '00 00 88 84 44 44 44 44 44 88 78 78 00 00 88 88'
 '88 88 4F 8F 8F 8F 8F 8F 00 00 88 80 88 88 48 F8'
 'F8 78 78 78 00 00 88 80 08 88 4F 8F 8F 8F 8F 8F'
 '67 14 00 00 00 88 48 F8 F8 F8 F8 F8 8A 44 00 00'
 '00 88 4F 8F 8F 8F 8F 8F CD EB 88 80 08 88 48 F8'
 'F8 F8 F8 F8 41 98 88 80 88 88 44 44 44 44 44 44'
 'E9 F3 88 88 88 88 88 88 88 88 88 88 93 30 88 88'
```

```
 '88 88 88 88 88 88 88 88 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_BUSCAR BITMAP {
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 FB FB FF FF FB'
 '44 4C 88 88 88 88 8F BF BF FF BF FF 23 10 88 88'
 '88 88 88 FF FF BF FF FF 88 C8 88 88 88 88 8F BF'
 'BF FF BF FB 00 31 88 88 88 88 88 FB FB FF FF FF'
 '10 25 88 88 88 88 84 44 7F BF FF FB 99 55 88 88'
 '88 88 44 73 44 FB FB FF 10 25 88 88 88 84 47 8F'
 '34 7F BF BB 9D 44 88 88 88 44 74 8F F7 4B FF FF'
 '22 26 88 88 88 47 47 4F F7 4F BF BF 00 01 88 88'
 '88 44 7F 84 44 78 F8 F8 19 A2 88 88 84 47 FF F8'
 '74 88 88 88 20 02 88 88 44 77 48 87 48 88 88 88'
 '44 4C 88 84 47 F8 74 44 88 88 88 88 01 03 88 44'
 '7F 88 48 88 88 88 88 88 22 44 84 47 F8 84 88 88'
 '88 88 88 88 41 26 84 7F 88 48 88 88 88 88 88 88'
 '21 91 84 78 84 88 88 88 88 88 88 88 02 54 88 44'
 '48 88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

DLG_PERS DIALOG 80, 39, 188, 193
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CLASS "bordlg"
CAPTION "PERSONA"
FONT 8, "MS Sans Serif"
{
 EDITTEXT IDC_NOMBRE, 29, 20, 120, 12
 EDITTEXT IDC_EDAD, 29, 55, 19, 12
```

```
 CONTROL "Estudiante", IDC_ESTUDIANTE, "BorRadio",
BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 43,
91, 46, 10
 CONTROL "Estudiante de Computación", IDC_ESTUDIANTECOMPU, "Bo-
rRadio", BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP, 43, 109, 101, 10
 CONTROL "Asistente", IDC_ASISTENTE, "BorRadio",
BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 43,
128, 41, 10
 CONTROL "OK", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 46, 162, 37, 25
 CONTROL "Cancelar", IDCANCEL, "BorBtn", BS_PUSHBUTTON |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 114, 163, 38, 25
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD |
WS_VISIBLE, -1, 154, 189, 3
 LTEXT "Nombre", -1, 29, 8, 26, 8
 LTEXT "Edad", -1, 29, 41, 18, 8
 CONTROL "", -1, "BorShade", BSS_GROUP | BSS_CAPTION | BSS_LEFT |
WS_CHILD | WS_VISIBLE | WS_GROUP, 29, 83, 130, 63
 LTEXT "Tipo de persona", -1, 29, 72, 127, 8, NOT WS_GROUP
}

DLG_ESTU DIALOG 90, 73, 189, 124
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CLASS "bordlg"
CAPTION "Datos de Estudiante"
FONT 8, "MS Sans Serif"
{
 EDITTEXT IDC_NOTA1, 84, 25, 20, 12
 EDITTEXT IDC_NOTA2, 84, 62, 21, 12
 CONTROL "", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 48, 92, 37, 25
 CONTROL "", IDCANCEL, "BorBtn", BS_PUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 104, 92, 37, 25
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD |
WS_VISIBLE, 0, 83, 189, 3
 LTEXT "Examen 1", -1, 78, 10, 33, 8
 LTEXT "Examen 2", -1, 78, 45, 33, 8
}
```

```
DLG_ESTC DIALOG 90, 50, 189, 147
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CLASS "bordlg"
CAPTION "Datos Estudiante de Computación"
FONT 8, "MS Sans Serif"
{
 EDITTEXT IDC_NOTA1, 86, 23, 20, 12
 EDITTEXT IDC_NOTA2, 86, 57, 21, 12
 EDITTEXT IDC_NOTA3, 86, 91, 22, 12
 CONTROL "", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 48, 115, 37, 25
 CONTROL "", IDCANCEL, "BorBtn", BS_PUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 104, 115, 37, 25
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD |
WS_VISIBLE, 0, 106, 189, 3
 LTEXT "Examen 1", -1, 79, 8, 33, 8
 LTEXT "Examen 2", -1, 79, 41, 33, 8
 LTEXT "Examen 3", -1, 79, 77, 33, 8
}

DLG_ASIS DIALOG 90, 73, 189, 124
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CLASS "bordlg"
CAPTION "Datos Asistente"
FONT 8, "MS Sans Serif"
{
 EDITTEXT IDC_PROM, 9, 23, 20, 12
 EDITTEXT IDC_HOR_ASIS, 9, 59, 21, 12
 EDITTEXT IDC_CARRERA, 87, 23, 95, 12
 EDITTEXT IDC_CURSO, 87, 59, 95, 12, ES_OEMCONVERT | WS_BORDER |
WS_TABSTOP
 CONTROL "", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP, 48, 92, 37, 25
 CONTROL "", IDCANCEL, "BorBtn", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP, 104, 92, 37, 25
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE,
0, 83, 189, 3
 LTEXT "Promedio Ponderado", -1, 9, 10, 69, 8
 LTEXT "Horas Asistente", -1, 9, 43, 53, 8
 LTEXT "Carrera", -1, 87, 10, 24, 8
 LTEXT "Curso Asignado", -1, 87, 43, 53, 8
}
```

```
DLG_DESPLIEGA DIALOG 80, 39, 217, 193
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CLASS "bordlg"
CAPTION "Despliega Lista"
FONT 8, "MS Sans Serif" {
 EDITTEXT IDC_NOMBRE, 43, 20, 120, 12
 EDITTEXT IDC_EDAD, 43, 55, 19, 12
 CONTROL "Estudiante", IDC_ESTUDIANTE, "BorRadio", BS_AUTORADIOBUTTON |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 53, 91, 46, 10
 CONTROL "Asistente", IDC_ASISTENTE, "BorRadio", BS_AUTORADIOBUTTON |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 53, 128, 41, 10
 CONTROL "OK", IDOK, "BorBtn", BS_DEFPUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 15, 162, 37, 25
 CONTROL "Cancelar", IDCANCEL, "BorBtn", BS_PUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 69, 162, 38, 25
 CONTROL "Más", IDC_MAS, "BorBtn", BS_PUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 117, 162, 37, 25
 CONTROL "Siguiente ->", IDC_SIGUIENTE, "BorBtn", BS_PUSHBUTTON |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 165, 162, 37, 25
 CONTROL "Estudiante de Computación", IDC_ESTUDIANTECOMPU, "BorRadio",
BS_AUTORADIOBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 53, 109,
101, 10
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE, -
1, 156, 218, 1
 LTEXT "Nombre", -1, 43, 8, 26, 8
 LTEXT "Edad", -1, 43, 41, 18, 8
 CONTROL "", -1, "BorShade", BSS_GROUP | BSS_CAPTION | BSS_LEFT |
WS_CHILD | WS_VISIBLE | WS_GROUP, 43, 83, 130, 63
 LTEXT "Tipo de persona", -1, 43, 72, 127, 8, NOT WS_GROUP
}
```

3.8 El método Paint de TWindow

En Windows todos los elementos visuales, aún los textos, son desplegados en pantalla en forma gráfica. Para esto Windows provee dispositivos que permiten crear una interfaz gráfica (Graphics Device Interface GDI), mediante los cuales se pueden crear efectos visuales bastante espectaculares.

El objetivo de esta sección es mostrar cómo el método de Paint de la clase TWindow de OWL permite desplegar gráficos por pantalla al res-

ponder al mensaje WM_PAINT del lenguaje Windows Estándar. Además se presentarán algunas de las clases más importantes en OWL para generar gráficos, como son TRect, TPoint, entre otras.

Cuando en Windows una ventana está sobre otra y es posteriormente cerrada, el mensaje WM_PAINT tiene como propósito indicarle a Windows que debe reconstruir la *región inválida* (esto es la parte en donde las ventanas estaban sobrepuestas). Para el proceso de redibujar la pantalla o porciones de esta, Windows utiliza un procedimiento bastante eficiente conocido como "*almacene ahora y dibuje después*", el cual consiste en guardar los parámetros de la pantalla (o porción de esta) como son coordenadas de las líneas, valor del color, diámetros del círculos, patrones, entre otros, en lugar de almacenar el Bitmap como tal.

Como ya hemos mencionado el método Paint de la clase TWindow es encargado de redibujar la pantalla cuando esta es invalidada, por ejemplo cuando un botón elevador es presionado. Este método es heredado por todas las clases derivadas de TWindow, como son toolbars, statusline, cajas de diálogo y muchas otras. Como se ilustra en el ejemplo 3.12 cualquier gráfico que deba desplegarse por pantalla debe codificarse como parte del método Paint.

En el ejemplo 3.12 se presenta un programa bastante simple que permite ilustrar

la forma en que se debe programar el método Paint, en este caso en la clase TVentana que hereda de la clase TFrameWindow, que a su vez hereda de la clase TWindow. La salida de este programa se presenta en la Figura 3.20.

🖫 Ejemplo 3.12. Implementación del método Paint

```
// E3_12.CPP
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\dc.h>
#include "e3_12.h"
```

```
class TVentana: public TFrameWindow {
public:
  TVentana(TWindow* parent, const char far* title);
  virtual void Paint(TDC& dc, BOOL erase, TRect& rect);
};
```

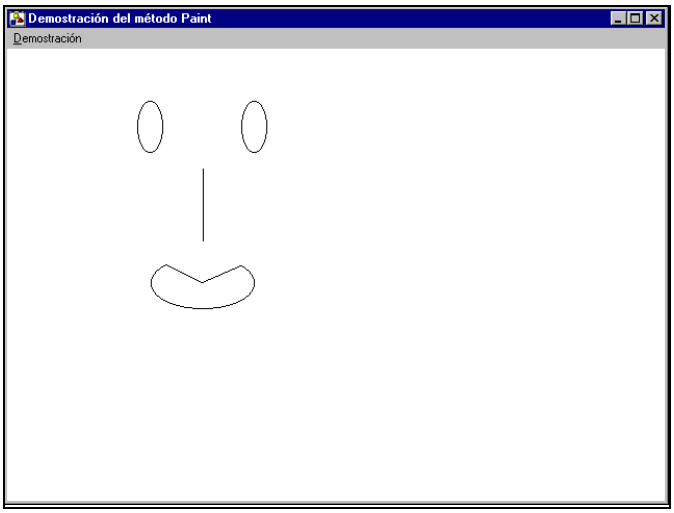


FIGURA 3.20 Salida del método Paint.

```
class TPrincipal: public TApplication {
public:
  TPrincipal() : TApplication() {};
  void InitMainWindow();
};

TVentana::TVentana(TWindow* parent, const char far* title)
  : TFrameWindow(parent, title), TWindow(parent, title) {

// Permite que la ventana se inicie máximizada
  Attr.X = 0;
  Attr.Y = 0;
  Attr.H = GetSystemMetrics(SM_CYSCREEN);
  Attr.W = GetSystemMetrics(SM_CXSCREEN);
  AssignMenu(E312_MENU);
}
```

```
void TVentana::Paint(TDC& dc, BOOL /*erase*/, TRect& /*rect*/) {
  dc.Ellipse(125, 50, 150, 100);
  dc.Ellipse(225, 50, 250, 100);
  dc.MoveTo(188,115);
  dc.LineTo(188,185);
  dc.Pie(138,200,238,250,138,200,238,200);
}

void TPrincipal::InitMainWindow() {
  MainWindow = new TVentana(0, "Demostración del método Paint");
}

int OwlMain(int /* argc */ , char* /* argv */ []) {
  TPrincipal Ap;
  return Ap.Run();
}

// E3_12.H
#define E312_MENU 100
#include <owl\window.rh>
#include "e3_12.h"

E312_MENU MENU
BEGIN
        POPUP "&Demostración"
        BEGIN
                MENUITEM "&Salir", CM_EXIT
        END
END
```

Nótese que en el listado de E3_12.CPP aparece #include <owl\dc.h> en el cual se declaran las clases para el dispositivo de contexto que encapsula las funciones y los datos del GDI de Windows. Antes de crear cualquier gráfico en pantalla se debe crear u obtener una clase para el dispositivo de contexto, la cual usualmente se refiere a un "DC". Piense por ahora que un DC es un dispositivo de interfaz al cual Windows enviará el gráfico, por ejemplo la pantalla, la impresora o un ploter.

En programación tradicional de Windows (sin OWL) un DC es una "estructura-clase" (struct de C estándar - recuerde que una estructura en

C++ también es una clase) tipo HDC. Object Windows Library (OWL) provee una clase denominada TDC la cual hereda de la estructura-clase DC del Windows estándar, esto con el fin de encapsular los atributos de la estructura DC y proveer a esta de las características de la programación orientada a objetos. Entonces, como es de esperar, en el método Paint de la clase TVentana se recibe un parámetro (una referencia) de tipo TDC, en el cual viene el dispositivo de contexto. Esto se ejemplifica en el siguiente fragmento de código: (la clase TDC de OWL se explica con mayor detalle adelante)

```
void TVentana::Paint(TDC& dc, BOOL /*erase*/, TRect& /*rect*/) {
  dc.Ellipse(125, 50, 150, 100);
  dc.Ellipse(225, 50, 250, 100);
  dc.MoveTo(188,115);
  dc.LineTo(188,185);
  dc.Pie(138,200,238,250,138,200,238,200);
}
```

El método Paint recibe tres parámetros, una referencia "dc" una clase de tipo TDC, una variable erase de tipo BOOL (que toma valores true o false el cual es un tipo del C estándar de Windows), y una referencia rect a una clase TRect (La clase TRect pertenece a OWL). Este método está redifiniendo al método Paint de la clase TWindow (por esto a los parámetros se les dan nombres en inglés) y se debe declarar virtual. El parámetro dc indica el dispositivo en donde el gráfico será enviado. El valor de erase es true si el mensaje WM_PAINT solicita que la pantalla sea redibujada. El parámetro rect indica la región de la pantalla que será redibujada.

En el ejemplo 3.12 el método Paint dibuja un rectángulo y una elipse, para esto utiliza la función Rectangle(100, 100, 250, 275) y la función Ellipse(125, 125, 260, 285) respectivamente. Estos métodos son parte de la clase TDC que explicaremos con detalle en los siguientes párrafos.

Clases de OWL para gráficos

OWL provee un conjunto de clases con el objetivo de darle las características de programación orientada a objetos a una serie de estructuras (struct) tipo C estándar de Windows, que son usadas para presentar gráficos en pantalla o impresoras. Si el programador lo dessea, puede usar en Borland C++ las estructuras y funciones del C-Windows directamente. Sin embargo esto no es recomendable, pues las clases de OWL tienen una serie de ventajas adicionales a las estructuras del C estándar de Windows.

Las cuatro principales categorías de clases gráficas son:

1. Clases de localización y tamaño.
2. Clases para colores.
3. Clases para objetos gráficos.
4. Clases para dispositivos de contexto.

Veamos en detalle cada uno de estos grupos de clases.

Clases de localización y tamaño

Las clases de localización y tamaño no tienen efectos visibles, éstas se usan meramente como parámetros de otros métodos de otras clases. Estas clases están definidas en el archivo POINT.H, y se presentan en la Figura 3.21.

La estructura-clase tagPOINT pertenece al C de Windows y su definición es la siguiente:

```
typedef struct tagPOINT {
        int x;
        int y;
} POINT;
```

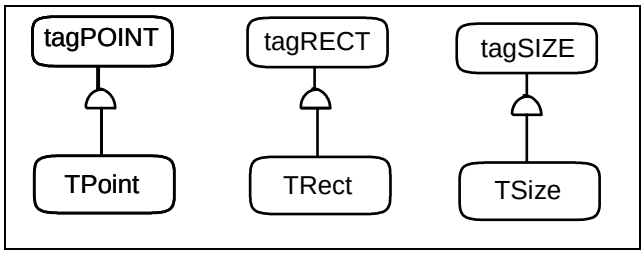


FIGURA 3.21 Jerarquía de clases de tamaño y loca- lización.

objetos a la clase

Claramente cualquier variable de tipo TPoint de OWL será compatible con un objeto tipo POINT debido a la relación de herencia que existe entre éstas.

Estas clases no producen ningún gráfico, por ejemplo, la clase TRect no grafica ningún rectángulo, lo único que hace es almacenar las coordenadas de una región rectangular y redefinir la clase tagRECT para proveerla de las características de la orientación a objetos. Aprovechándose de tales características, en un programa o método se podrían tener instrucciones como las siguientes:

```
TPoint p1(10,20);
TPoint p2(p1);
TPoint p3;
p3 = p2;
if(p1 == p3) {
        . . .
}
p1 = p2+p3;
. . .
```

La clase TRect derivada de la estructura tagRECT de Windows tiene como propósito almacenar las coordenadas de una región rectangular de la pantalla y también agregarle las características de la orientación a

```
TRect   r1(10,20,200,300);
TRect   r2(r1);
TRect(p1,p2);
TRect(p1+p2,p3);
```

Clases para color

Windows ha venido a representar un mundo de colores para las computadoras personales PC. Para el manejo de colores OWL tiene una clase denominada TColor que viene a sustituir a la estructura-clase de Windows denominada COLORREF. Para declarar una variable de tipo TColor se hace como en el siguiente fragmento de código, en donde se definen los colores Rojo, Verde y Azul:

```
TColor  Rojo(255,0,0);
TColor  Verde(0,255,0);
TColor  Azul(0,0,255);
```

Como puede verse, debido a que el constructor de TColor recibe tres parámetros cuyos posibles valores varían de 0 a 255 se pueden crear 256^3 colores diferentes. Por ejemplo se puede tener el color:

```
TColor  Celeste(58,140,245);
```

TColor también provee algunos colores estándar, éstos son:

```
static const TColor Black;
static const TColor LtGray;
static const TColor Gray;
static const TColor LtRed;
static const TColor LtGreen;
static const TColor LtYellow;
static const TColor LtBlue;
static const TColor LtMagenta;
static const TColor LtCyan;
static const TColor LtWhite;
```

Para utilizar un color de estos no olvide usar prefijo TColor::, por ejemplo:

```
TColor  Blanco = TColor::LtWhite;
```

Clases para crear objetos gráficos

En OWL todos los objetos gráficos son instancias de clases derivadas de TGdiObject, declaradas en el archivo GDIOBJECT.H. Debido a que todos estos objetos derivan de la misma clase su uso es muy similar. Esta jerarquía de clases se presenta parcialmente en la Figura 3.22.

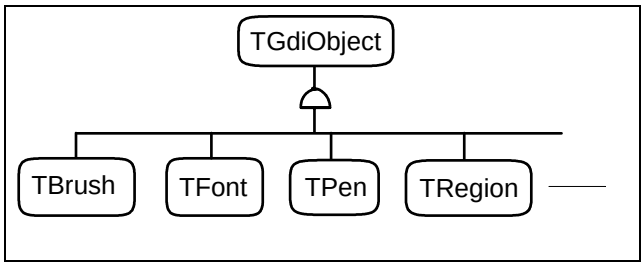


FIGURA 3.22 Jerarquía de clases de objetos gráficos.

TGdiObject es una clase abstracta en OWL, cuyo único fin es agrupar las clases de gráficos. Además de este modo, como se estudió en el capítulo 2, un puntero de tipo TGdiObject podrá apuntar a cualquiera de sus clases derivadas, permitiéndose el polimorfismo entre las clases gráficas.

La clase TPen permite seleccionar el color, el ancho y el tipo de la "pluma" con que se va a dibujar, en un programa se podría tener:

```
TColor Rojo(255,0,0);
TPen Pluma(Rojo);
TPen Pluma(Rojo,4);
TPen Pluma(Rojo,4,PS_SOLID);
```

En el ejemplo anterior se invocaron tres constructores diferentes de TPen, el primer parámetro es el color, el segundo es el ancho y el tercero es el tipo.

La clase TBrush funciona de manera similar. Permite seleccionar la "brocha" con la que se dibujará, por ejemplo:

```
TColor Verde(0,255,0);
TBrush Relleno(Verde,HS_CROSS);
TBrush Relleno(TColor::Red);
```

En el ejemplo anterior se invocaron dos constructores diferentes de TBrush. El primer parámetro del constructor es el color y el segundo es el tipo de trazo, en este caso HS_CROSS es un trazo cuadriculado (los demás tipos de trazos pueden consultarse en la ayuda de Borland C++ presionando la tecla F1 sobre la palabra HS_CROSS).

El tipo TFont es un poco más complicado debido a que el constructor recibe más parámetros, por ejemplo:

```
TFont* font;
font = new TFont(
        "Courier New",                  // Nombre del font en Windows
        -::MulDiv(36, 96, 120),         // forma del font
        0, 0, 0,                        // ancho, escape, orientación
        FW_NORMAL,          // peso
        FIXED_PITCH,                    // pitchy
        FALSE, FALSE, FALSE,   // itálica, subrayada, doblesubrayada
        ANSI_CHARSET,                   // charSet
        OUT_TT_PRECIS,                  // outputPrecision
        CLIP_DEFAULT_PRECIS,            // clipPrecision
        PROOF_QUALITY                   // calidad
);
```

Clases para dispositivos de contexto

La última categoría de clases para los GDI en OWL son las clases derivadas de TDC, conocidas como "*Device Context Classes*". Entre ellas tenemos TWindowDC, TScreenDC, TPrintDC, etc. No obstante la más importante de todas es la clase misma TDC, pues en esta se encapsulan todas las funciones del "*Graphics Device Interface*" (GDI) de Windows estándar, por ejemplo, las funciones encargadas de dibujar rectángulos y elipses, entre otras.

Para invocar adecuadamente una función de la clase TDC se deben seguir los siguientes cuatro pasos:

1. Obtener o construir un dispositivo de contexto usando una instancia de clase TDC. En el método Paint de TWindow; esta instancia siempre viene como parámetro.
2. Construir los objetos gráficos necesarios, por ejemplo, TPen, TColor o TBrush.
3. Llamar una o más funciones de la clase TDC, o de algunas de sus clases derivadas.
4. Restaurar la pluma (TPen), la brocha (TBrush) y cualquier otro objeto gráfico seleccionado con el dispositivo de contexto TDC.

En el siguiente fragmento de código tomado del ejemplo 3.13 se ilustran estos cuatro pasos:

```
        // Paso 1 el dispositivo de contexto es dc y viene como parámetro
void TVentana::Paint(TDC& dc, BOOL /*erase*/, TRect& /*rect*/) {
        // Paso 2
        TColor Negro(0,0,0);
        TBrush RellenoNegro(Negro);
        TRect Rectangulo(100, 100, 250, 275);
        TRect RecElipse(125, 125, 260, 285);
        TColor Rojo(255,0,0);
        TPen Pluma(Rojo,4,PS_SOLID);

        // Paso 3
        dc.SelectObject(RellenoNegro); // Se activa el relleno
        dc.SelectObject(Pluma);// Se activa el Pluma
        dc.Ellipse(125, 50, 150, 100);
        dc.Ellipse(225, 50, 250, 100);
        dc.MoveTo(188,115);
        dc.LineTo(188,185);
        dc.Pie(138,200,238,250,138,200,238,200);
        // Paso 4
        dc.RestorePen();
        dc.RestoreBrush();
}
```

En el ejemplo 3.13 se presenta el código completo; la salida de este programa se ilustra en la Figura 3.23.

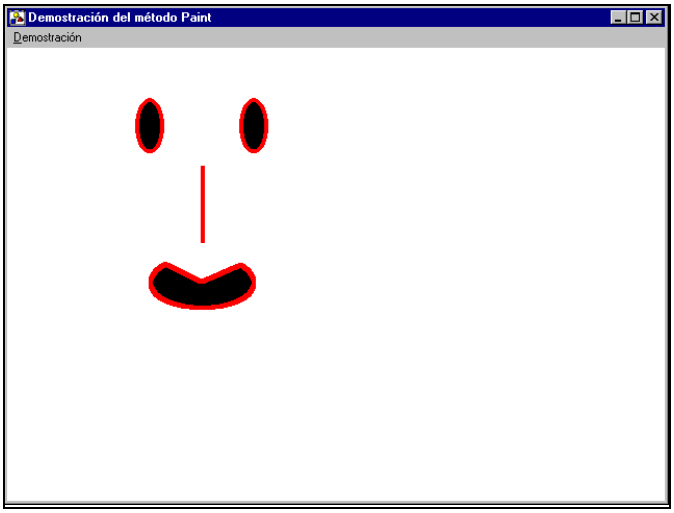


FIGURA 3.23
Salida Gráfica.

Ejemplo 3.13. Uso de las clases en OWL para gráficos

```
// E3_13.CPP
#include <owl\applicat.h>
#include <owl\framewin.h>
#include <owl\dc.h>
#include <owl\gdiobjec.h>
#include "e3_13.h"

class TVentana: public TFrameWindow {
public:
  TVentana(TWindow* parent, const char far* title);
  virtual void Paint(TDC& dc, BOOL erase, TRect& rect);
};

class TPrincipal: public TApplication {
public:
  TPrincipal() : TApplication() {};
  void InitMainWindow();
};
```

```
TVentana::TVentana(TWindow* parent, const char far* title)
  : TFrameWindow(parent, title), TWindow(parent, title) {
// Permite que la ventana se inicie máximizada
  Attr.X = 0;
  Attr.Y = 0;
  Attr.H = GetSystemMetrics(SM_CYSCREEN);
  Attr.W = GetSystemMetrics(SM_CXSCREEN);
  AssignMenu(E313_MENU);
}

void TVentana::Paint(TDC& dc, BOOL /*erase*/, TRect& /*rect*/) {
        TColor Negro(0,0,0);
        TBrush RellenoNegro(Negro);
        TColor Rojo(255,0,0);
        TPen Pluma(Rojo,4,PS_SOLID);
        dc.SelectObject(RellenoNegro);
        dc.SelectObject(Pluma);
        dc.Ellipse(125, 50, 150, 100);
        dc.Ellipse(225, 50, 250, 100);
        dc.MoveTo(188,115);
        dc.LineTo(188,185);
         dc.Pie(138,200,238,250,138,200,238,200);
        dc.RestorePen();
        dc.RestoreBrush();
}

void TPrincipal::InitMainWindow() {
  MainWindow = new TVentana(0, "Demostración del método Paint");
}

int OwlMain(int /* argc */ , char* /* argv */ []) {
  TPrincipal Ap;
  return Ap.Run();
}

// E3_13.RC
#include <owl\window.rh>
#include "e3_13.h"

E313_MENU MENU
BEGIN
```

```
        POPUP "&Demostración"
        BEGIN
                MENUITEM "&Salir", CM_EXIT
        END
END
```

// E3_13.H

```
#define E313_MENU 100
```

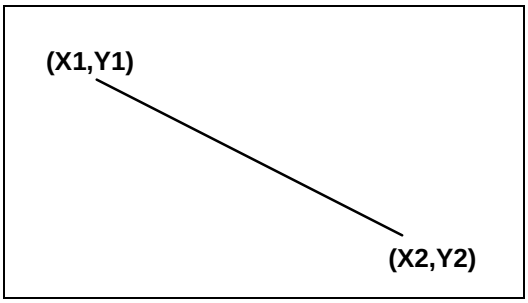
Como ya se ha visto en los ejemplos anteriores, los gráficos en la pantalla son dibujados a través de funciones (métodos) de la clase TDC. La clase TDC de OWL posee un gran número de estas funciones, algunas para dibujar rectángulos, líneas, elipses, otras para seleccionar brochas y colores y también para restaurar las brochas y colores predeterminados, entre otras. Basta colocar el cursor sobre la palabra TDC en Borland C++ y presionar la tecla F1 para obtener la información completa de todas las funciones de la clase TDC. Seguidamente presentamos algunas de las más utilizadas.

Líneas

Para dibujar una línea se deben usar dos métodos de la TDC, éstos son:

```
dc.MoveTo(X1,Y1);
dc.LineTo(X2,Y2);
```

El método MoveTo(X1,Y1) se encarga de establecer el punto inicial de la recta en (X1,Y1) y el método LineTo(X2,Y2) se encarga de dibujar una línea desde ese punto inicial hasta el punto final (X2,Y2), tal como se muestra en la siguiente Figura:

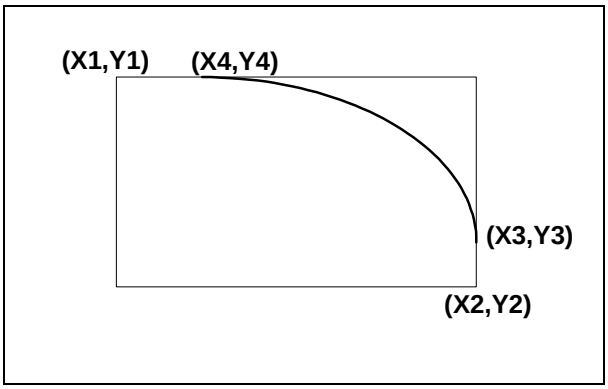


Arcos

Para dibujar un arco se debe usar la siguiente función de la clase TDC:

```
dc.Arc(X1, Y1, X2, Y2, X3, Y3, X4,Y4);
```

donde los puntos (X1,Y1) y (X2,Y2) determinan el rectángulo donde está inscrito el arco, (X3,Y3) es el punto inicial del arco y (X4,Y4) es el punto final del arco, como se muestra en la siguiente Figura:

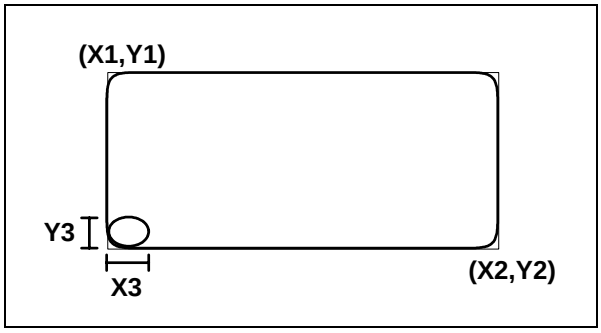


Rectángulos

Para dibujar rectángulos la clase TDC tiene varios métodos, dos de ellos son:

```
dc.Rectangle(X1, Y1, X2, Y2);
dc.RoundRect(X1, Y1, X2, Y2, X3, Y3);
```

en ambos métodos (X1,Y1) es la esquina superior izquierda y (X2,Y2) es la esquina inferior derecha. En el método RoundRect(X1, Y1, X2, Y2, X3, Y3) el punto (X3,Y3) define un rectángulo en el cual se inscribe una elipse que determina el redondeo de las esquinas. Vea la siguiente Figura:

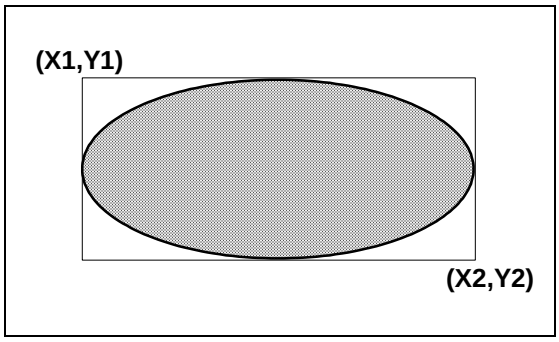


Elipses

Para dibujar una elipse se usa el siguiente método de la clase TDC:

```
dc.Ellipse(X1, Y1,X2,Y2);
```

donde los puntos (X1,Y1) y (X2,Y2) determinan un rectángulo donde está inscrita la elipse, como se muestra en el siguiente gráfico:

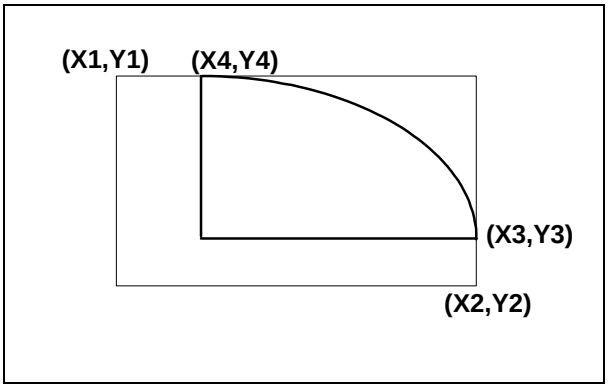


Pasteles

Para producir gráficos tipo pastel, la clase TDC tiene el método:

```
dc.Pie(X1, Y1, X2, Y2, X3, Y3, X4, Y4);
```

donde, al igual que en los arcos, los puntos (X1,Y1) y (X2,Y2) determinan en rectángulo donde está inscrito el arco, (X3,Y3) es el punto inicial del arco y (X4,Y4) es el punto final del arco, como se muestra en la siguiente Figura:



Texto

Como ya hemos mencionado, en Windows todos los objetos desplegados por pantalla son gráficos, incluso los textos. Para desplegar texto por pantalla se usa el método:

```
TextOut(X, Y, Str, strlen(Str));
```

donde (X,Y) es el punto inicial de despliegue de texto, Str es una variable de tipo hilera donde se almacena el texto y el último parámetro de la función es la longitud de la hilera; para evitar errores en este parámetro generalmente se usa la función strlen(Str) que retorna la longitud de Str.

TDC proporciona una función SetTextAlign(UINT flags) con la cual se puede cambiar el tipo de alineación del texto y una función SetTextColor(TColor color) para cambiar color del texto, entre otras.

Todos los métodos vistos hasta ahora poseen otras definiciones que permiten usar una variable de tipo TPoint en lugar de X1, Y1 y una variable del tipo TRect en lugar de X1, Y1, X2, Y2.

Selección y restauración de la pluma y la brocha

Para seleccionar la pluma y la brocha con que se va a dibujar, se deben declarar una variable tipo TPen y una de tipo TBrush respectivamente, luego se seleccionan con el método SelectObject de la clase TDC, como se ilustra en el siguiente fragmento de código:

```
TBrush RellenoNegro(Negro);
TPen Pluma(Rojo,4,PS_SOLID);
dc.SelectObject(RellenoNegro);
dc.SelectObject(Pluma);
```

Para restaurar la brocha y la pluma originales se utilizan los métodos:

```
dc.RestorePen();
dc.RestoreBrush();
```

En ejemplo 3.14 se ilustran algunas de las funciones presentadas anteriormente, además se analiza cómo se invoca el método Paint desde otros métodos.

Para que los gráficos sean redibujados automáticamente cuando se abre y cierra una ventana o cuando se usan los elevadores (scrolling), estos deben codificarse en el método Paint de la clase TVentana. Por esta razón en el ejemplo 3.14 el método que dibuja un rectángulo se implementa como sigue:

```
void TVentana::Rectangulo() {
        GrafAct=CRectangulo;
        Invalidate();
}
```

Invalidate() es un método de TWindow que se encarga de invalidar el área de trabajo de la ventana, por lo que automáticamente se invoca el método Paint de TVentana. Para que el Paint conozca que gráfico debe desplegar en un momento dado, se ha definido un tipo enumerado al inicio del ejemplo 3.14 denominado GraficoActivo en el cual se incluye un nombre para cada posible gráfico, como sigue:

```
enum GraficoActivo {Ninguno=0,CElipse,CRectangulo,CRecta,CTexto};
```

Además en la clase TVentana se define una variable GrafAct en la cual se llevará el control del gráfico activo en un momento dado, como se muestra en el siguiente fragmento de código:

```
class TVentana : public TWindow {
        GraficoActivo GrafAct;
public:
        TVentana(TWindow *Padre, const char far *titulo);
        . . . . . .
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};
```

La instrucción ***GrafAct=CRectangulo;*** en el método Rectángulo de TVentana lo único que hace es establecer como gráfico activo a CRectangulo para luego invalidar la pantalla para que sea redibujada adecuadamente.

En el ejemplo 3.14 el constructor de TVentana fue implementado como sigue:

```
TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                                : TWindow(Padre, titulo) {
        GrafAct=Ninguno;
        Attr.Style |= WS_VSCROLL | WS_HSCROLL;
        Scroller = new TScroller(this,20,20,300,300);
}
```

La instrucción GrafAct=Ninguno; tiene como propósito inicializar el gráfico activo en vacío.

Por otra parte las instrucciones:

```
Attr.Style |= WS_VSCROLL | WS_HSCROLL;
Scroller = new TScroller(this,20,20,300,300);
```

tienen como objetivo elevadores colocarle a la ventana principal, es decir, un "scroll" horizontal y uno vertical. Esto con el objetivo de ilustrar cómo el gráfico es reconstruido automáticamente cuando los botones del *scroll* son presionados.

En el ejemplo 3.14 el programa tiene una barra de herramientas mediante con la cual se pueden dibujar un círculo, rectángulo, una recta, un texto o bien borrar el gráfico. Una de las salidas de este programa se presenta en la Figura 3.24.

🖫 Ejemplo 3.14 Uso del Paint desde otros métodos de TVentana

```
// E3_14.CPP
#include <owl\owlpch.h>
#include <owl\framewin.h>
```

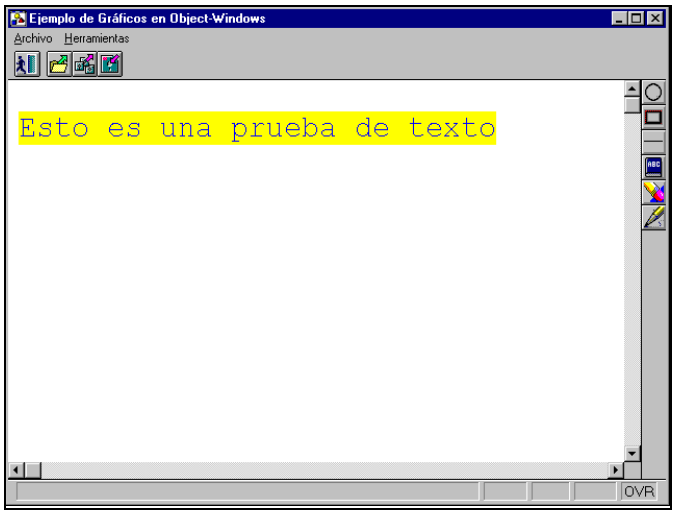


FIGURA 3.24
Salida Gráfica.

```
#include <owl\edit.h>
#include <owl\menu.h>
#include <owl\applicat.h>
#include <owl\decframe.h>
#include <owl\controlb.h>
#include <owl\gadget.h>
#include <owl\buttonga.h>
#include <owl\messageb.h>
#include <owl\statusba.h>
#include <owl\toolbox.h>
#include <owl\scroller.h>
#include "e3_14.h"

enum GraficoActivo {Ninguno=0,CElipse,CRectangulo,CRecta,
                    CTexto};

class TPrincipal : public TApplication {
            TControlBar* BarraIconos;
            TStatusBar*  BarraEstado;
            TToolBox*    BarraHerramientas;
 public:
           TPrincipal() : TApplication() {}
           void InitMainWindow();
```

```
};
class TVentana : public TWindow {
        GraficoActivo GrafAct;
public:
        TVentana(TWindow *Padre, const char far *titulo);
        virtual void Paint(TDC& dc, BOOL erase, TRect& rect);
        void EvLButtonDown(UINT, TPoint&);
        void EvRButtonDown(UINT, TPoint&);
        void Nuevo();
        void Abrir();
        void Guardar();
        void Recuperar();
        void Circulo();
        void Rectangulo();
        void Recta();
        void Texto();
        void Borrador();
        void Lapiz();
        void Salir();
        DECLARE_RESPONSE_TABLE(TVentana);
};

DEFINE_RESPONSE_TABLE1(TVentana, TWindow)
  EV_COMMAND(CM_NUEVO,Nuevo),
  EV_COMMAND(CM_ABRIR,Abrir),
  EV_COMMAND(CM_GUARDAR,Guardar),
  EV_COMMAND(CM_RECUPERAR,Recuperar),
  EV_COMMAND(CM_CIRCULO,Circulo),
  EV_COMMAND(CM_RECTANGULO,Rectangulo),
  EV_COMMAND(CM_RECTA,Recta),
  EV_COMMAND(CM_TEXTO,Texto),
  EV_COMMAND(CM_BORRADOR,Borrador),
  EV_COMMAND(CM_LAPIZ,Lapiz),
  EV_COMMAND(CM_SALIR,Salir),
END_RESPONSE_TABLE;

void TPrincipal::InitMainWindow() {
        TDecoratedFrame* NVentana = new TDecoratedFrame(0, "Ejemplo
         de Gráficos en Object-Windows",new TVentana(0,NULL),TRUE);
// Construye una BarraIconos
  BarraIconos = new TControlBar(NVentana);
```

```
  BarraIconos->Insert(*new TButtonGadget(IDB_SALIR,CM_SALIR));
  BarraIconos->Insert(*new TSeparatorGadget(6));
  BarraIconos->Insert(*new TButtonGadget(IDB_NUEVO,CM_NUEVO));
  BarraIconos->Insert(*new TButtonGadget(IDB_ABRIR,CM_ABRIR));
  BarraIconos->Insert(*new TButtonGadget(IDB_GUARDAR,
                                         CM_GUARDAR));
  // Habilta la  barra de estado a enviar mensajes asociados a los íconos
  BarraIconos->SetHintMode(TGadgetWindow::EnterHints);
  // Inserta la BarraIconos en la ventana NVentana
  NVentana->Insert(*BarraIconos, TDecoratedFrame::Top);
  BarraEstado = new TStatusBar(NVentana, TGadget::Recessed,
                               TStatusBar::CapsLock | TStatusBar::NumLock |
                               TStatusBar::ScrollLock | TStatusBar::Overtype);
  NVentana->Insert(*BarraEstado, TDecoratedFrame::Bottom);
  // Construye una Barra de herramientas
  BarraHerramientas = new TToolBox(NVentana, 1);  // una columna
  BarraHerramientas->Insert(*new TButtonGadget(IDB_CIRCULO,
                                               CM_CIRCULO));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTANGULO,
                                             CM_RECTANGULO));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_RECTA,
                                               CM_RECTA));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_TEXTO,
                                               CM_TEXTO));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_BORRADOR,
                                             CM_BORRADOR));
  BarraHerramientas->Insert(*new TButtonGadget(IDB_LAPIZ, CM_LAPIZ));
  NVentana->Insert(*BarraHerramientas, TDecoratedFrame::Right);
  // Para maximizar la ventana
  NVentana->Attr.X = 0;
  NVentana->Attr.Y = 0;
  NVentana->Attr.H = GetSystemMetrics(SM_CYSCREEN);
  NVentana->Attr.W = GetSystemMetrics(SM_CXSCREEN);
  SetMainWindow(NVentana);
  GetMainWindow()->AssignMenu("E3_14_MENU");
}

TVentana::TVentana(TWindow *Padre, const char far *titulo)
                                          : TWindow(Padre, titulo) {
        GrafAct=Ninguno;
        Attr.Style |= WS_VSCROLL | WS_HSCROLL;
```

```
        // Scroller es una variable de TWindow
        Scroller = new TScroller(this,20,20,300,300);
}

void TVentana::Paint(TDC& dc, BOOL /*erase*/, TRect& /*rect*/) {
        switch(GrafAct) {
                case CElipse : {
                        TColor Azul(0,0,255);
                        TBrush Relleno(Azul);
                        TRect RecElipse(125, 125, 260, 285);
                        TColor Rojo(255,0,0);
                        TPen Pluma(Rojo,4,PS_SOLID);
                        dc.SelectObject(Relleno);
                        dc.SelectObject(Pluma);
                        dc.Ellipse(RecElipse);
                        dc.RestorePen();
                        dc.RestoreBrush();
                        break;
                }
                case CRectangulo : {
                        TColor Verde(0,255,0);
                        TBrush Relleno(Verde,HS_CROSS);
                        TRect Rec(200, 50, 500, 300);
                        TColor Color(58,140,245);
                        TPen Pluma(Color,4,PS_SOLID);
                        dc.SelectObject(Relleno);
                        dc.SelectObject(Pluma);
                        dc.Rectangle(Rec);
                        dc.RestorePen();
                        dc.RestoreBrush();
                        break;
                }
                case CRecta : {
                        TPoint P1(10,30);
                        TPoint P2(300,70);
                        TPoint P3(150,195);
                        TColor Color(100,0,100);
                        TPen Pluma(Color,2,PS_SOLID);
                        dc.SelectObject(Pluma);
                        dc.MoveTo(P1);
                        dc.LineTo(P2);
```

```
                    dc.LineTo(P3);
                    dc.RestorePen();
                    break;
                }
                case CTexto : {
                    TFont* font;
                    font = new TFont(
                        "Courier New",
                        -::MulDiv(36, 96, 120),
                        0, 0, 0,
                        FW_NORMAL,
                        FIXED_PITCH,
                        FALSE, FALSE, FALSE,
                        ANSI_CHARSET,
                        OUT_TT_PRECIS,
                        CLIP_DEFAULT_PRECIS,
                        PROOF_QUALITY
                    );
                    dc.SelectObject(*font);
                    TPoint P1(10,30);
                    TColor Azul(0,0,255);
                    char aux[50];
                    strcpy(aux,"Esto es una prueba de texto");
                    dc.SetTextColor(Azul);
                    dc.SetBkColor(TColor::LtYellow);
                    dc.TextOut(P1,aux,strlen(aux));
                    dc.RestoreFont();
                    delete font;            // No olvidar esta sentencia
                    break;
                }
        }
}

void TVentana::Nuevo() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Abrir() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}
```

```
void TVentana::Guardar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Recuperar() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Circulo() {
        GrafAct=CElipse;
        Invalidate();
}

void TVentana::Rectangulo() {
        GrafAct=CRectangulo;
        Invalidate();
}

void TVentana::Recta() {
        GrafAct=CRecta;
        Invalidate();
}

void TVentana::Texto() {
        GrafAct=CTexto;
        Invalidate();
}

void TVentana::Borrador() {
        GrafAct=Ninguno;
        Invalidate();
}

void TVentana::Lapiz() {
  MessageBox("No implementado aún", "Opciones del menú", MB_OK);
}

void TVentana::Salir() {
        CloseWindow();
}
```

```
// Programa Principal en Object-Windows
int OwlMain(int /* argc */, char* /* argv */ [] ) {
  TPrincipal Ap;
  return Ap.Run();
}
```

// E3_14.H

```
#define CM_NUEVO     101
#define CM_ABRIR     102
#define CM_GUARDAR 103
#define CM_RECUPERAR      104
#define CM_SALIR     1
#define CM_MARCAR   105
#define CM_HABILITAR_DESHABILITAR       106
#define CM_MENSAJE  107
#define CM_BORRAR   108
#define CM_AGREGAR 109
#define CM_CIRCULO   110
#define CM_RECTANGULO 111
#define CM_RECTA  112
#define CM_TEXTO  113
#define CM_BORRADOR       114
#define CM_LAPIZ     115

// Identificadores de los íconos en el ToolBar
#define IDB_SALIR      1001
#define IDB_NUEVO 1101
#define IDB_ABRIR     1102
#define IDB_GUARDAR1103
#define IDB_CIRCULO   1110
#define IDB_RECTANGULO 1111
#define IDB_RECTA  1112
#define IDB_TEXTO  1113
#define IDB_BORRADOR       1114
#define IDB_LAPIZ     1115
```

// E3_14.RC

```
#include "e3_14.h"

E3_14_MENU MENU
{
```

```
 POPUP "&Archivo"
 {
  MENUITEM "&Nuevo", CM_NUEVO
  MENUITEM "&Abrir", CM_ABRIR
  MENUITEM "&Guardar", CM_GUARDAR
  MENUITEM "&Recuperar", CM_RECUPERAR
  MENUITEM SEPARATOR
  MENUITEM "&Salir", CM_SALIR
 }
 POPUP "&Herramientas"
 {
  MENUITEM "&Círculo", CM_CIRCULO
  MENUITEM "&Rectángulo", CM_RECTANGULO
  MENUITEM "R&ecta", CM_RECTA
  MENUITEM "&Texto", CM_TEXTO
  MENUITEM "&Borrar", CM_BORRADOR
  MENUITEM "L&ápiz", CM_LAPIZ
 }
}

IDB_SALIR BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 87 87 87 84 08 FF FF FF FF F8'
 '00 00 88 88 88 87 48 F4 46 68 68 F8 00 00 48 88'
 '87 40 48 F4 76 66 86 F8 00 00 04 88 04 04 88 F4'
 '46 68 68 F8 00 00 40 48 40 48 88 F4 76 66 86 F8'
 '00 00 74 04 04 88 88 F4 46 68 68 F8 00 00 87 40'
 '40 88 88 F4 76 66 86 F8 00 00 88 74 04 88 88 F4'
 '46 68 68 F8 00 00 88 70 40 84 88 F4 76 66 86 F8'
 '00 00 04 04 04 84 88 F4 46 68 68 F8 00 00 48 40'
 '40 84 88 F4 76 66 86 F8 00 00 88 04 04 04 88 F4'
 '46 68 68 F8 00 00 88 70 40 48 88 F4 76 66 86 F8'
 '00 00 88 87 44 88 88 F4 46 68 68 F8 00 00 88 04'
 '78 88 88 F4 76 66 86 F8 00 00 88 44 48 88 88 F4'
```

```
 '46 68 68 F8 00 00 88 04 08 88 88 F4 76 66 66 F8'
 '00 00 88 88 88 88 88 F4 47 77 78 F8 00 00 88 88'
 '88 88 88 FF FF FF FF F8 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_NUEVO BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '00 00 80 00 00 00 00 00 00 00 88 88 00 00 80 0F'
 'BF BF BF BF BF B0 88 88 00 00 80 0B FB FB FB FB'
 'FB F0 88 88 00 00 80 70 BF BF BF BF BF BF 08 88'
 '00 00 80 B0 FB FB FB FB FB FB 08 88 00 00 80 70'
 'BF BF BF BF BF BF 08 88 00 00 80 B7 0B FB FB FB'
 'FB FB F0 88 00 00 80 7B 0F BF BF BF BF BF B0 88'
 '00 00 80 B7 00 00 00 00 00 00 00 88 00 00 80 7B'
 '7B 7B 0A EA 0B 08 88 88 00 00 80 00 B7 B0 00 AE'
 'A0 08 80 88 00 00 88 88 00 08 88 0A EA 08 00 88'
 '00 00 88 88 88 88 88 80 AE A0 A0 88 00 00 88 88'
 '88 88 88 88 0A EA E0 88 00 00 88 88 88 88 88 88'
 '80 AE A0 88 00 00 88 88 88 88 88 88 0A EA E0 88'
 '00 00 88 88 88 88 88 80 00 00 00 88 00 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_ABRIR BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
```

```
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 80 00 00 00 00'
 'EF 73 88 88 88 88 88 80 F6 67 66 60 00 8C 88 88'
 '88 88 88 80 F6 87 86 60 00 00 80 00 00 00 00 80'
 'F8 70 78 60 00 00 80 F6 60 66 60 80 F6 08 06 60'
 '00 00 80 F6 80 86 60 80 F8 70 78 60 00 00 80 F8'
 '70 78 60 80 77 77 86 00 00 00 80 F6 08 06 60 00'
 '00 07 8F 80 00 00 80 F8 70 78 60 76 66 00 00 00'
 '00 00 80 F6 86 86 00 78 66 08 88 88 00 00 80 FF'
 '8F 8F 80 07 86 08 88 88 00 00 80 00 00 00 00 80'
 '60 88 88 88 00 00 88 88 88 0F 87 06 0D 08 88 88'
 '67 14 88 88 88 0F 68 60 DD D0 88 08 8A 44 88 88'
 '88 0F F8 F8 0D DD 00 08 CD EB 88 88 88 00 00 08'
 '80 DD DD 08 41 98 88 88 88 88 88 88 88 0D DD 08'
 'E9 F3 88 88 88 88 88 88 88 0D DD 08 93 30 88 88'
 '88 88 88 88 80 00 00 08 26 69 88 88 88 88 88 88'
 '88 88 88 88 32 26'
}

IDB_GUARDAR BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 00 00 00 00 00 00 00 00 08 23 10 88 0F'
 '76 76 76 77 67 67 67 08 88 C8 88 0F 66 66 66 7F'
 '66 66 66 08 00 31 88 0F 66 66 66 0F 66 66 67 08'
 '10 25 88 0F 66 66 66 06 66 66 66 08 99 55 88 0F'
 '66 66 66 66 66 66 67 08 10 25 88 0F 66 66 6F FF'
 '66 66 66 08 9D 44 88 0F 66 66 07 88 F6 66 67 08'
 '22 26 88 0F 66 66 07 88 86 66 66 08 00 01 88 0F'
 '66 66 07 80 00 00 07 08 19 A2 88 0F 66 66 60 80'
 'DD D0 66 08 20 02 88 0F 66 66 66 60 DD D0 67 08'
 '44 4C 88 0F D9 D9 66 60 DD DD 06 88 01 03 88 0F'
 '9D 9D 66 60 00 DD D0 88 22 44 88 0F D9 D9 66 60'
 '66 0D DD 08 41 26 88 0F FF FF FF FF F6 60 DD D0'
```

```
 '21 91 88 00 00 00 00 00 00 88 0D 08 02 54 88 88'
 '88 88 88 88 88 88 80 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE
{
  CM_SALIR, "Selecciona salir del programa"
  CM_NUEVO, "Crear un archivo nuevo"
  CM_ABRIR, "Abrir un archivo existente"
  CM_GUARDAR, "Guardar un archivo existente"
}

IDB_CIRCULO BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 88 88 87 00 00 78 88 88 88 23 10 88 88'
 '87 00 07 70 00 78 88 88 88 C8 88 88 70 08 88 88'
 '80 07 88 88 00 31 88 87 07 88 88 88 88 70 78 88'
 '10 25 88 80 08 88 88 88 87 80 08 88 99 55 88 70'
 '88 88 88 88 88 88 07 88 10 25 88 07 88 88 88 88'
 '88 88 70 88 9D 44 88 07 88 88 88 88 88 88 70 88'
 '22 26 88 07 88 88 88 88 88 88 70 88 00 01 88 07'
 '88 88 88 88 88 88 70 88 19 A2 88 07 88 88 88 88'
 '88 88 70 88 20 02 88 70 88 88 88 88 88 88 07 88'
 '44 4C 88 80 08 88 88 88 88 80 08 88 01 03 88 87'
 '07 88 88 88 88 70 78 88 22 44 88 88 70 07 88 88'
 '70 07 88 88 41 26 88 88 87 00 07 70 00 78 88 88'
 '21 91 88 88 88 87 00 00 78 88 88 88 02 54 88 88'
 '88 88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

IDB_RECTANGULO BITMAP
```

```
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '44 4C 88 88 88 88 88 88 88 88 88 88 23 10 87 77'
 '77 77 77 77 77 77 77 78 88 C8 87 00 00 00 00 00'
 '00 00 01 78 00 31 87 90 00 00 00 00 00 00 00 78'
 '10 25 87 00 77 77 77 77 77 77 00 78 99 55 87 90'
 '78 88 88 88 88 87 00 78 10 25 87 00 78 88 88 88'
 '88 87 00 78 9D 44 87 90 78 88 88 88 88 87 00 78'
 '22 26 87 00 78 88 88 88 88 87 00 78 00 01 87 90'
 '78 88 88 88 88 87 00 78 19 A2 87 00 78 88 88 88'
 '88 87 00 78 20 02 87 90 78 88 88 88 88 87 00 78'
 '44 4C 87 00 77 77 77 77 77 77 00 78 01 03 87 90'
 '00 00 00 00 00 00 00 78 22 44 87 19 09 09 09 09'
 '09 09 09 78 41 26 87 77 77 77 77 77 77 77 77 78'
 '21 91 88 88 88 88 88 88 88 88 88 88 02 54 88 88'
 '88 88 88 88 88 88 88 45 F4 88 88 88 88 88 88'
 '88 88 88 88 0B 0A'
}

IDB_RECTA BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 88 88 88 88 88 88 88 88 65 F0 88 88'
 '88 88 88 88 88 88 88 65 F0 88 88 88 88 88 88'
 '88 88 88 88 65 F0 88 88 88 88 88 88 88 88 88 88'
 '65 F0 88 88 88 88 88 88 88 88 88 65 F0 88 88'
 '88 88 88 88 88 88 88 A2 C0 88 88 88 88 88 88'
```

```
 '88 88 88 88 00 00 88 88 88 88 88 88 88 88 88 88'
 '88 88 88 88 88 88 88 88 88 88 88 88 04 6D 80 00'
 '00 00 00 00 00 00 00 08 08 80 88 88 88 88 88 88'
 '88 88 88 88 01 09 88 88 88 88 88 88 88 88 88 88'
 '00 88 88 88 88 88 88 88 88 88 88 88 00 E0 88 88'
 '88 88 88 88 88 88 88 88 2A 20 88 88 88 88 88 88'
 '88 88 88 88 22 A4 88 88 88 88 88 88 88 88 88 88'
 '04 08 88 88 88 88 88 88 88 88 88 88 99 00 88 88'
 '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

IDB_TEXTO BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 C0 C0 C0 00 80 80 80 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 77 80 00 00 00 00 00 00 00 87'
 '00 00 77 08 77 87 87 87 87 87 80 87 00 00 77 07'
 '78 78 78 78 78 78 80 87 00 00 77 00 48 48 48 48'
 '48 48 40 87 00 00 77 08 00 00 00 00 00 00 00 87'
 '00 00 77 07 44 44 44 44 44 44 40 87 00 00 77 07'
 '44 44 44 44 44 44 40 87 00 00 77 07 44 44 44 44'
 '44 44 40 87 00 00 77 07 44 44 44 44 44 44 40 87'
 '00 00 77 07 44 44 44 44 44 44 40 87 00 00 77 07'
 '4F 4F 4F F7 47 F7 40 87 00 00 77 07 4F 4F 4F 4F'
 '4F 4F 40 87 00 00 77 07 4F 7F 4F F7 4F 44 40 87'
 '00 00 77 07 4F 4F 4F 4F 4F 4F 40 87 00 00 77 07'
 '46 F6 4F F7 47 F7 40 87 00 00 77 07 44 44 44 44'
 '44 44 40 77 00 00 77 07 44 44 44 44 44 44 40 77'
 '00 00 77 08 84 44 44 44 44 44 40 77 00 00 77 70'
 '00 00 00 00 00 00 08 77 00 00 77 77 77 77 77 77'
 '77 77 77 77 00 00'
}

IDB_BORRADOR BITMAP
{
```

```
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 C0 C0 C0 00 80 80 80 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 77 77 77 7C CC CC C0 CC CC CC'
 '00 00 77 77 77 7C CC CC C0 C9 99 CC 00 00 77 77'
 '77 77 CC CC 09 99 99 9C 00 00 77 77 77 77 7C C0'
 '99 99 9D DD 00 00 77 77 77 77 77 09 99 99 DD DD'
 '00 00 77 77 77 77 70 99 99 9D DD DD 00 00 77 77'
 '77 77 0B 09 99 DD DD FD 00 00 77 77 77 77 0B 09'
 '99 DD DD FD 00 00 77 77 77 70 B0 B0 9D DD DF DD'
 '00 00 77 77 77 0B 0B 0B 0D DD DD DC 00 00 77 77'
 '70 B9 B0 B0 BC DD DD CC 00 00 77 77 0B 9B BB 0B'
 'CF CD DC CC 00 00 77 70 B9 BB BB BC FC F0 7C CC'
 '00 00 77 70 B9 BB BB BC FC F0 7C CC 00 00 77 0B'
 '9B BB BB BF CF 07 77 CC 00 00 70 B9 BB BB BB FF'
 'F0 77 77 7C 00 00 7B 9B BB BB BF FF 07 77 77 77'
 '00 00 79 BB BB BB FF F0 77 77 77 77 00 00 7B BB'
 'BB BF FF 07 77 77 77 77 00 00 77 77 77 77 77 77'
 '77 77 77 77 00 00'
}

IDB_LAPIZ BITMAP
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 10 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 80 44 47 77 78 78 78 78 88 88'
 '00 00 87 00 78 88 88 88 88 88 48 88 00 00 88 00'
 '00 88 88 88 88 88 87 88 00 00 88 70 07 07 88 88'
 '88 88 48 88 00 00 88 80 77 70 08 88 88 84 88 88'
 '00 00 88 88 08 37 70 78 88 87 88 88 00 00 88 88'
 '74 83 77 00 88 88 87 87 00 00 88 88 80 38 37 74'
 '07 88 88 88 00 00 88 88 88 08 83 77 44 08 88 88'
```

```
 '00 00 88 88 88 74 88 37 74 70 88 88 00 00 88 88'
 '88 80 88 87 44 47 08 88 00 00 88 88 88 88 08 B7'
 '78 34 70 88 00 00 88 88 88 88 70 07 B3 33 47 08'
 '67 14 88 88 88 88 80 3B FB 33 34 70 8A 44 88 88'
 '88 88 88 03 BB B3 33 47 CD EB 88 88 88 88 88 80'
 '3B FB 33 34 41 98 88 88 88 88 88 08 03 BF B3 33'
 'E9 F3 88 88 88 88 88 88 80 3B BB 33 93 30 88 88'
 '88 88 88 88 88 03 BF B3 26 69 88 88 88 88 88 88'
 '88 80 3B FB 32 26'
}
```