4. Desarrollo del sistema

4.1 Requerimientos y justificación

Necesitamos tener un IDE(Entorno de Desarrollo Integrado), para este proyecto recomendamos NetBeans 6.5 por ser de software libre y tener un apoyo por parte de Sun Microsystems, además de tener las librerías para realizar las conexiones con MySQL que será el manejador de bases de datos que se propone utilizar.

Como ya tenemos el planteamiento realizado en el trabajo anterior, únicamente cabe destacar algunas modificaciones de tipo funcional y ajustándolo a las herramientas utilizadas para el desarrollo.

En el trabajo anterior se ha propuesto la utilización de tecnologías PHP con MySQL, Linux CentOS y Samba. Ahora se propone el uso de Java con MySQL por ser ambas herramientas multiplataformas, lo cual facilitará su desarrollo en cualquier entorno o sistema operativo.

4.2 Creación de la base de datos

En principio se tiene que crear la base de datos que contendrá toda la estructura (tablas y relaciones entre los distintos datos).

Mediante el mysql command line client, al abrirlo pide una contraseña, para nuestro caso será nula es decir, con un solo enter estará listo el programa para comenzar a trabajar, iniciamos con la sentencia:

Create database TOOLBDH;

Con lo que se creará una base de datos llamada toolbdh como se propone en principio, con esta sentencia, únicamente se estará reservando un espacio en memoria para ir llenando la base de datos, es decir, la base recién creada, se encuentra vacía, por lo que ahora se creará la tabla que contendrá los registros, mediante las siguientes sentencias:

Use TOOLBDH;

Con lo que indicamos que estaremos trabajando sobre esta base de datos.

mysql> create table ingreso(nombreProyecto varchar(50), fechaInicio date, fechaFin date, tiempoEstimado varchar(10), Duracion varchar(10), Objetivos varchar(250), objetivosNoLogrados varchar(250), costoEstimado varchar(20), costoReal varchar(20));

Con esta sentencia creamos la estructura que llevará nuestra tabla que contiene los registros, tales como: nombre del proyecto, fecha de inicio, fecha de finalización, duración, objetivos y costos, entre otros.

4.3 Creación del proyecto

Una vez con la base de datos creada, abrimos el IDE Netbeans, seleccionamos File->New Project-> y a continuación aparecerá una ventana como en la figura 4.1

Dentro de la categoría Java elegimos el tipo de proyecto Aplicación Java de Escritorio (Fig.4.1), damos clic en Siguiente.

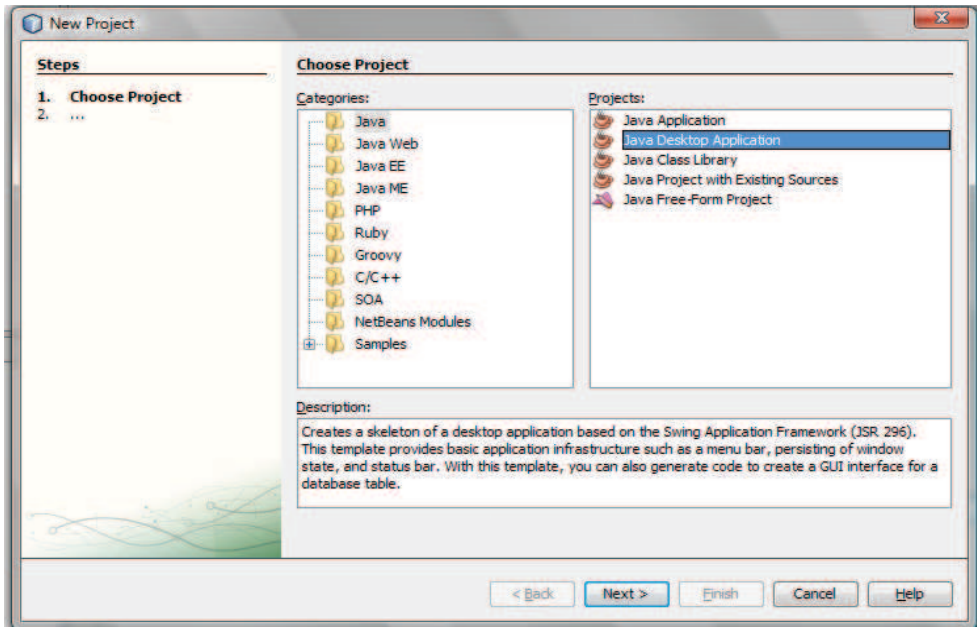


Fig.4.1 Selección de proyecto en Netbeans

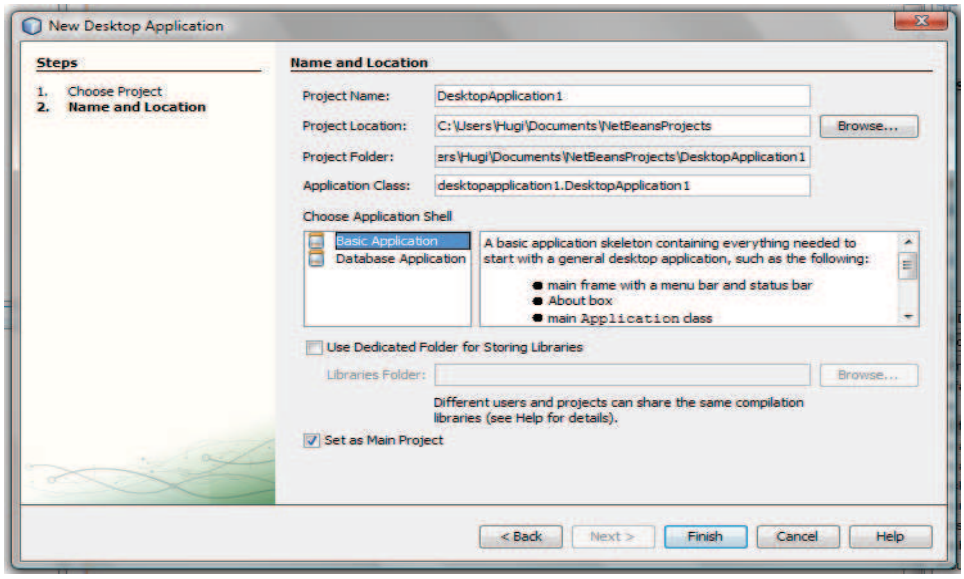


Fig.4.1.1 Nombre y localización del proyecto

En la nueva ventana (Fig.4.1.1) se introducirá el nombre de nuestro proyecto, así como la ubicación en el disco duro de nuestro sistema y para finalizar seleccionamos la opción de Aplicación Básica y damos clic en Finalizar.

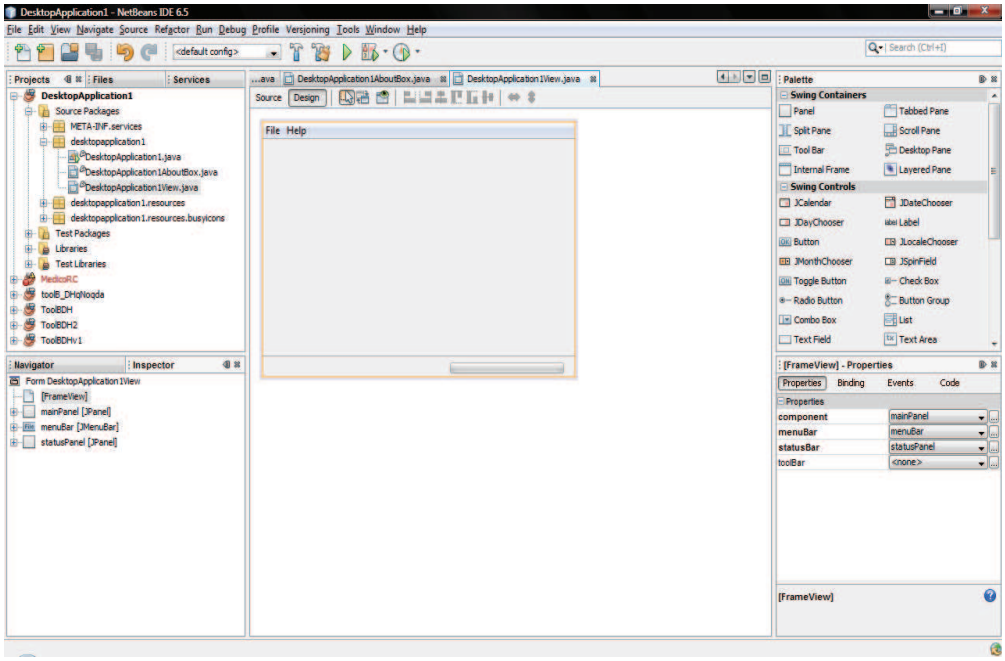


Fig.4.1.2 Presentación básica del proyecto

4.4 Diseño de la aplicación

Se nos muestra una ventana sin ninguna decoración (Fig.4.1.2), esta será el lienzo sobre el cual comenzaremos nuestro programa, ahora iremos implantando cada uno de los siguientes elementos, arrastrándolos, desde el Palette Manager(Lado superior derecho de la Fig.4.1.2):

jLabel “Usuario” acompañado por un jTextField

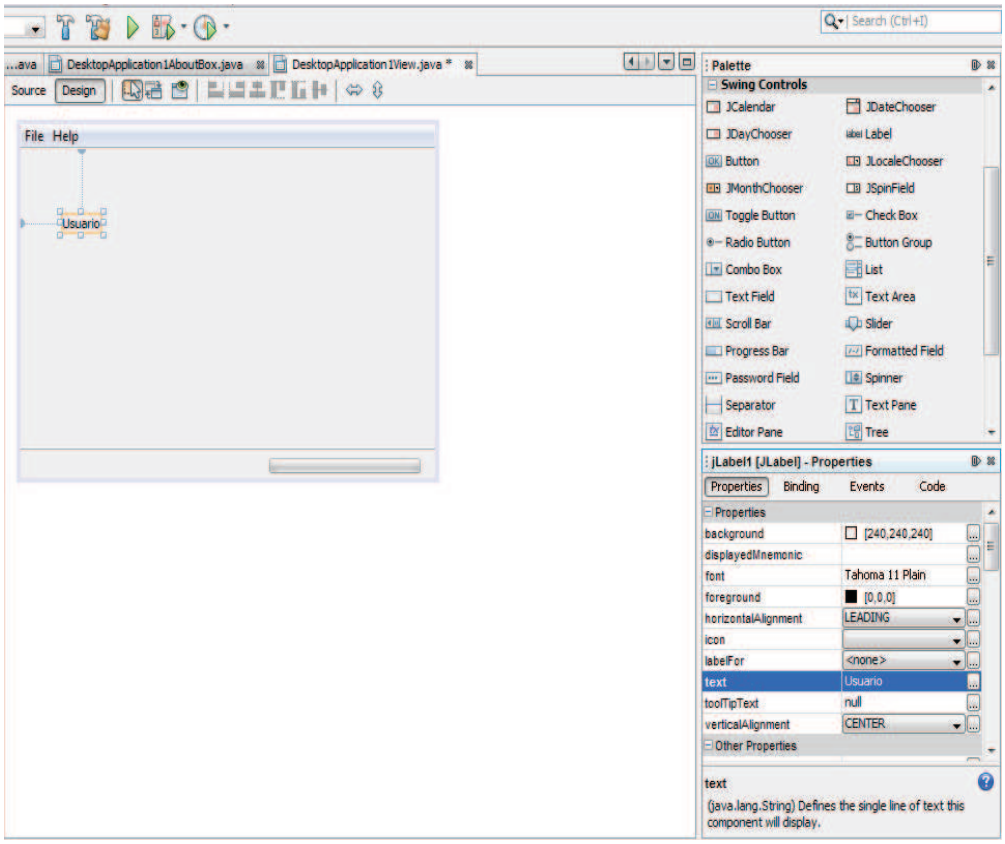


Fig.4.1.3 Propiedades de los controles Swing

Arrastramos desde el palette manager un jlabel(En swing controls dentro de palette) hasta la ventana sin decoración, una vez colocado, lo seleccionamos y en el menú de properties(Debajo del palette manager), buscamos por la propiedad text, aquí cambiamos el predeterminado por “Usuario”, ahora de la misma manera, dentro del palette manager, buscamos y arrastramos a la ventana un jtextfield, de tal forma que quede colocada junto a nuestro jlabel que acabamos de ubicar. Al igual que con la jlabel, dentro de las propiedades del jtextfield, buscamos por text y borramos el contenido para que el interior de este quede en blanco.

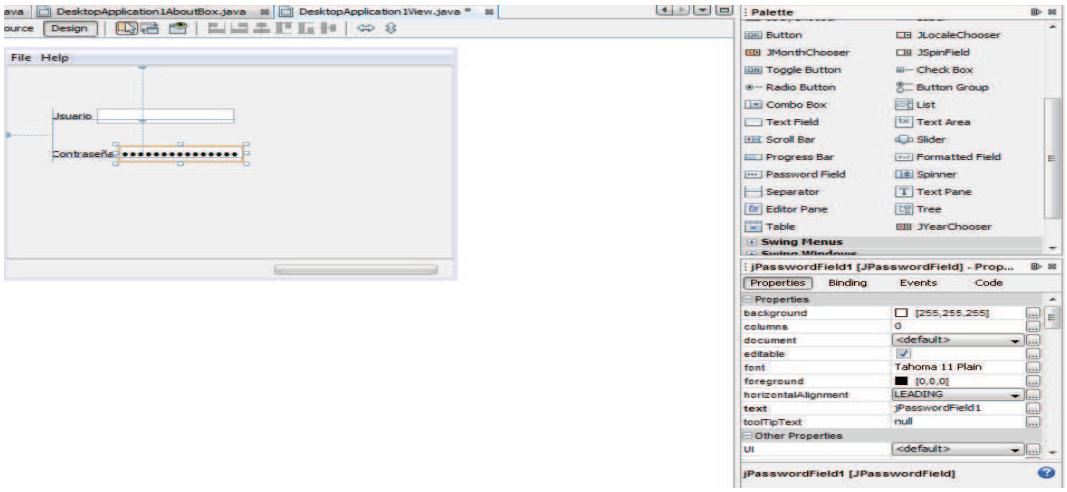


Fig.4.1.4 jLabel "Usuario" junto con un jPasswordField "Contraseña"

De la misma forma que en el paso anterior, arrastramos desde el palette manager un jlabel hacia la ventana de trabajo y cambiamos en sus propiedades el text por el de "Contraseña", ahora arrastramos el componente de password field hacia la ventana, lo ubicamos y en sus propiedades, cambiamos lo que tenga en text, para dejarlo nulo y dejar libre el campo.

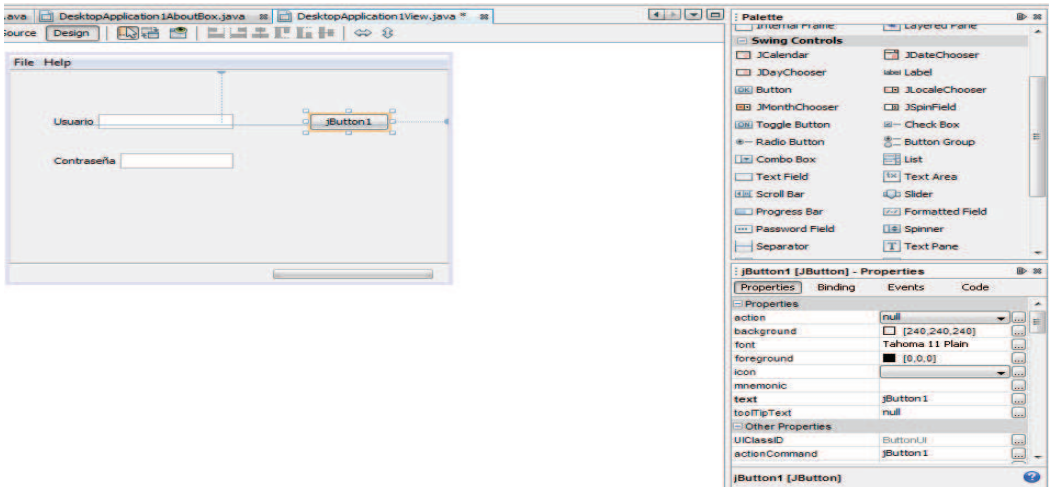


Fig.4.1.5 jButton "Crear"

Para la colocación de los jbutton o botones, necesitamos arrastrar desde el palette manager el componente ok button, dentro de los swing controls, una vez colocado el componente dentro de la ventana, lo seleccionamos y dentro de las propiedades de este buscamos el text y cambiamos su contenido por el de ingresar.

jButton “Ingresar”

Se procede de la misma forma que con el botón ingresar, solo que en este caso se cambia el contenido de “text” por el de “Crear”

jButton “Aceptar”

Lo mismo que en los casos anteriores, el text contendrá “Aceptar”

jButton “Cancelar”

En este caso el text será “Cancelar”

Con esto hemos terminado la ventana principal de nuestro programa, la ventana debería tener un aspecto similar al de la siguiente figura:

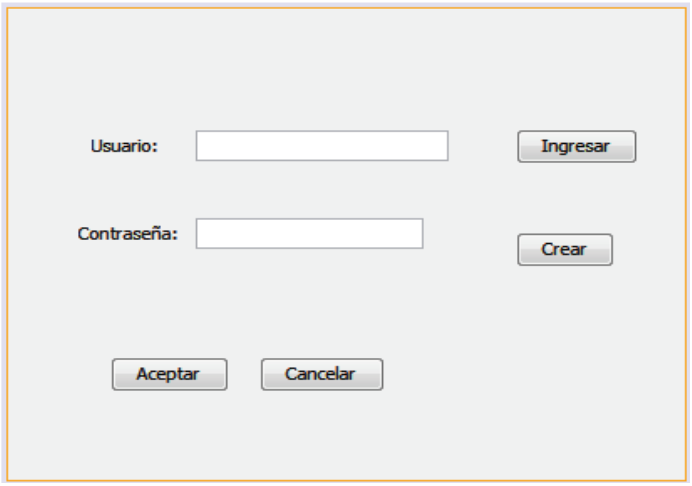


Fig.4.1.6 Ventana principal

4.5 Código de la aplicación

Ahora lo que tendremos que agregar es un método para poder cifrar nuestros datos, para tal efecto, damos clic en Source como en la Fig.4.1.7

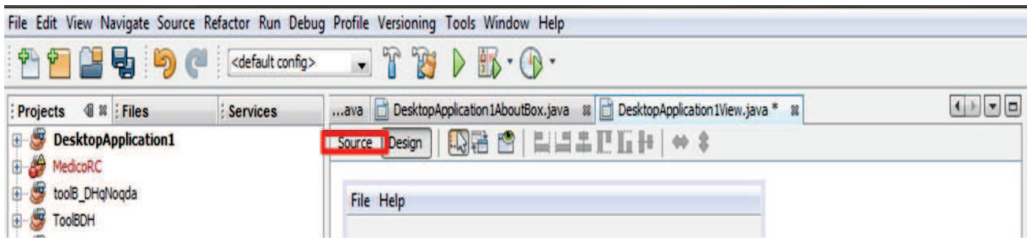


Fig.4.1.7 Pestaña Source

Con lo que se abrirá una nueva ventana que contiene el código fuente de nuestra aplicación, una vez dentro del código, buscamos la línea:

// Variables declaration - do not modify

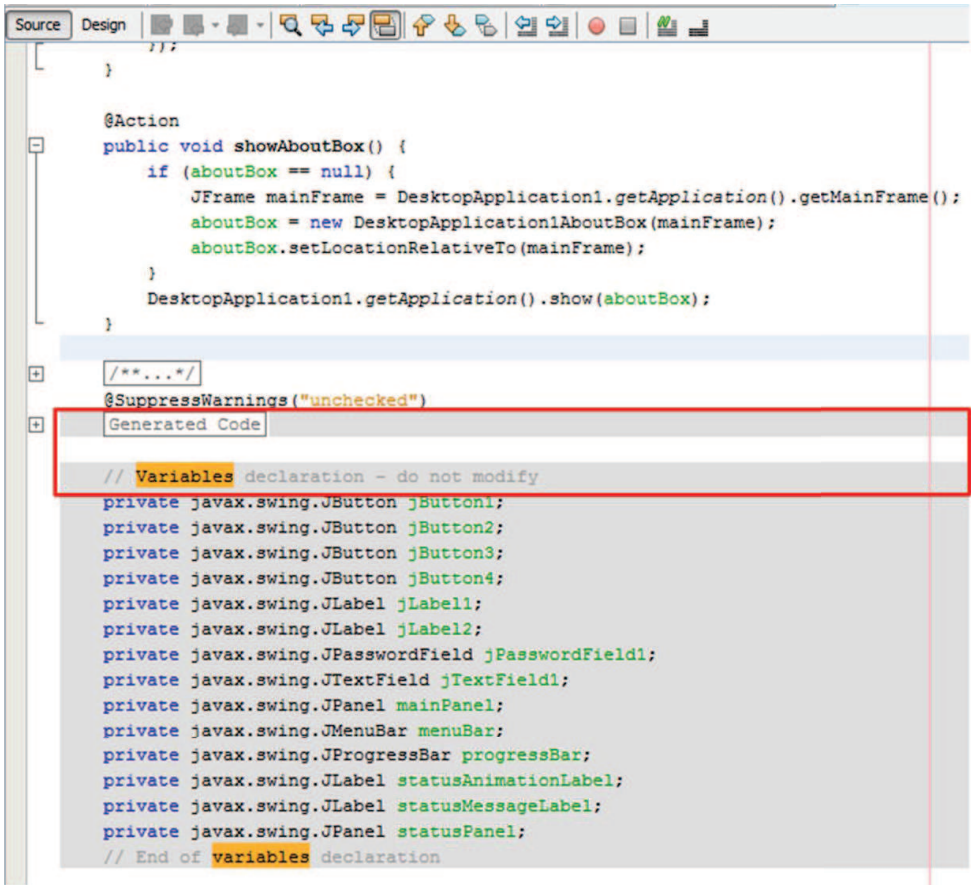


Fig.4.1.8 Localización del código

Y copiamos el siguiente código antes de esta línea, en el renglón en blanco de la fig-4.1.8

```
private static final char[] HEXADECIMAL = {'0', '1', '2', '3',  '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};
   public String hash(String stringToHash) {
      try {
         MessageDigest md = MessageDigest.getInstance("SHA");
         byte[] bytes = md.digest(stringToHash.getBytes());
         StringBuilder sb = new StringBuilder(2 * bytes.length);
         for (int k = 0; k < bytes.length; k++) {
            int low = (int) (bytes[k] & 0x0f);
            int high = (int) ((bytes[k] & 0xf0) >> 4);
            sb.append(HEXADECIMAL[high]);
            sb.append(HEXADECIMAL[low]);
         }
         return sb.toString();
      } catch (NoSuchAlgorithmException e) {
         //exception handling goes here
         return null;
      }
   }
```

4.5.1 Manejo de eventos jButton

Aceptar: Damos clic derecho sobre este botón, a continuación seleccionamos la opción **Events->Action->actionPerformed** , con esto se abrirá el código fuente

justo en el lugar donde tendremos que ajustar las acciones de este botón, ahora procedemos a agregar el siguiente código:

```
String user = hash(tfUsuario.getText());
        String contra = hash(tfContraseña.getText());
        File filK = new File("Mu.md5");
        File filL = new File("Mc.md5");
        File filA = new File("Uu.md5");
        File filB = new File("Uc.md5");
        if (tfUsuario.getText().isEmpty() || tfContraseña.getText().isEmpty()) {
           JOptionPane.showMessageDialog(null, "No pueden ser datos nulos");
        } else {
           if (filA.exists() && filB.exists()) {
              FileReader leebe = null;
              try {
                 String clv = "";
                 leebe = new FileReader("Uu.md5");
                 BufferedReader leebi = new BufferedReader(leebe);
                 String lector;
                 while ((lector = leebi.readLine()) != null) {
                    clv += lector;
                 }
                 FileReader leeb = null;
                 try {
                    String cl = "";
                    leeb = new FileReader("Uc.md5");
                    BufferedReader lee = new BufferedReader(leeb);
                    String reactor;
```

```
            while ((reactor = lee.readLine()) != null) {
               cl += reactor;
            }

            if (clv.equals(user) || cl.equals(contra)) {
               jButton3.setVisible(true);
               //jButton4.setVisible(true);
            }

         } catch (Exception ex) {
            JOptionPane.showMessageDialog(null, "" + ex);
         }

      } catch (Exception ex) {
         JOptionPane.showMessageDialog(null, "" + ex);
      }
   }

   if (filK.exists() && filL.exists()) {
      FileReader leebe = null;
      try {
         String clv = "";
         leebe = new FileReader("Mu.md5");
         BufferedReader leebi = new BufferedReader(leebe);
         String lector;
         while ((lector = leebi.readLine()) != null) {
            clv += lector;
```

```
        }
        FileReader leeb = null;
        try {
            String cl = "";
            leeb = new FileReader("Mc.md5");
            BufferedReader lee = new BufferedReader(leeb);
            String reactor;
            while ((reactor = lee.readLine()) != null) {
                cl += reactor;
            }

            if (clv.equals(user) || cl.equals(contra)) {
                jButton3.setVisible(true);
                jButton4.setVisible(true);
            }

        } catch (Exception ex) {
            JOptionPane.showMessageDialog(null, "" + ex);
        }

    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "" + ex);
    }
} else {
    try {
        FileWriter fw = new FileWriter("Mu.md5");
        BufferedWriter bw = new BufferedWriter(fw);
```

```
            PrintWriter salida = new PrintWriter(bw);
            salida.print("" + user);
            salida.close();

        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "" + e);
        }
        try {
            FileWriter fw = new FileWriter("Mc.md5");
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter salida = new PrintWriter(bw);
            salida.print("" + contra);
            salida.close();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "" + e);
        }
    }
    tfUsuario.setText(null);
    tfContraseña.setText(null);
}
```

Cancelar: Se procede en la misma forma que con el botón aceptar, pero en esta ocasión el código es distinto:

```
tfUsuario.setText(null);
tfContraseña.setText(null);
```

Ingresar: De igual forma que con los botones anteriores, en este agregaremos un código que nos permitirá ya ingresar a la base de datos para su consulta y edición:

```
JFrame mainFrame = ToolWk2App.getApplication().getMainFrame();
N4 db = new N4(mainFrame, false);
db.setVisible(true);
```

Crear: El código que agregaremos para este botón, será el siguiente:

```
String user = hash(tfUsuario.getText());
      String contra = hash(tfContraseña.getText());
      File filA = new File("Uu.md5");
      File filB = new File("Uc.md5");
      if (tfUsuario.getText().isEmpty() || tfContraseña.getText().isEmpty()) {
        JOptionPane.showMessageDialog(null, "Introduzca los datos de usuario y oprima de nuevo este botón");
      } else {
        if (filA.exists() && filB.exists()) {
          FileReader leebe = null;
          try {
            String clv = "";
            leebe = new FileReader("Uu.md5");
            BufferedReader leebi = new BufferedReader(leebe);
            String lector;
            while ((lector = leebi.readLine()) != null) {
              clv += lector;
            }
            FileReader leeb = null;
```

```
                try {
                    String cl = "";
                    leeb = new FileReader("Uc.md5");
                    BufferedReader lee = new BufferedReader(leeb);
                    String reactor;
                    while ((reactor = lee.readLine()) != null) {
                        cl += reactor;
                    }
                    if (clv.equals(user) && cl.equals(contra)) {
                        JOptionPane.showMessageDialog(mainPanel, "Ya existe ese
usuario");
                    }
                    else{JOptionPane.showMessageDialog(mainPanel, "Verifique los
datos");}
                } catch (Exception ex) {
                    JOptionPane.showMessageDialog(null, "" + ex);
                }

            } catch (Exception ex) {
                JOptionPane.showMessageDialog(null, "" + ex);
            }
        } else {
            try {
                FileWriter fw = new FileWriter("Uu.md5");
                BufferedWriter bw = new BufferedWriter(fw);
                PrintWriter salida = new PrintWriter(bw);
                salida.print("" + user);
```

```
            salida.close();

            FileWriter fw1 = new FileWriter("Uc.md5");
            BufferedWriter bw1 = new BufferedWriter(fw1);
            PrintWriter salida1 = new PrintWriter(bw1);
            salida1.print("" + contra);
            salida1.close();
            if (filA.exists() && filB.exists()) {
          JOptionPane.showMessageDialog(null, "Usuario Creado");}
          } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "" + e);
          }
        }

        tfUsuario.setText(null);
        tfContraseña.setText(null);
    }
```

4.6 Contenedor Base de datos

Ahora hace falta crear el contenedor de nuestra base de datos; creamos un nuevo Formulario de ejemplo Maestro/Detalle, dando click derecho en nuestra carpeta de fuente principal y seleccionando esta opción.

El nombre podría ser TOOLBDH como se sugiere en el trabajo anterior.

Se nos pedirá elegir una conexión de base de datos, aquí entrará la base que se creó anteriormente, se selecciona la conexión junto con las tablas a utilizar. O bien se creará una nueva conexión con la base de datos, con los siguientes datos:

name: MySQL/JConnector

host: localhost

port:3306

database: TOOLBDH

User Name: root

Password: nulo

jdbc:mysql://localhost:3306/TOOLBDH

Si se obtiene algún error, habrá que modificar la tabla recién creada, pues netbeans requiere de al menos una llave primaria para poder utilizar la base de datos, es por esto que se ejecuta la siguiente sentencia desde la línea de comandos MySQL:

alter table ingreso modify nombreProyecto varchar(50) unique primary key;

Realizados estos cambios, elegimos todas las columnas posibles para incluir y por último finalizamos el asistente de netbeans para la conexión con la base de datos. Ahora nos aparecerá una ventana ya con campos y botones configurados, así como una tabla en la que aparecerán los datos de la Base de Datos (Fig. 4.2).

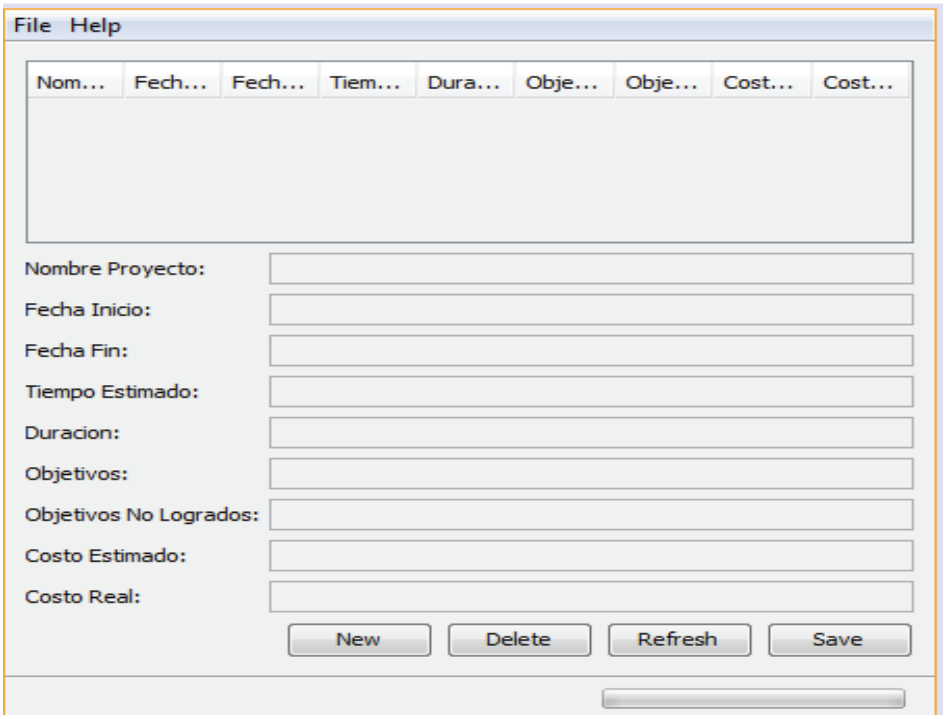


Fig.4.2 Ventana automatizada para bases de datos

4.6.1 Diseño del contenedor

Para la introducción de los datos para el proyecto, primero necesitaremos definir algunos de los campos a utilizar:

Una vez que aparece la página con las opciones automáticas, se necesita ir cambiando los campos de texto según se requiera. Para empezar ajustaremos el tamaño de los campos de texto, arrastrando cada uno de los campos de texto o las etiquetas, aumentándolos o disminuyéndolos según se necesite:

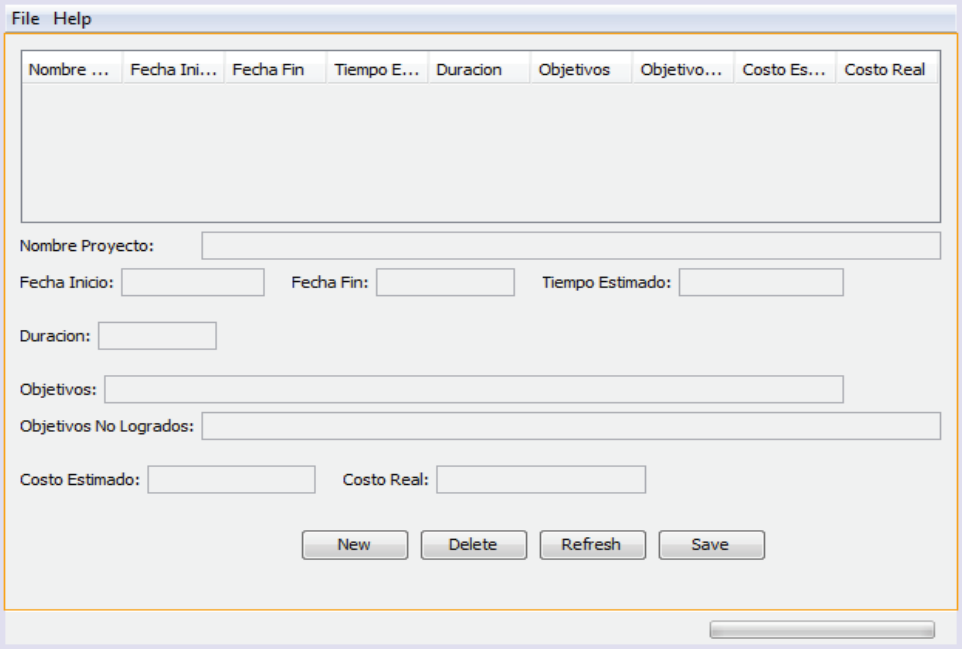


Fig. 4.3 Ventana modificada

Ahora cambiaremos el campo de texto de "Objetivos" por un campo del tipo área de texto, para que de esta manera se vea en su totalidad la descripción de los objetivos del proyecto. Para realizar este cambio lo que se necesita hacer es tomar desde la paleta de herramientas la opción de textArea, la seleccionamos y arrastramos a la ventana, cuidando que no salga demasiado de la ventana o que no modifique la posición de cada campo. Ahora, damos, click sobre el textField previo que se encontraba, buscamos en sus propiedades la etiqueta de binding, en esta copiamos las propiedades que tenga el textfield (text y enabled) y las pegamos en esas mismas propiedades pero ahora en el textArea. Una vez que la copia de estas propiedades esta lista, ya podemos borrar el textfield previo para “Objetivos”. Hacemos lo mismo para el textfield de “Objetivos no logrados”, para cambiarlos también al área de texto (Fig.4.4).

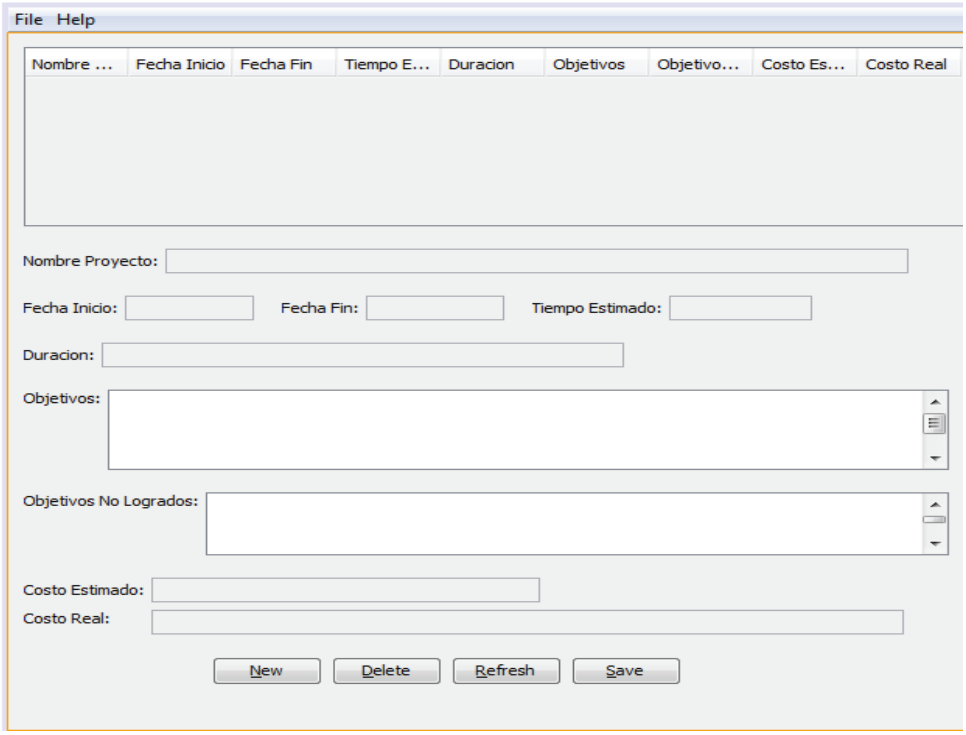


Fig. 4.4 Cambios de jTextField por jTextArea

4.6.2 Calendario

Por lo pronto conviene dejar los botones y la tabla tal cual se encuentran. Ahora se realizará otro cambio a la ventana generada automáticamente, este cambio consiste en utilizar un calendario que haga a nuestra aplicación tener un estilo más profesional, además de evitarnos el desarrollo de un calendario y hacer las validaciones correspondientes a las fechas, para utilizar este calendario es necesario hacer uso de los recursos para descargar un archivo desde la página: http://www.toedter.com/en/jcalendar/index.html, o bien, utilizar nuestro buscador favorito y teclear ”jcalendar” y descargar el archivo jcalendar -1.3.3.zip, en el cual se encuentra los archivos que utilizaremos. Descomprimimos los archivos

que se encuentran en la carpeta lib, esta carpeta la debemos tener bien localizada, pues será la que utilizaremos más adelante.

Para utilizar este paquete es necesario ir a la paleta de herramientas en netbeans, dar click derecho sobre esta y elegir la opción 'palette manager…' posteriormente en 'Add from jar…'

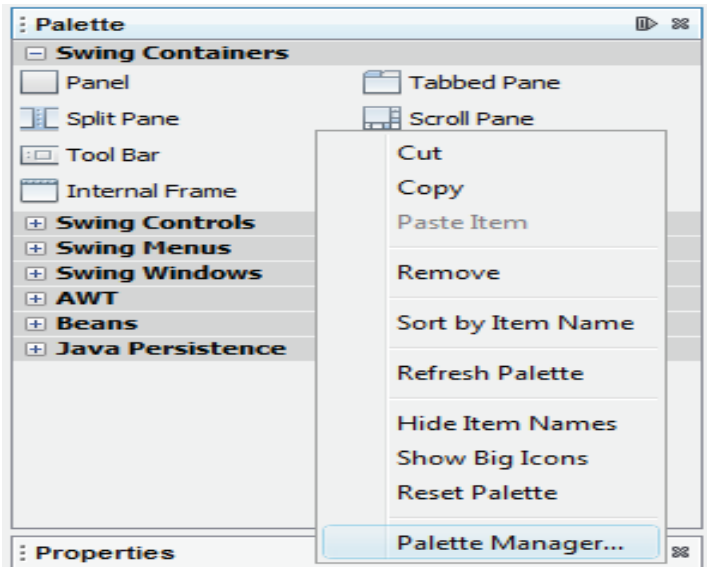


Fig. 4.5 Administración de componentes

Aquí localizaremos el archivo descomprimido 'jcalendar-1.3.3.jar', seleccionamos todas las opciones que nos aparezcan, elegimos la carpeta de la paleta en la que aparecerán las nuevas herramientas y finalizamos el proceso para poder utilizar el calendario.

Ahora tendremos que introducir los calendarios en lugar de los campos de texto "fecha inicio" y "fecha fin".

Simplemente arrastramos el componente jDateChooser, uno de los que acabamos de introducir, a nuestra ventana, lo ajustaremos a las medidas deseadas para la

ventana. Los campos de texto de las fechas no se eliminarán, simplemente los haremos más chicos y apenas visibles, ya dentro del código los haremos invisibles, para esto damos clic en la opción source dentro del marco de nuestro proyecto (Fig.4.6), localizamos la línea en la que aparece initcomponents(); y justo debajo de ella colocaremos las dos líneas que harán nuestros campos de texto invisible, estas son:

fechaFinField.setVisible(false);

fechaInicioField.setVisible(false);

```
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import org.jdesktop.beansbinding.AbstractBindingListener;
import org.jdesktop.beansbinding.Binding;
import org.jdesktop.beansbinding.PropertyStateEvent;

/**
 * The application's main frame.
 */
public class ToolBDHView extends FrameView {

    public ToolBDHView(SingleFrameApplication app) {
        super(app);

        initComponents();
        fechaInicioField.setVisible(false);
        fechaFinField.setVisible(false);

        // status bar initialization - message timeout, idle icon and busy animation, etc
        ResourceMap resourceMap = getResourceMap();
        int messageTimeout = resourceMap.getInteger("StatusBar.messageTimeout");
    messageTimer = new Timer(messageTimeout, new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                statusMessageLabel.setText("");
            }
        });
    messageTimer.setRepeats(false);
        int busyAnimationRate = resourceMap.getInteger("StatusBar.busyAnimationRate");
        for (int i = 0; i < busyIcons.length; i++) {
            busyIcons[i] = resourceMap.getIcon("StatusBar.busyIcons[" + i + "]");
        }
        busyIconTimer = new Timer(busyAnimationRate, new ActionListener() {
```

Fig.4.6 Fuente del programa

4.6.3 Generación de eventos

Regresamos a la parte gráfica dando clic en Design en el marco de nuestro proyecto. Ahora hay que programar los eventos que generará el calendario y que

pasarán la fecha a las cajas de texto que hemos desaparecido, para realizar tal acción, debemos dar clic derecho encima de la forma de calendario que recién agregamos, en cualquiera de las dos, pues el proceso es el mismo, a continuación elegir la opción Events->PropertyChange->propertychange y agregamos el siguiente código dependiendo del calendario sobre el que estemos trabajando:

```
try {
        SimpleDateFormat formato = new
SimpleDateFormat(this.jDateChooser1.getDateFormatString());
        String fechaRe[] = {formato.format(this.jDateChooser1.getDate())};
        fechaInicioField.setText("" + fechaRe[0]);
            } catch (Exception e) {
            }
```

Y para el calendario fechaFin, las sentencias quedan como:

```
try {
        SimpleDateFormat formato = new
SimpleDateFormat(this.jDateChooser2.getDateFormatString());
        String fechaRe[] = {formato.format(this.jDateChooser2.getDate())};
        fechaInicioField.setText("" + fechaRe[0]);
            } catch (Exception e) {
            }
```

Con esto, lo que se está logrando, es que al cambiar la fecha de los calendarios, este cambio se refleje en los campos de texto y estos a su vez en la base de datos, se tiene que realizar esto, debido a que los componentes para ocupar los calendarios no son nativos de netbeans y al ser realizados por terceros, no cumplen

con todas las propiedades necesarias para un manejo más práctico y sencillo que el aquí presentado. Ahora bien, el siguiente paso sugerido es que para el campo del tiempo estimado agreguemos 3 combo box, para ser utilizados como la fecha en la que se estime sea el final del proyecto, uno de estos será utilizados como los meses, otro como las semanas y el final como los días, el proceso para realizar estos es el siguiente:

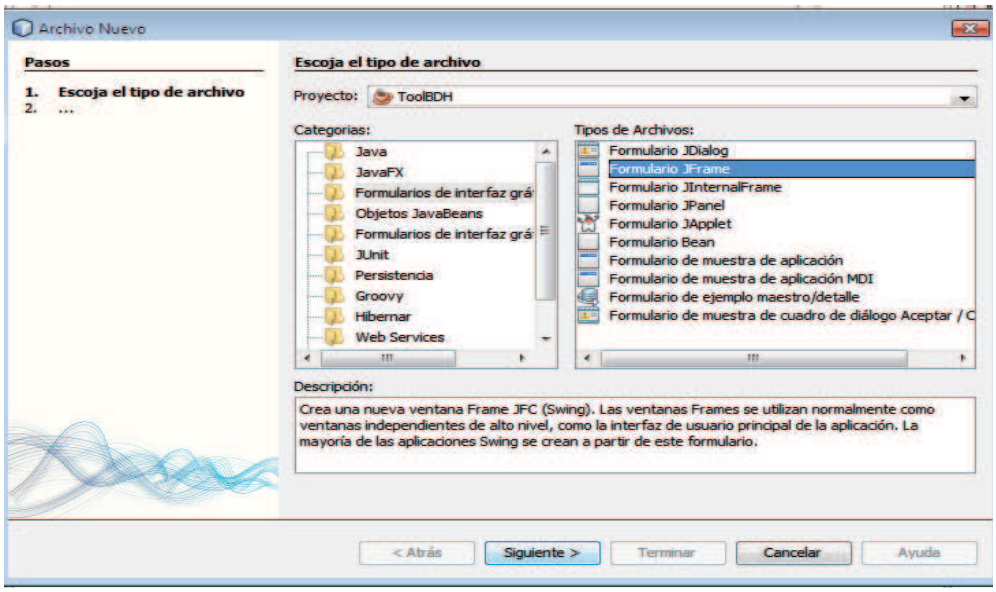
-Se toma de la paleta de herramientas un combo box y se arrastra hasta la ventana de desarrollo.
-Se cambia el nombre de la variable por otro, en este caso cbMes, lo mismo el contenido por default, este se cambia por 48,36,24,12,11,10,9,8,7,6,5,4,3,2,1,0, que será la cantidad de meses que se podrán elegir , no se agregan más, debido a que los proyectos de software por lo general no son tan largos, de la misma forma que para otros componentes, se ajusta el tamaño de este componente a uno acorde a las medidas de la ventana.
-De la paleta se toma nuevamente un combo box, se cambia el nombre de la variable por cbSemana, el contenido por default se cambia por 0,1,2,3 que será el número de semanas que se podrán elegir.
-Por último, de la paleta tomamos un jcombo box y lo ajustamos a nuestra ventana, cambiamos su contenido previo por 1,2,3,4,5,6 y 7 que serán el número de días que se podrán elegir.
-En el modo Design damos clic derecho sobre cada uno de los jcomboBox y elegimos la opción de Event->Item->ItemStateChange, NetBeans creará en automático cada uno de los eventos, y para asegurarnos que se actualice se tiene que realizar en cada uno de los eventos la misma sentencia, la cual será:

tiempoEstimadoField.setText(cbMes.getSelectedItem().toString()+" meses
"+cbSemana.getSelectedItem().toString()+" semanas
"+cbDia.getSelectedItem().toString()+" dias");

y ahora en cada ocasión en que se cambie la jcomboBox de cada uno de los elementos se determinará el nuevo tiempo estimado.

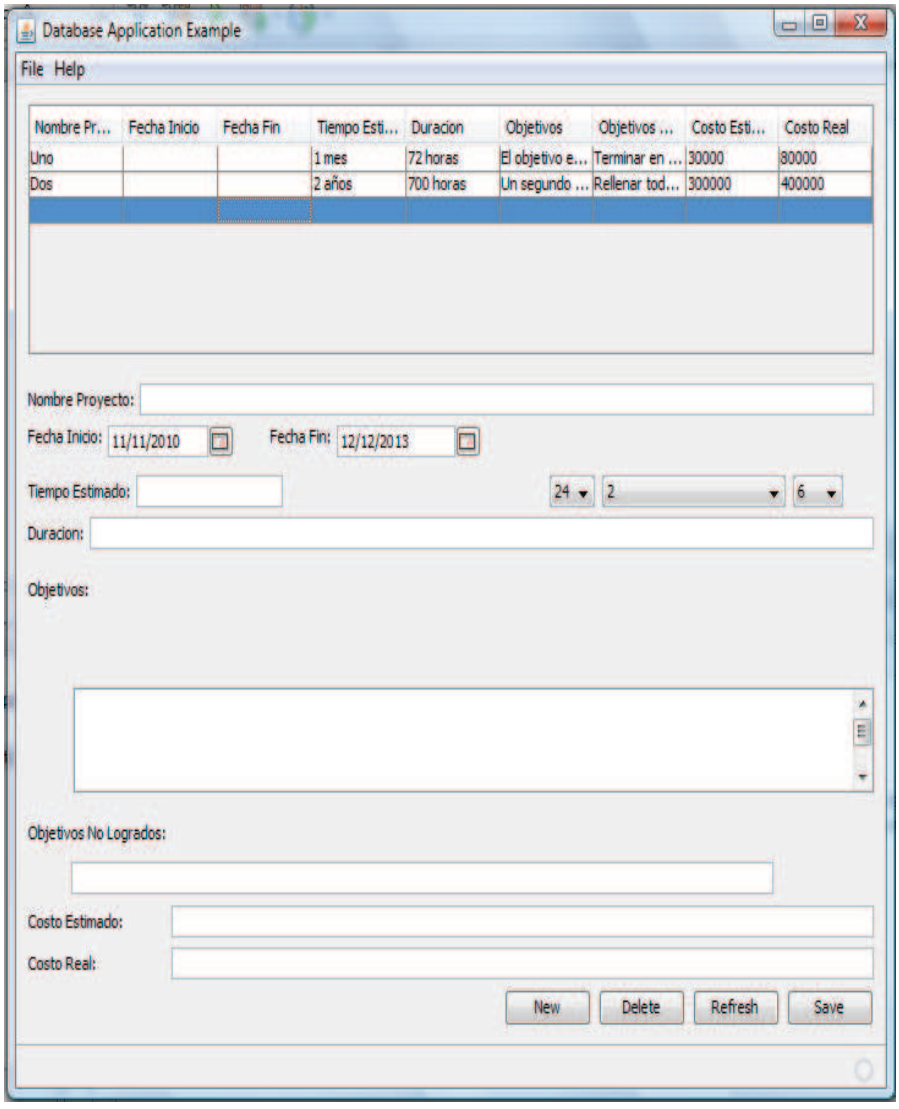
Realizados estos cambios, se verifica que no exista ningún error en el programa, compilándolo y corriéndolo dentro del mismo netbeans.

Ahora con los nuevos cambios, procedemos a crear un nuevo jFrame que será la forma en la que nuestra ventana principal se comunicará con la base de datos, lo que debemos hacer es simplemente darle click derecho a la carpeta fuente principal y seleccionar un formulario jFrame con el nombre N4 y finalizar el asistente.



4.7 Selección de un JFrame

Nos aparecerá una ventana sin ningún decorado u objeto, a esta ventana simplemente arrastramos el archivo que creamos con la base de datos a su interior y nos aparecerá ya nuestra interfaz para la base de datos lista para ser utilizada, únicamente se ajustará el tamaño del JDialog para que empate con este archivo.



4.8 Programa de pruebas

Es así como ya terminado este proyecto, se puede proceder con las pruebas que tendrán como fin el dar una vista previa de lo que se pretende que sea el proyecto final, y con el cual nos percataremos de las acciones a emprender para el trabajo definitivo.