PROYECTO DE PLANIFICACIÓN RECURSOS EMPRESARIALES – ERP

Sistema de Planeamiento – UTN

Módulo de Planeamiento y Evaluación Integral

Prototipo de Interfaz de Usuario

Versión 1.0

Historial de Revisiones

Fecha	Versión	Descripción	Autor
10/10/2011	1.0	Creación del prototipo de interfaz de usuario	Angélica López

Tabla de Contenidos

1. Introducción

En el desarrollo de sistemas y aplicaciones informáticas que aseguren su eficiencia, es necesario contar con una interfaz de usuario, sencilla y fácil de utilizar adaptada al usuario que es quien va a manipular la aplicación.

Por lo que el desarrollador está en la responsabilidad de seleccionar adecuadamente los elementos que la conformarán, mismos que sirvan como medio de comunicación entre personas y ordenadores.

Es decir el diseño gráfico aplicado a la construcción de interfaces, para conseguir un medio de interacción entre los usuarios y el conjunto de formas de la aplicación y las funciones y procedimientos que se ejecutan bajo de estas, para dar la funcionalidad al sistema.

2. Propósito

Dar a conocer a los interesados la plantilla que se usará en el desarrollo del Módulo de Planeamiento que se encuentra implementando en la UTN, así como también los archivos de configuración, el mismo que servirá de base para las aplicaciones desarrolladas dentro del sistema ERP.

3. Descripción

Este documento presenta al interesado los siguientes aspectos:

- Archivos y configuraciones necesarias para la personalización de interfaces gráficas.
- Diseño de la plantilla estándar.
- Funciones y procedimientos para la ejecución de los procesos básicos de la plantilla estándar.

Utilizando la plataforma Oracle ® 10g como servidor de base de datos, OAS (Oracle® Application Server 10.1.2) como servidor de aplicaciones y como IDE de programación Oracle® Developer Suite 10.1.2 con lenguaje de programación PL/SQL.

3.1. Archivos de Configuración

- **Visualización de Íconos**

Para visualizarlos en tiempo de ejecución haremos lo siguiente:

1. Editamos el archivo **orion-web.xml** localizado en **ORA-HOME/j2ee/ DevSuite/Application-deployments/forms/formsweb** y añadimos el directorio virtual donde se va a encontrar los iconos:

```
<virtual-directory virtual-path="/icons" real-path="C:MyAplicacion/iconos"
/>
```

2. Le indicamos ahora al servicio que extensión van a tener y en que directorio virtual se encuentran. Editamos el archivo **Registry.dat** que está en la ruta **ORA-HOME/forms/java/oracle/forms/registry** y añadimos o modificamos las siguientes líneas:

```
default.icons.iconpath=icons/
default.icons.iconextension=jpg
```

Si estamos trabajando con Developer Forms en tiempo de diseño, se puede observar que los botones icónicos aparecen en blanco aunque hayamos introducido la ruta correcta de donde se encuentran. La forma de implantarlos es la siguiente:

a) Los nombres de los archivos icónicos no deben tener el path ni la extensión, únicamente el nombre.

b) Editamos el registro de Windows y en **HKEY_LOCAL_MACHINE/Software/Oracle/HOMEo** creamos la variable **UI_ICON_EXTENSION** con valor jpg ya que estamos utilizando los iconos con esta extensión. Lógicamente debemos indicar el path de los iconos en la clave **UI_ICON** (esta clave normalmente ya está creada, si no es así debemos crearla).

Con esto tendríamos configurada la visualización de íconos.

3.2. Personalización de la Página Principal de la Aplicación

- **Configuración del Archivo formsweb.cfg**

El archivo formsweb.cfg se encuentra ubicado en el siguiente directorio:

toolsOracle\oracle\produc\10.2.0\db_2\forms90\server\

En este archivo se definen los valores de parámetro usados por el FormsServlet (f90servlet). Cualquiera de ellos se puede eliminar o modificar en las secciones de configuración nombradas.

A continuación se presenta un listado de los parámetros más importantes para la personalización de la página principal.

- **pageTitle**

Nombre del título de la página. Ejemplo.

```
# HTML page title
pageTitle=Aplicaciones UTN
```

- **width**

Especifica el ancho del applet del formulario, en pixeles. Por defecto es 650.Ejemplo.

Formsappletparameter

width=980

- **height**

Especifica el alto del applet del formulario, en pixeles. Por defecto es 500. Ejemplo.

Formsappletparameter

height=590

- **separateFrame**

Se determina si el applet aparece dentro de una ventana separada. Valores legales: Verdad o falso. Ejemplo.

Formsappletparameter

separateFrame=false

- **splashScreen**

Especifica el archivo .GIF que debe aparecer antes de que aparezca el applet. Fijar a NO para no aparecer. Dejar vacío para utilizar la imagen por defecto.

Para fijar el parámetro incluir el nombre del archivo (por ejemplo, myfile.gif) o la trayectoria virtual y nombre del archivo (por ejemplo, imágenes/myfile.gif). Ejemplo.

Formsappletparameter

splashScreen=utn2.gif

- **background**

Especifica el archivo .GIF que debe aparecer en el fondo. Fijar a NO para ningún fondo. Dejar vacío para utilizar el fondo por defecto.

Formsappletparameter

background=utn1.gif

- **lookAndFeel**

Para modificar la apariencia de la aplicación, los valores que puede tomar son:

- generic: Apariencia típica de Windows
- oracle: Apariencia por defecto definida por Oracle.

Ejemplo.

Forms applet parameter

lookAndFeel=oracle

- **colorScheme**

Es el valor del parámetro lookAndFeel es oracle en colorScheme se puede definir el siguiente conjunto de colores:

- teal
- red
- titanium
- blue
- khaki
- olive

- purple

Ejemplo.

Forms applet parameter

colorScheme=blue

- **Logo**

Especifica el archivo .GIF que debe aparecer en la barra de menú de las formas. Fijar a NO para ninguna insignia. Dejar vacío para utilizar la insignia de Oracle por defecto. Ejemplo.

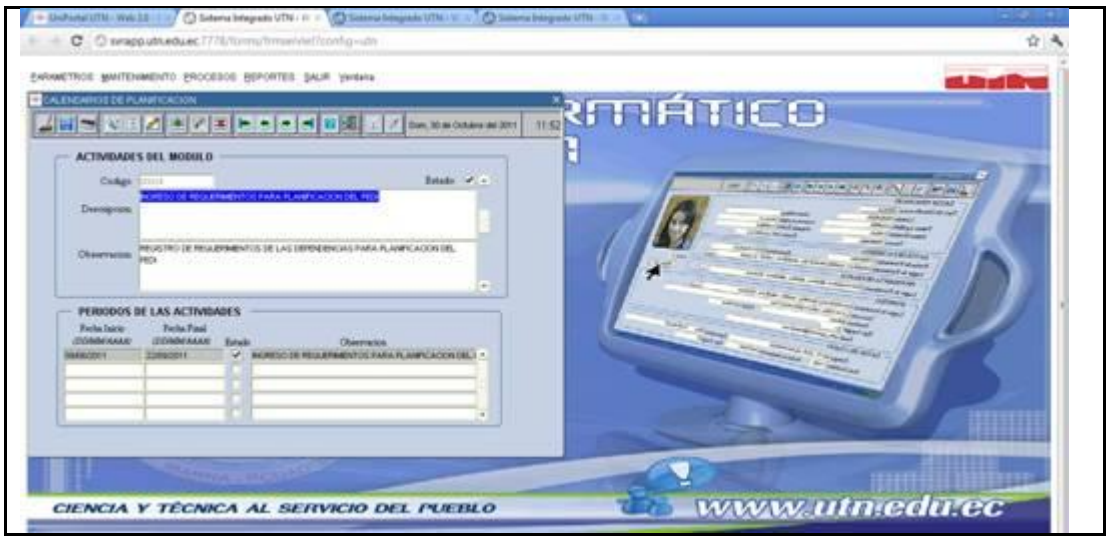
Formsappletparameter

logo=utn.gif

3.3. Diseño de la Plantilla Estándar

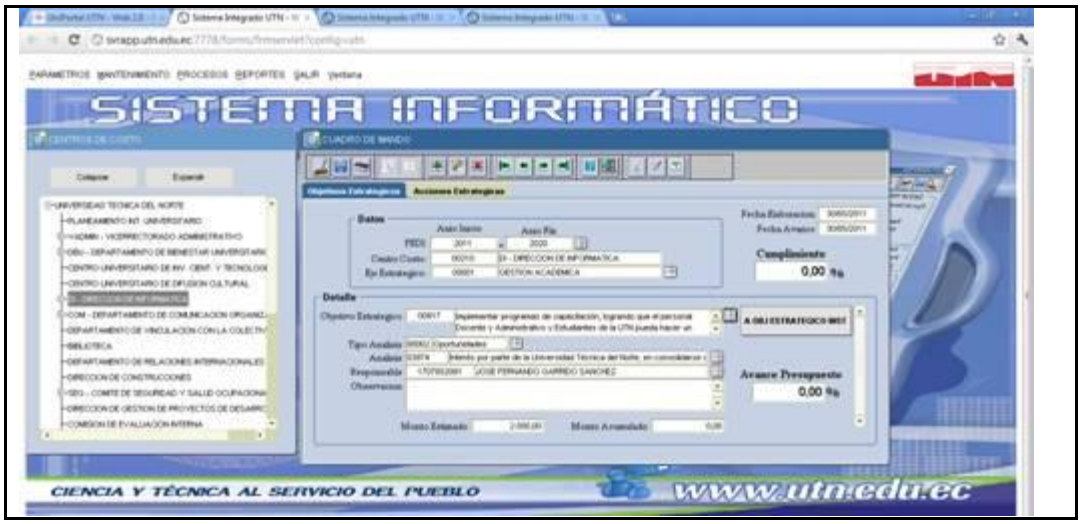
Fuente: Propia

Figura C.1. Menú de Planeamiento y Evaluación Integral



Fuente: Propia

Figura C.2. Plantilla de formulario simple



Fuente: Propia

Figura C.3. Plantilla de formulario con árbol

3.4. Funciones y Procedimientos Fijos para Establecer Atributos Visibles de a Forma a Usar

- **FUNCTION FUN_ALERTA_2BOTONES**

Esta función permite establecer una alerta con 2 botones personalizada, y retorna 1, 2 o 0.

Descripción de Variables:

pvarc2NombreAlerta Este parámetro recibe el nombre para la alerta, cuyos valores pueden ser.

ALE_ATENCION

ALE_INFORMACION

ALE_ERROR

pvarc2MensajeAlerta Este parámetro recibe el mensaje para la alerta

pvarc2Boton1Alerta Este parámetro recibe el nombre del boton1

pvarc2Boton2Alerta Este parámetro recibe el nombre del boton2

pvarc2TituloAlerta Este parámetro recibe el título de la alerta

lnumbBanderaBoton Esta variable obtiene el valor que retorna la alerta

FUNCTION FUN_ALERTA_2BOTONES

```
(
  pvarc2NombreAlerta  VARCHAR2,
  pvarc2TituloAlerta  VARCHAR2,
  pvarc2MensajeAlerta VARCHAR2,
  pvarc2Boton1Alerta  VARCHAR2,
pvarc2Boton2Alerta  VARCHAR2
)
RETURN NUMBER
IS
```

```
lnumbBanderaBoton NUMBER;

BEGIN

SET_ALERT_PROPERTY (pvarc2NombreAlerta, ALERT_MESSAGE_TEXT,

 pvarc2MensajeAlerta);

  SET_ALERT_PROPERTY (pvarc2NombreAlerta, TITLE, pvarc2TituloAlerta);

  SET_ALERT_BUTTON_PROPERTY (pvarc2NombreAlerta, ALERT_BUTTON1,
LABEL,

 pvarc2Boton1Alerta);

  SET_ALERT_BUTTON_PROPERTY (pvarc2NombreAlerta, ALERT_BUTTON2,
LABEL,

 pvarc2Boton2Alerta);

lnumbBanderaBoton := SHOW_ALERT (pvarc2NombreAlerta);

  IF lnumbBanderaBoton = ALERT_BUTTON1 THEN

     RETURN 1;

  ELSIF lnumbBanderaBoton = ALERT_BUTTON2 THEN

     RETURN 2;

  ELSE

     RETURN 0;

  END IF;

END;
```

- **FUNCTION FUN_OBTENER_FECHA_LARGA**

Esta función permite obtener la fecha actual en el siguiente formato (01 DE ENERO DEL 2007) recibiendo como parámetro la fecha actual del sistema.

Descripción de Variables:

lvarc2FechaLarga Variable en la que se va concatenando la fecha larga.

lvarc2Mes Variable que almacena el número de mes.

lvarc2Anio Variable que almacena el año.

FUNCTION FUN_OBTENER_FECHA_LARGA

```
(
pdatFechaCorta DATE
)
RETURN VARCHAR2
IS
   lvarc2FechaLarga VARCHAR2 (100)          ;
  lvarc2Mes VARCHAR2(2);
  lvarc2Año VARCHAR2(4);
BEGIN
lvarc2FechaLarga := TO_CHAR (pdatFechaCorta, 'Dy') || ', ';
lvarc2FechaLarga := lvarc2FechaLarga || TO_CHAR (pdatFechaCorta,   'DD') || ' DE ';
lvarc2Mes := TO_CHAR (pdatFechaCorta, 'MM');
  IF lvarc2Mes = '01' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'ENERO ';
  ELSIF lvarc2Mes = '02' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'FEBRERO ';
  ELSIF lvarc2Mes = '03' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'MARZO ';
  ELSIF lvarc2Mes = '04' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'ABRIL ';
```

```
  ELSIF lvarc2Mes = '05' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'MAYO ';
  ELSIF lvarc2Mes = '06' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'JUNIO ';
  ELSIF lvarc2Mes = '07' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'JULIO ';
  ELSIF lvarc2Mes = '08' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'AGOSTO ';
  ELSIF lvarc2Mes = '09' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'SEPTIEMBRE ';
  ELSIF lvarc2Mes = '10' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'OCTUBRE ';
  ELSIF lvarc2Mes = '11' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'NOVIEMBRE ';
  ELSIF lvarc2Mes = '12' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'DICIEMBRE ';
      END IF;
lvarc2Año := TO_CHAR (pdatFechaCorta, 'YYYY');
IF substr (lvarc2Año, 1, 1) = '2' THEN
lvarc2FechaLarga := lvarc2FechaLarga || 'DEL ' || lvarc2Año;
  ELSE
lvarc2FechaLarga := lvarc2FechaLarga || 'DE ' || lvarc2Año;
END IF;
  RETURN lvarc2FechaLarga;
END;
```

- **FUNCTION FUN_OBTENER_HORA_ACTUAL**

Esta función permite obtener la hora actual en el siguiente formato 21:30

Descripción de Variables:

lvarc2Hora Variable que almacenan las horas, en este caso en formato de 24 horas.

lvarc2Minuto Variable que almacenan los minutos.

lvarc2HoraActual Variable que almacena la hora tal como se va a mostrar.

FUNCTION FUN_OBTENER_HORA_ACTUAL

```
RETURN VARCHAR2
IS
  lvarc2Hora VARCHAR2 (4);
  lvarc2Minuto VARCHAR2 (2);
  lvarc2HoraActual VARCHAR2(10) ;
BEGIN
lvarc2Hora := TO_CHAR (SYSDATE, 'HH24');
lvarc2HoraActual := lvarc2HoraActual || '  ' || lvarc2Hora;
lvarc2Minuto := TO_CHAR (SYSDATE, 'MI');
lvarc2HoraActual := lvarc2HoraActual || ':' || lvarc2Minuto;
RETURN lvarc2HoraActual;
END;
```

- **PROCEDURE PRO_ACCIONES_TOOLBAR**

Este procedimiento permite determinar que botón ha sido seleccionado de la barra y se le asigna una acción.

Descripción de Variables:

lvarc2NombreElemento Esta variable sirve para recuperar el nombre del elemento seleccionado en la barra.

lvarc2NombreBloqueElemento Esta variable sirve para recuperar el nombre del bloque y el elemento seleccionado en la barra.

lnumbBanderaAlertanumber Esta variable sirve para obtener el valor retornado de la alerta.

PROCEDURE PRO_ACCIONES_TOOLBAR

```
IS
  lvarc2NombreElemento VARCHAR2(30);
  lvarc2NombreBloqueElemento VARCHAR2(60);
lnumbBanderaAlerta NUMBER;
BEGIN
lvarc2NombreBloqueElemento := NAME_IN('SYSTEM.TRIGGER_ITEM');
lvarc2NombreElemento := SUBSTR(lvarc2NombreBloqueElemento, INSTR(
                lvarc2NombreBloqueElemento, '.') + 1);

IF(lvarc2NombreElemento = 'CMD_GUARDAR') THEN
lnumbBanderaAlerta := FUN_ALERTA_2BOTONES('ALE_INFORMACION'
                ,'Atención UTN','Desea Guardar Los
Cambios','Sí','No');
IF(lnumbBanderaAlerta = 1) THEN
```

```
    DO_KEY('COMMIT_FORM');

   END IF;

ELSIF(lvarc2NombreElemento = 'CMD_IMPRIMIR')THEN

   DO_KEY('PRINT');

 ELSIF (lvarc2NombreElemento = 'CMD_LIMPIAR_FORMA') THEN

   DO_KEY('CLEAR_FORM');

:BLOQ_TOOLBAR.TXT_MOSTRAR_FECHA :=

FUN_OBTENER_FECHA_LARGA(SYSDATE);

:BLOQ_TOOLBAR.TXT_MOSTRAR_HORA := FUN_OBTENER_HORA_ACTUAL();

:BLOQ_TOOLBAR.TXT_MOSTRAR_USUARIO:=get_application_property(USERNAME
);

 ELSIF (lvarc2NombreElemento = 'CMD_BUSCAR')    THEN

   IF (name_in('SYSTEM.MODE') != 'ENTER-QUERY') THEN

        DO_KEY('ENTER_QUERY');

   ELSE

        DO_KEY('EXECUTE_QUERY');

   END IF;

elsif (lvarc2NombreElemento = 'CMD_INSERTAR_REGISTRO') THEN

CREATE_RECORD;

elsif (lvarc2NombreElemento = 'CMD_BORRAR_REGISTRO') THEN

lnumbBanderaAlerta :=
FUN_ALERTA_2BOTONES('ALE_ATENCION','AtenciónUTN','Desea Eliminar El
Cliente','Aceptar','Cancelar');

IF(lnumbBanderaAlerta = 1) THEN

        DELETE_RECORD;

   END IF;
```

```
elsif (lvarc2NombreElemento = 'CMD_LIMPIAR_REGISTRO') THEN
        CLEAR_RECORD;
elsif (lvarc2NombreElemento = 'CMD_PRIMER_REGISTRO') THEN
        FIRST_RECORD;
elsif (lvarc2NombreElemento = 'CMD_SIGUIENTE_REGISTRO') THEN
        NEXT_RECORD;
elsif (lvarc2NombreElemento = 'CMD_ANTERIOR_REGISTRO') THEN
        PREVIOUS_RECORD;
elsif (lvarc2NombreElemento = 'CMD_ULTIMO_REGISTRO') THEN
        LAST_RECORD;
elsif (lvarc2NombreElemento = 'CMD_LISTAR') THEN
        DO_KEY('LIST_VALUES');
elsif (lvarc2NombreElemento = 'CMD_EDITAR')         THEN
        DO_KEY('EDIT_FIELD');
elsif (lvarc2NombreElemento = 'CMD_AYUDA') THEN
        show_keys;
elsif (lvarc2NombreElemento = 'CMD_SALIR') THEN
lnumbBanderaAlerta := FUN_ALERTA_2BOTONES('ALE_ATENCION','Atención
UTN','Desea Salir De La Aplicación','Sí','No');
IF(lnumbBanderaAlerta = 1) THEN
    DO_KEY ('exit_form');
  END IF;
 END IF;
END;
```

- **PROCEDURE PRO_INFORMACION_OBJETO**

Este procedimiento permite obtener información de un objeto al pasar el mouse sobre él.

Descripción de Variables:

varc2Objeto Parámetro que almacena nombre de un objeto

varc2Informacion Parámetro que almacena la información que va a aparecer al pasar el mouse.

```
PROCEDURE PRO_INFORMACION_OBJETO (
pvarc2Objeto VARCHAR2,
pvarc2Informacion VARCHAR2
)IS
BEGIN
SET_ITEM_PROPERTY(pvarc2Objeto, TOOLTIP_TEXT,  pvarc2Informacion);
SET_ITEM_PROPERTY(pvarc2Objeto, TOOLTIP_FONT_SIZE, 800);
SET_ITEM_PROPERTY(pvarc2Objeto,TOOLTIP_FOREGROUND_COLOR, '
r0g50b0')
SET_ITEM_PROPERTY(pvarc2Objeto, BACKGROUND_COLOR,
'r180g220b180');
END;
```

- **PROCEDURE PRO_INFORMACION_TOOLBAR**

Este procedimiento permite obtener información de cada uno de los objetos de la barra de herramientas al pasar el mouse.

```
PROCEDURE PRO_INFORMACION_TOOLBAR
IS
```

```
BEGIN

:BLOQ_TOOLBAR.TXT_MOSTRAR_FECHA :=
FUN_OBTENER_FECHA_LARGA(SYSDATE);

:BLOQ_TOOLBAR.TXT_MOSTRAR_HORA:=FUN_OBTENER_HORA_ACTUAL();

:BLOQ_TOOLBAR.TXT_MOSTRAR_USUARIO:=get_application_property(USERNAME
);

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_SALIR','Salir');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_AYUDA','Ayuda');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_BUSCAR','Buscar');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_IMPRIMIR','Imprimir');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_LIMPIAR_FORMA','Limpiar

Forma');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_LISTAR','Lista');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_EDITAR','Editar');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_GUARDAR','Guardar');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_INSERTAR_REGISTRO','Inse
rtar

Registro');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_LIMPIAR_REGISTRO','Limpi
ar

Registro');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_BORRAR_REGISTRO','Borrar

Registro');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_SIGUIENTE_REGISTRO','Re
gistro

Siguiente');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_ANTERIOR_REGISTRO','Reg
istro
```

```
Anterior');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_PRIMER_REGISTRO',' Primer

 Registro');

PRO_INFORMACION_OBJETO('BLOQ_TOOLBAR.CMD_ULTIMO_REGISTRO','Ultimo

 Registro');

END;
```

- **PROCEDURE PRO_ALERTA**

Este procedimiento permite establecer una alerta personalizada.

Descripción de Variables:

pvarc2NombreAlerta Este parámetro recibe el nombre para la alerta, cuyos valores pueden ser.

ALE_ATENCION

ALE_INFORMACION

ALE_ERROR

pvarc2MensajeAlerta Este parámetro recibe el mensaje para la alerta

pvarc2TituloAlerta Este parámetro recibe el título de la alerta

lnumbBanderaBoton Esta variable obtiene el valor que retorna la alerta

```
PROCEDURE PRO_ALERTA

(

  pvarc2NombreAlerta  VARCHAR2,

pvarc2TituloAlerta  VARCHAR2,

  pvarc2MensajeAlerta VARCHAR2
```

```
)
IS
lnumbBanderaBoton NUMBER;
BEGIN
SET_ALERT_PROPERTY (pvarc2NombreAlerta, ALERT_MESSAGE_TEXT,
pvarc2MensajeAlerta);
SET_ALERT_PROPERTY (pvarc2NombreAlerta, TITLE, pvarc2TituloAlerta);
lnumbBanderaBoton := SHOW_ALERT (pvarc2NombreAlerta);
END;
```

- **PROCEDURE PRO_TITULO_COLOR_VENTANA**

Este procedimiento permite poner título a la ventana, además se define el color y se maximiza.

Descripción de Variables:

pvarc2NombreVentana Parámetro que recibe el nombre de la Ventana

pvarc2TituloVentana Parámetro que recibe el título de la Ventana

PROCEDURE PRO_TITULO_COLOR_VENTANA

```
(
pvarc2NombreVentana VARCHAR2,
pvarc2TituloVentana VARCHAR2
)
IS
```

```
BEGIN
PRO_VENTANA_CENTRADA('WINDOW1');
SET_WINDOW_PROPERTY('WINDOW1', WINDOW_STATE, MAXIMIZE);
 SET_WINDOW_PROPERTY (pvarc2NombreVentana, TITLE,
varc2TituloVentana || '   Form:(' ||
get_application_property(CURRENT_FORM_NAME) || ')');
SET_WINDOW_PROPERTY (pvarc2NombreVentana, BACKGROUND_COLOR
,'r200g230b210');
END;
```

- **PROCEDURE PRO_VENTANA_CENTRADA**

Este procedimiento permite centrar la ventana

Descripción de Variables:

pvarc2win	Parámetro que recibe el nombre de la ventana
lwinWinId	Variable que almacena el nombre de la ventana
lnumbWinX	Variable para la posición en x de la ventana
lnumbWinY	Variable para la posición en y de la ventana
lnumbWinW	Variable para el ancho de la ventana
lnumbWinH	Variable para el largo de la ventana
lnumbDisplayW	Variable para el ancho de la pantalla
lnumbDisplayH	Variable para el largo de la pantalla
lnumbHeightOffsett	Variable para el largo de la ventana

PROCEDURE PRO_VENTANA_CENTRADA

```
(
```

```
pvarc2Win VARCHAR2
 )
IS
lwinWinId  window;
lnumbWinX   NUMBER;
lnumbWinY   NUMBER;
lnumbWinW   NUMBER;
lnumbWinH   NUMBER;
lnumbDisplayW NUMBER;
lnumbDisplayH NUMBER;
lnumbHeightOffset NUMBER := 0;
BEGIN
  IF Get_Application_Property(USER_INTERFACE)='MSWINDOWS' THEN
lnumbHeightOffset := .05; -- inches;
  END IF;
lwinWinId := FIND_WINDOW(pvarc2Win);
  IF ID_NULL(lwinWinId) THEN
    RETURN;
  END IF;
lnumbDisplayH := TO_NUMBER(GET_APPLICATION_PROPERTY(DISPLAY_
HEIGHT));
lnumbDisplayW :=
TO_NUMBER(GET_APPLICATION_PROPERTY(DISPLAY_WIDTH));
lnumbWinX := GET_WINDOW_PROPERTY(lwinWinId, X_POS);
lnumbWinY := GET_WINDOW_PROPERTY(lwinWinId, Y_POS);
```

```
lnumbWinW := GET_WINDOW_PROPERTY(lwinWinId, WIDTH);
lnumbWinH := GET_WINDOW_PROPERTY(lwinWinId, HEIGHT);
lnumbWinH := lnumbWinH+100;
  IF ( lnumbWinW>= lnumbDisplayW ) THEN
lnumbWinX := 0;
  ELSE
lnumbWinX := (lnumbDisplayW - lnumbWinW) / 2;
  END IF;
  IF ( lnumbWinH>= lnumbDisplayH ) THEN
lnumbWinY := 0;
  ELSE
lnumbWinY := (lnumbDisplayH - lnumbHeightOffset - lnumbWinH) / 2;
  END IF;
  -- Set window's new position
  SET_WINDOW_PROPERTY(lwinWinId, X_POS, lnumbWinX-20);
  SET_WINDOW_PROPERTY(lwinWinId, Y_POS, lnumbWinY-55);
  SHOW_WINDOW(lwinWinId);
END;
```