**REPUBLICA DE NICARAGUA**

**MINISTERIO DE FOMENTO INDUSTRIA Y COMERCIO**

**PROGRAMA DE MICRO CREDITO USURA CERO**

Equipo de Sistemas de Información

**ESTANDARES DE DESARROLLO**

**Sistema de Micro Crédito Usura Cero**

**SMUSURA0**

**Versión 1.0**

Managua, Agosto 2007

**STATUS DEL DOCUMENTO**

**Estándares de Desarrollo para el Sistema de Micro Crédito Usura Cero**

**“SMUSURA0”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Número de versión** | **Incorporado por** | **Fecha** | **Descripción** |
| 1.0 | Equipo de Sistemas de Información Programa de Micro Crédito Usura Cero | 06/Agosto/2007 | Primer borrador del documento |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contenido

[1. Introducción 2](#_Toc174156406)

[2. Ambiente de Desarrollo 3](#_Toc174156407)

[3. Convenciones Generales 4](#_Toc174156408)

[4. Convenciones de Nomenclatura 6](#_Toc174156409)

[4.1. Convenciones para el Nombre de los módulos 6](#_Toc174156410)

[4.2. Convenciones para el Nombre de los Objetos de la Base de Datos 6](#_Toc174156411)

[4.3. Convenciones para el Nombre de los Controles 7](#_Toc174156412)

[4.4. Convenciones para el Nombre de Objetos del Proyecto VB 8](#_Toc174156413)

[4.5. Convenciones para el Nombre de campos del Proyecto VB 8](#_Toc174156414)

[4.6. Convenciones para el Nombre de Variables 9](#_Toc174156415)

[4.7. Convenciones para el Nombre de Clases 9](#_Toc174156416)

[4.8. Convenciones para el Nombre de Propiedades 10](#_Toc174156417)

[4.9. Convenciones para el Nombre de Métodos y Funciones 10](#_Toc174156418)

[4.10. Convenciones para el Nombre de Tipos de Enumeración 10](#_Toc174156419)

[4.11. Uso y Nomenclatura para Variables Globales 10](#_Toc174156420)

[4.12. Convenciones de Nomenclatura para Constantes 10](#_Toc174156421)

[5. Comentarios 11](#_Toc174156422)

[6. Uso del GUI (Interfaz Gráfica del Usuario) 12](#_Toc174156423)

[7. Consejos para el Uso de la Arquitectura 13](#_Toc174156424)

[8. Manejo de la Seguridad 16](#_Toc174156425)

[9. Variables Globales de Sistema 17](#_Toc174156426)

[10. Carga de Formularios Principales 18](#_Toc174156427)

[11. Normas Técnicas definidas para los Reportes 19](#_Toc174156428)

[12. Manejo de Filtros y Rangos en Consultas y Reportes 21](#_Toc174156429)

# Introducción

Este documento contiene los estándares de codificación y otras normas establecidas para el desarrollo del Sistema de Información del Programa Usura Cero (SMUSURA0) del Programa de Micro Crédito Usura Cero del Ministerio de Fomento, Industria y Comercio (MIFIC).

El uso de estos estándares será efectivo si desde un inicio todos los desarrolladores que participan en el proyecto hacen uso de ellos. Esto influirá en producir un código fuente que refleje un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. Otra ventaja a largo plazo será la facilidad que tendrán los sistemas para la incorporación de cambios entre ellas: funcionalidad nueva, depuración de errores, mejoras de rendimiento, entre otras. Aunque la legibilidad y el mantenimiento son el resultado de muchos factores, una faceta del desarrollo de software en la que todos los programadores influyen de manera especial es en la técnica de codificación y demás normas de desarrollo.

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento.

# Ambiente de Desarrollo

* Sistema Operativo: Windows XP Professional SP 2
* Motor de Base de Datos: SQL Server 2005
* Lenguaje de Programación: Visual Basic .NET 2005
* Controles a utilizar

| **Herramientas a utilizar** | **Nombre del Control** |
| --- | --- |
| Visual Basic .NET 2005 | Button  TextBox  CheckBox  GroupBox  PictureBox  Label  ProgressBar  RadioButton  StatusStrip  ToolStrip  TreeView  SaveFileDialog  OpenFileDialog |
| Component One para NET | C1Combo  C1TrueDBGrid  C1DateEdit  C1DockingTab  C1List  C1MainMenu  C1NumericEdit  C1Sizer  C1TextBox |
| Infragistics | UltraListBar |

# Convenciones Generales

* Para conservar recursos sea muy selectivo en la elección del tipo de dato, asegúrese que el tamaño de una variable no sea excesivamente grande. Por ejemplo en los ciclos for es mejor, en la mayoría de veces utilizar un tipo de dato tipo short que integer.
* Mantenga el tiempo de vida de las variables tan corto como sea posible, esto es muy importante por ejemplo cuando se utiliza un recurso finito como una conexión a una Base de Datos.
* Mantenga el scope (ámbito) de las variables tan corto como sea posible, esto sirve para evitar confusiones, mejora el futuro mantenimiento de la aplicación y además minimiza la dependencia, es decir si por algún motivo se comete el error de borrar una variable es más fácil de encontrar el error si esta tiene un ámbito más pequeño.
* Dentro de una clase, evite el uso de variables públicas, en cambio utilice procedimientos y propiedades que accedan a dichas variables (privadas), así provee una capa de encapsulación y brinda la posibilidad de validar valores de cambio sobre las mismas, antes de manipularlas directamente.
* Use siempre rutinas de manejo de excepciones.
* Use los procedimientos y variables con un solo propósito. Evite crear procedimientos multipropósito que lleven a cabo una variedad de funciones no relacionadas.
* No utilice números o cadenas literales, como por ejemplo for i = 1 to 7. En su lugar, emplee constantes con nombre, del tipo for i = 1 to Enumeración.length para que resulten fáciles de mantener y comprender.
* Use siempre sentencias Select-Case en lugar de utilizar sentencias if-then repetitivas.
* Evite albergar múltiples clases en un solo archivo.
* Establezca un tamaño estándar de sangría (por ejemplo, cuatro espacios o una tabulación) y úselo siempre. Alinee las secciones de código mediante la sangría predeterminada.
* Establezca una longitud de línea máxima para el código.
* Siempre que sea posible, no coloque más de una instrucción por línea, a excepción de los bucles.
* Cuando escriba instrucciones SQL utilice mayúsculas para las palabras clave: SELECT, UPDATE, WHERE, FROM, etc.
* Coloque las cláusulas SQL principales en líneas separadas, de modo que las instrucciones sean más fáciles de leer y editar:

SELECT Nombre, Apellido

FROM Clientes

WHERE Codigo = 70;

* No compare variables tipo string (cadenas) con “” o con cadena. empty, use la propiedad length de la cadena:

If cadena.length = 0 then

* Evite llamadas a métodos dentro de sentencias condicionales.
* Evite utilizar recursividad, use mejor ciclos for anidados.
* Use el operador condicional ternario solo en casos triviales, cuando sean casos complejos evítelo.
* Evite comparar variables booleanas contra false o trae, en vez de eso utilice el operador de negación:

Ejemplo:

If puedeEliminarse = True then if puedeEliminarse = False then

Mejor use

If puedeEliminarse then if !puedeEliminarse then

* Evite anidar bloque try … catch.
* Ordene la captura de excepciones (match) siempre en orden descendente desde la más particular hasta la más genérica.
* Use los bloques Finally solo para realizar funciones de limpieza y liberación de recursos.
* Siempre que sea posible prefiera la validación al manejo de excepciones:

Try

Conexión.close()

catch ex.exception

…Manejo de la excepcion

End Try

Cambielo por:

If Conexión.state() != Conecction.state.closed then

Con.close()

End If

# Convenciones de Nomenclatura

* Los nombres de todas las estructuras de código deben ser en español.
* Un nombre debe expresar el “qué” y no el “cómo”. Al evitar referirse a la implementación se estará conservando la abstracción de la estructura ya que la implementación está sujeta a cambios, de esta manera se describe lo que hace la estructura y no como lo hace. Por ejemplo es más claro nombrar un procedimiento de acceso a datos SeleccionarRegistro() que RealizarConsultaSelect(), porque lo que importa (para que otra persona entienda el código) es qué se supone que hace el método y no cómo lo hace.
* Es conveniente utilizar nombres largos para que se entienda para qué son, pero que no den información irrelevante. Por ejemplo es mejor SeleccionarComida() que SeleccionarComidaDelMenu().

## Convenciones para el Nombre de los módulos

Cada módulo será identificado por un prefijo formado por 2 caracteres, de manera que cualquier formulario, reporte, módulo, tabla, vista, procedimiento almacenado o componente sea fácilmente relacionado con el módulo al que pertenece.

| **Prefijo** | **Descripción del SubSistema** |
| --- | --- |
| Cl | Módulo de Control de Clientes |
| Se | Módulo de Seguridad| |
| Cc | Módulo de Cartera y Cobro |
| Tb | Módulo de Tablas Básicas |

## Convenciones para el Nombre de los Objetos de la Base de Datos

* Cada objeto de la Base de Datos lleva un prefijo que identifica el tipo de objeto (vista, procedimiento almacenado, función, etc.) a excepción de la tabla que no lleva prefijo, seguido por el prefijo del módulo al que pertenece y finaliza con el nombre del objeto.

**Sintaxis:** <prefijo del objeto><prefijo del módulo><Nombre del Objeto>

**Ejemplo (vista):** vwClCliente

| **Prefijo** | **Objeto** |
| --- | --- |
| vw | Vista (View) |
| sp | Procedimiento Almacenado (stored procedure) |
| fn | Función (function) |

*Nota*: Un ejemplo de nombre de tabla es *ClCliente*, en este caso nos podemos dar cuenta que el objeto tabla no tiene un prefijo que lo identifique.

* Los nombres de las tablas deben ser en singular. Por ejemplo: ClCliente, en vez de ClClientes.
* En los nombres de columnas en las tablas, no repita el nombre de la tabla. Por ejemplo, evite un campo llamado ClienteApellido de una tabla llamada Cliente.
* No incorpore el tipo de datos en el nombre de una columna.
* Los nombre de los campos o columnas deben empezar con Mayúscula.
* Para nombrar el Primary key y los foreign key tenemos las siguientes convenciones:
* **Primary key (Llave primaria)**

Su nombre estará formado por:

**Sintaxis:**

<nombre de la tabla>ID

Ejemplo:

Si la tabla se llama ClCliente, el primary key se llamaría ClClienteID

* **Foreign key (Llave foránea)**

Su nombre corresponderá al nombre asignado como llave primaria a la cual hace referencia.

## Convenciones para el Nombre de los Controles

Cada control que se utilice en un formulario deberá nombrarse partiendo de alguno de los prefijos presentados en la tabla, seguido de un nombre descriptivo que vaya acorde a la funcionalidad que tendrá.

**Sintaxis:** <prefijo><Nombre del Control>

Ejemplo (Control Button): btnAgregar

| **Prefijo** | **Nombre del Control** |
| --- | --- |
| **Visual Basic .NET 2005** | |
| btn | Button |
| txt | TextBox |
| chk | CheckBox |
| grb | GroupBox |
| hlp | HelpProvider |
| pic | PictureBox |
| lbl | Label |
| prg | ProgressBar |
| rad | RadioButton |
| sta | StatusStrip |
| tb | ToolStrip |
| drv | FolderBrowserDialog |
| fil | OpenFileDialog |
| sav | SaveFileDialog |
| tre | TreeView |
| **Component One para NET** | |
| cbo | C1Combo |
| cde | C1DateEdit |
| tab | C1DockingTab |
| lst | C1List |
| mnu | C1MainMenu |
| cne | C1NumericEdit |
| csi | C1Sizer |
| ctb | C1TextBox |
| grd | C1TrueDBGrid |
| **Infragistics** | |
| ulb | UltraListBar |

## Convenciones para el Nombre de Objetos del Proyecto VB

El nombre del Objeto (Ítem) deberá llevar el prefijo que lo identifique, seguido por el prefijo del módulo al que pertenece y finalizará con un nombre descriptivo de lo que representa.

**Sintaxis**:

<prefijo del Objeto><prefijo del módulo><Nombre del Objeto>

Ejemplo (Formulario): frmClPrincipal

| **Prefijo** | **Nombre del Objeto** |
| --- | --- |
| prj | Proyecto (igual que la Solución) |
| frm | Formulario |
| mod | Módulo |
| cls | Clase |
| rpt | Reporte |
| uct | User Control |
| ccl | Component Class |

## Convenciones para el Nombre de campos del Proyecto VB

El nombre de los campos deberá llevar el prefijo que lo identifique de acuerdo con su tipo de dato conforme la tabla siguiente.

**Sintaxis**:

<prefijo del tipo de dato><Nombre del campo>

Ejemplo (Formulario): sNombreCliente

| **Prefijo** | **Nombre del Objeto** |
| --- | --- |
| s | Tipo de dato string |
| n | Tipo de dato numérico |
| d | Tipo de dato fecha |
| b | Tipo de dato boolean |

## Convenciones para el Nombre de Variables

* Debe llevar el prefijo que indica el tipo de dato, seguido por un nombre descriptivo de lo que representa la variable.
* Cada palabra independiente del nombre de la Variable debe iniciar con una letra mayúscula.
* Evitar el uso de los acentos en el nombramiento de las variables.
* Es recomendado que las variables booleanas contengan una palabra que describa su estado: blnPuedeEliminarse, blnEsGrande, blnTieneHijos, etc. Y siempre se debe referir al estado verdadero: blnTieneCredito, en vez de blnNoTieneCredito.
* Toda variable debe llevar un nombre descriptivo, excepto las utilizadas para índices de ciclos. Ejemplo: i o j.
* Evite el uso de caracteres como $ o %.

**Sintaxis**:

<prefijo del Tipo de Dato><Nombre de la Variable>

Ejemplos: Dim intContador as integer

Dim strNombreCompleto as string

| **Prefijo** | **Tipo de Dato** |
| --- | --- |
| Int | Integer (entero) |
| Bln | Bolean (booleano) |
| Lng | Long (entero largo) |
| Dbl | Double (doble precisión) |
| Chr | Char (carácter) |
| Dtm | Datetime (Fecha y hora) |
| Str | String (cadena) |
| Obj | Objeto (objeto) |
| Ds | Dataset |
| Da | SqlDataAdapter |
| Cn | SqlConnection |
| Cm | sqlCommand |

## Convenciones para el Nombre de Clases

* Debe iniciar cada palabra independiente de un nombre con una letra mayúscula. Ejemplo: ManejoTransaccion
* Empiece el nombre de clase con un sustantivo. Ejemplo: AccesoInformacion
* Evitar el uso de los acentos en el nombramiento de las clases.
* Utilice las abreviaturas con moderación.
* No utilice el carácter de subrayado (\_).

## Convenciones para el Nombre de Propiedades

* Utilice un sustantivo para asignar nombres a las propiedades.
* Utilice el estilo de Mayúsculas y minúsculas Pascal (cada palabra independiente de un nombre debe iniciar con una letra mayúscula).
* No utilice la notación húngara.

## Convenciones para el Nombre de Métodos y Funciones

* Debe iniciar cada palabra independiente de un nombre con una letra mayúscula. Ejemplo: CargarDatos()
* Empiece el nombre del Método o Clase con un verbo seguido por un sustantivo. Ejemplo: RemoverInformacion()

## Convenciones para el Nombre de Tipos de Enumeración

* Debe iniciar cada palabra independiente de un nombre con una letra mayúscula.
* Utilice un nombre en singular.

## Uso y Nomenclatura para Variables Globales

* Deben ser utilizadas únicamente cuando no hay otra manera de pasar datos entre formularios, reportes, etc.
* Se declararán en un único módulo llamado “modPublico”.
* El nombre tendrá el prefijo gbl.

**Sintaxis**: gbl<Nombre de la Variable>

Ejemplo: gblCodigoCliente

## Convenciones de Nomenclatura para Constantes

* El nombre de las Constantes deberá ser declarado en Mayúscula.

Ejemplo: PORCENTAJE

# Comentarios

* Los comentarios deben ser en español.
* Evite añadir comentarios al final de una línea de código, porque lo hacen más difícil de leer. Sin embargo, los comentarios de final de línea sí son apropiados al anotar declaraciones de variables. En este caso, alinee todos los comentarios de final de línea en la misma posición de tabulación.
* Use frases completas cuando escriba comentarios. Los comentarios deben aclarar el código, no añadirle ambigüedad.
* Vaya comentando al mismo tiempo que programa, porque probablemente no tenga tiempo de hacerlo más tarde. Por otro lado, aunque tuviera oportunidad de revisar el código que ha escrito, lo que parece obvio hoy es posible que seis semanas después no lo sea.
* Evite comentarios superfluos o inapropiados, como comentarios divertidos al margen.
* Use los comentarios para explicar el propósito del código. No los use como si fueran traducciones literales.
* Para evitar problemas recurrentes, haga siempre comentarios al depurar errores y solucionar problemas de codificación.
* Cada declaración de variable importante debe incluir un comentario en la misma línea que describa el uso de la variable que se declara.
* Después de la secuencia de continuación de línea no puede escribirse un comentario en la misma línea.
* Todos los comentarios deben estar ortográficamente bien escritos, una escritura incorrecta demuestra un desarrollo negligente.
* Evite comentarios que expliquen cosas obvias, en la mayoría de las cosas el código debe ser auto explicativo. Un buen código con buenas prácticas de nombrado no debe ser comentado.

# Uso del GUI (Interfaz Gráfica del Usuario)

Este componente será utilizado para estandarizar la apariencia de cada uno de los controles que se utilizan en la aplicación y poder configurar de forma dinámica la apariencia de las ventanas. Para esto el desarrollador deberá incluir las siguientes líneas de código en el evento Form\_Load del formulario:

Dim ObjGUI As GUI.ClsGUI

ObjGUI = New GUI.ClsGUI

ObjGUI.AppPath = Application.StartupPath

ObjGUI.SetFormLayout(Me, pNombreTema)

pNombreTema es el nombre del tema (configuración de las propiedades de apariencia del conjunto de controles). Los temas definidos son: general, morado, verde, rojo, celeste. Para efectos del SMUSURA0 el tema definido es celeste.

Para el uso del componente GUI cada desarrollador deberá incluir el proyecto llamado GUI.vbproj.

# Consejos para el Uso de la Arquitectura

1. Los **Grids** deben tener la propiedad **AllowFilter = False** para que el código programado en el evento filter sea válido.
2. Se debe programar el evento **Filter** de los Grids para actualizar la condición de filtro de la clase y refrescar la cantidad de registros.

Ejemplo:

Private Sub grdOrdenes\_Filter(ByVal sender As Object, ByVal e As \_

C1.Win.C1TrueDBGrid.FilterEventArgs) Handles grdOrdenes.Filter

Try

ObjOrdenes.Orders.FilterLocal(e.Condition)

Me.grdOrdenes.Caption = " Listado de Ordenes (" + \_

Me.grdOrdenes.RowCount.ToString + " registros)"

Catch ex As Exception

Control\_Error(ex)

End Try

End Sub

1. Se debe programar el evento RowColChange de cada Grids padre para refrescar la cuenta de registros del Grids Hijo.

Ejemplo: grdOrdenes -> Grid Padre

grdDetOrdenes -> Grid Hijo

Private Sub grdOrdenes\_RowColChange(ByVal sender As Object, ByVal e As \_

C1.Win.C1TrueDBGrid.RowColChangeEventArgs) Handles grdOrdenes.RowColChange

Try

Me.grdDetOrdenes.Caption = " Detalle de la Orden (" + \_

Me.grdDetOrdenes.RowCount.ToString + " registros)"

Catch ex As Exception

Control\_Error(ex)

End Try

End Sub

1. Para establecer el MaxLength de los campos tipo cadena se recomienda utilizar el método GetColumnLength(COLUMNNAME).

Ejemplo:

Me.txtCiudad.MaxLength = ObjOrdenRow.GetColumnLenght("ShipCity")

1. Para mostrar datos en combos ó grids que no tienen datos relacionados se sugiere utilizar clases a nivel de DataTable y no de Entidad.
2. Cuando se requiera inicializar el combo en el primer elemento se recomienda utilizar el método **SelectedIndex** en lugar del Bookmark para sincronizar la clase con el control. Esto es para los casos en que la pantalla se levanta en modo Nuevo y se requiere inicializar los combos en el primer elemento.
3. Cuando se requiera inicializar el combo en cualquier elemento de la clase se recomienda utilizar el método SetCurrentById.

Ejemplo:

'Ubicar el combo en el primer registro

If ObjClientes.Count > 0 Then

ObjClientes.SetCurrentByID(IdCliente)

End If

1. Cuando se utilice la clase XDataSet y se requiere relacionar datos hay que crear la relación antes de asignar la fuente de datos al DataSource.

Ejemplo:

If XdsOrdenes.ExistRelation("OrdersConDetalle") = False Then

'Creando la relación entre el Primer Query y el Segundo

XdsOrdenes.NewRelation("OrdersConDetalle", "Orders", "OrdersDetails", "OrderID", "OrderID")

End If

XdsOrdenes.SincronizarRelaciones()

'Asignando a las fuentes de datos

Me.grdOrdenes.DataSource = XdsOrdenes("Orders").Source

Me.grdDetOrdenes.DataSource = XdsOrdenes("OrdersDetails").Source

1. Cuando se utilice la propiedad **FILTER** en los grids de las pantallas de mantenimiento principal tener en cuenta que cuando se Agrega o Edita un elemento este cumpla con las condiciones de filtro establecidas de lo contrario el elemento se agrega o edita pero los cambios no son visibles para el usuario.
2. En el caso de la programación de formularios padres con formularios Hijos que hagan uso del XDataSet para la muestra de datos, luego que se agrega o edita es necesario REFRESCAR y ubicarse en el elemento en cuestión haciendo uso del método **SetCurrentById.**

Ejemplo:

'Refrescar el Grid del Detalle de Orden

RefrescarDetalleOrden()

XdsOrdenes("OrdersDetails").SetCurrentByID("ProductID", ObjFrmEditDetOrden.IdProducto)

1. Cada vez que se requieran realizar operaciones de Inserción, Actualización o Eliminación debe utilizar una clase typada a nivel de **Datatable ó Datarow**. Para eliminar o actualizar *varios registros* se recomienda combinar los métodos (Filter y Delete) ó (Filter y Update) del DataTable. En caso que el requerimiento sea la actualización, inserción o eliminación de *un solo* registro los métodos son: Update, New Row y Update y DeleteCurrent respectivamente.

Ejemplo:

ObjOrdenes.Orders.DeleteCurrent()

# Manejo de la Seguridad

1. Referenciar el Dll SEA.dll en su proyecto .NET
2. En el evento Load del formulario al cual se le desea incorporar seguridad invocar el procedimiento SEGURIDAD().
3. Definir el procedimiento SEGURIDAD().

Ejemplo de la definición del procedimiento Seguridad:

Private Sub Seguridad()

Try

Seg.RefreshPermissions()

‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

‘Controlar la opción SELECCIONAR PERIODO

'En este caso, mnuMantPeriodo es la acción descrita en el UC y que debe estar definida además en la BD

'lblMantPeriodo es el control que se desea habilitar o deshabilitar, el mas común sería un botón de la toolbar.

‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

If Seg.HasPermission("mnuMantPeriodo") Then ‘Habilita

Me.lblMantPeriodo.Enabled = True

Me.lblMantPeriodo.Cursor = Cursors.Hand

Else '''Deshabilita

Me.lblMantPeriodo.Enabled = False

Me.lblMantPeriodo.Cursor = Cursors.Default

End If

Catch ex As Exception

Control\_Error(ex)

End Try

End Sub

# Variables Globales de Sistema

Se ha definido una estructura que se llama InfoSistema la cual contiene las variables globales del sistema como son:

1. **DBName**: Es de tipo String y almacena el nombre de la base de datos.
2. **ServerName**: Es de tipo String y almacena el nombre del servidor de base de datos.
3. **UsrName**: Es de tipo String y almacena el nombre del usuario conectado.
4. **LoginName**: Es de tipo String y almacena el Login del usuario conectado.

# Carga de Formularios Principales

Se debe evitar que una pantalla que ya está abierta se vuelva a abrir, para eso en lugar de usar el método Show se debe hacer el uso de la función **OpenForm** que está definida en el **modGeneral**. Esto es solamente en el llamado de las pantallas principales no en las que se abren como pop-up (showdialog).

Ejemplo:

Private Sub mnuCtaBco\_Click(ByVal sender As System.Object, ByVal e As C1.Win.C1Command.ClickEventArgs) Handles mnuCtaBco.Click

LlamarCuentasBancarias()

End Sub

Private Sub LlamarCuentasBancarias()

Try

Dim ObjFrmCtasBancarias As frmStbCtaBancaria

ObjFrmCtasBancarias = New frmStbCtaBancaria

'-- Poner el cursor en un estado de ocupado

Me.Cursor = System.Windows.Forms.Cursors.WaitCursor

ObjFrmCtasBancarias.MdiParent = Me

ObjFrmCtasBancarias.WindowState = FormWindowState.Maximized

**OpenForm**(ObjFrmCtasBancarias, Me)

'-- Para enviar el foco al formulario

'-- que se está llamando.

ObjFrmCtasBancarias.BringToFront()

'-- Poner el cursor en un estado de desocupado

Me.Cursor = System.Windows.Forms.Cursors.Default

Catch ex As Exception

Control\_Error(ex)

End Try

End Sub

# Normas Técnicas definidas para los Reportes

1. Los reportes deben mostrarse de forma **maximizada**, para esto es necesario que agreguen una pantalla de tipo Visualizador.
2. Las plantillas de reportes deben tener definidos los siguientes valores para los márgenes:

|  |  |
| --- | --- |
| **Márgenes del Reporte** | |
| Top Margin | 0.4 |
| Botton Margin | 1.00 |
| Left Margin | 0.4 |
| Right Margin | 0.4 |

1. Las plantillas de reporte de Orientación vertical deben diseñarse en un ancho de 7.5 pulgadas. El reporte en modo diseño debe verse de la siguiente manera:

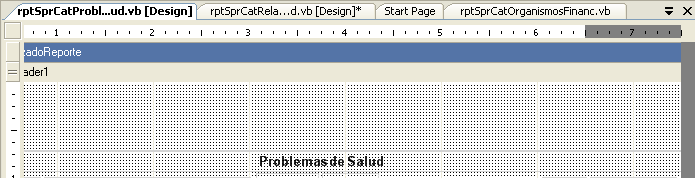


Ilustración 1.Rep\_Ancho en pulgadas de una plantilla de reporte vertical

1. Las plantillas de Reporte de Orientación Horizontal deben diseñarse en un ancho de 10 pulgadas.

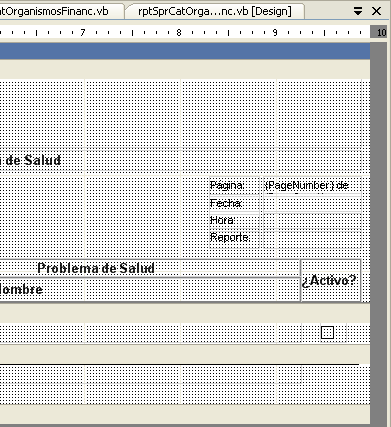


Ilustración 2.Rep\_Ancho en pulgadas de una plantilla de reporte horizontal

1. El subreporte de encabezado Vertical y Horizontal debe tener un alto de 0.7 pulgadas

# Manejo de Filtros y Rangos en Consultas y Reportes

En los filtros y rangos que se utilizan para condicionar los datos de consultas por pantalla o de reportes, se deberá de tomar en cuenta las siguientes normas en dependencia del tipo de dato del filtro:

1. Filtro de fechas: Para un campo Fecha, se muestra por defecto la fecha del servidor de base de datos.
2. Filtro de Código: Para un campo de código, se muestra por defecto la palabra “TODOS”, lo que indica que no se filtra por ningún código en particular cuando se ejecuta la consulta.
3. Rango de fechas:
   1. Para el campo Desde se mostrara por defecto la menor fecha de los registros que se consultaran. Se debe de permitir que el usuario lo cambie.
   2. Para el campo Hasta se mostrara por defecto la fecha del servidor de base de datos. Se debe de permitir que el usuario lo cambie.
4. Rango de Códigos:
   1. Para el campo Desde se mostrara por defecto el menor código de los registros que se consultaran. Se debe de permitir que el usuario lo cambie
   2. Para el campo Hasta se mostrara por defecto el mayor código de los registros que se consultaran. Se debe de permitir que el usuario lo cambie.