UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CIENCIAS

MIGRACION DE APLICACIONES DE SOFTWARE DEL SISTEMA DE INFORMACION DE DEUDA DE PEMEX A TECNOLOGIA .NET

T E S I S

QUE PARA OBTENER EL TITULO DE

LICENCIADO EN CIENCIAS DE LA COMPUTACION

P R E S E N T A :

DAVID BALL ESTRADA

DIRECTORA DE TESIS: DRA. HANNA OKTABA

FACULTAD DE CIENCIAS
UNAM

2005

ACT. MAURICIO AGUILAR GONZÁLEZ
Jefe de la División de Estudios Profesionales de la
Facultad de Ciencias
Presente

Comunicamos a usted que hemos revisado el trabajo escrito:
"Migración de aplicaciones de Software del Sistema de Información de Deuda de PEMEX a tecnología .NET"

realizado por Ball Estrada David

con número de cuenta 09831082-9, quien cubrió los créditos de la carrera de: Lic. en Ciencias de la Computación

Dicho trabajo cuenta con nuestro voto aprobatorio.

Atentamente

Director a de Tesis Propietario	Dra. Hanna Oktaba
Propietario	Ing. Enrique Mejía Velázquez
Propietario	M. en C. María Guadalupe Elena Ibargüengoitia González
Suplente	M. en C. Jorge Luis Ortega Arjona
Suplente	Act. Alejandro Talavera Rosales

Consejo Departamental de Matemáticas

Dr. Francisco Hernández Quiroz

FACULTAD DE CIENCIAS
CONSEJO DEPARTAMENTAL
DE
MATEMÁTICAS

DEDICATORIAS

A mis padres:

Gracias por el apoyo que me brindaron para poder concluir mis estudios.

A Mariana:

Gracias por creer en mí y tenerme paciencia. Por estar conmigo y animarme.

AGRADECIMIENTOS

A Dios:

Por haberme puesto los medios para realizar éste proyecto y porque gracias a sus bendiciones pude terminar con mis estudios.

A mis padres:

Por haberme brindado lo indispensable para salir adelante en mis estudios.

Al Ing. Enrique Mejía Velázquez:

Por su gran apoyo durante mi estancia en PEMEX y por sus buenos consejos.

Al Ing. Fabián García Rodríguez:

Por haber permitido mi estancia PEMEX para el desarrollo de éste proyecto, y por su apoyo brindado durante el mismo.

A Mariana:

Por su comprensión, apoyo, paciencia y por sus palabras de ánimo.

A la Dra. Hanna Oktaba

Por su apoyo brindado durante el desarrollo del proyecto.

ÍNDICE

Capítulo 3. Descripción del diseño

INTRODUCCIÓN

Desde hace aproximadamente cincuenta años, las instituciones tanto privadas como gubernamentales que manejan grandes cantidades de información se vieron en la necesidad de utilizar computadoras para facilitar su almacenamiento. Para ello comenzaron a utilizar aplicaciones de software y bases de datos. Petróleos Mexicanos no fue la excepción. Actualmente utiliza aplicaciones de software que fueron desarrolladas varios años atrás. Pero dichas aplicaciones al ser desarrolladas con tecnologías obsoletas, ya no cumplen con las necesidades que tiene el usuario. Así que se tiene la necesidad de migrar las aplicaciones utilizando nuevas tecnologías.

El objetivo del presente proyecto es construir aplicaciones nuevas, en base a las anteriores, que cumplan las necesidades actuales del usuario; para ello se utilizará la tecnología .NET de Microsoft.

En el capítulo 1 se explica cuáles son las necesidades del usuario en cuanto al manejo de información, y se hace un esbozo de cómo es que es manejada la información que está almacenada y qué cambios va sufriendo conforme es procesada. También se hace una reseña histórica de los antecedentes de software, es decir, qué tecnologías se han utilizado con el paso de los años. Principalmente se describe cada una de las aplicaciones a migrar enlistando los requerimientos del usuario.

En el capítulo 2 se hace un estudio de cómo opera la plataforma .NET de Microsoft, en qué consiste el Framework y las bibliotecas de clases. Se describe brevemente las ventajas de utilizar Visual Studio .NET. Por último se menciona cómo es el manejo de bases de datos con la tecnología ADO .NET, así como los elementos que la componen.

En el capítulo 3 se muestra cómo fueron diseñadas las aplicaciones: los diagramas de casos de uso, diagramas de clases, y una breve explicación del diseño.

En el capítulo 4 se describen las estrategias de implementación, orden de integración y qué herramientas de .NET fueron utilizadas. Asimismo se hace una descripción de la forma en que fueron probadas cada una de las clases implementadas.

CAPÍTULO I

DESCRIPCIÓN DEL PROBLEMA

1.1 Preámbulo

Actualmente, Petróleos Mexicanos (PEMEX) tiene la necesidad de pedir préstamos o financiamientos a bancos tanto nacionales como extranjeros para poder invertir en los distintos proyectos que realiza para la explotación del petróleo. La Gerencia de Financiamientos y Análisis de Mercado es el departamento que se encarga de controlar estos préstamos.

Para pedir un préstamo (llamado también desembolso), primeramente se tiene que abrir un Contrato de Financiamiento con algún banco (llamado también acreedor), y cuando el contrato se firma, ya se pueden generar desembolsos con dicho acreedor.

Cada desembolso se hace en un tipo de moneda en particular. Las deudas pueden ser de diferentes tipos, dependiendo del monto financiado, el acreedor y el tiempo a pagar. Cada préstamo tiene un plazo para pagarse y la tasa de interés puede variar dependiendo del plazo. El intervalo del interés también es variable y en cada instante del tiempo se puede calcular su interés acumulado al cual se le llama interés devengado. La deuda de cada desembolso se va cubriendo con un número diverso de pagos, esto es, con base en las diferentes tablas de amortización de cada pago. Dichas tablas contienen los pagos a realizar, su interés total y su interés devengado en función del tiempo. Los pagos son de diferentes tipos, dependiendo del tipo de deuda y del acreedor. En cualquier instante del tiempo, se requiere conocer para cada uno de los desembolsos, cuánto se ha pagado y cuánto falta por pagar, sus respectivas fechas y los tipos de pago efectuados y por efectuar. Asimismo se requiere conocer toda la información detallada respecto a cada pago. Los pagos son efectuados por los distintos organismos que conforman PEMEX bajo cierta proporción asignada a cada organismo.

Para cada cierre mensual es necesario conocer el tipo de cambio de cierre de cada una de las monedas extranjeras para poder hacer el balance mensual en moneda nacional. Y es necesario conocer toda la información a detalle de los acreedores, contratos, desembolsos, pagos, intereses devengados, etc. Actualmente PEMEX tiene más de 2000 contratos abiertos con los acreedores de todo el mundo.

Como puede observarse, todo lo anterior representa una gran cantidad de información. Es por ello que esta información se encuentra almacenada en una base de datos, y existe un gran número de aplicaciones de software que se encargan de que los usuarios puedan consultar y actualizar toda esta información.

1.1.1 Descripción del Sistema de Información de Deuda

La Base de Datos cuya información está relacionada con la deuda de PEMEX se encuentra en un sistema manejador de bases de datos Oracle en plataforma UNIX. Dicha Base de Datos es llamada "Sistema de Información de Deuda"; de aquí en adelante para simplificar, al referirnos a ella, le llamaremos Sistema.

Las distintas tablas del Sistema se alimentan de información por diversas aplicaciones de software donde se consulta, actualiza, elimina y se inserta nueva información. Dichas aplicaciones fueron desarrolladas en Oracle Developer y Power Builder. Posteriormente parte de la información es procesada por Data Base Triggers o Stored Procedures; ya que se requiere hacer distintos cálculos como intereses, montos, devengados, impuestos, etc. Esto se realiza mediante procesos que son ejecutados en el Sistema Operativo UNIX.

1.1.2 Entradas al Sistema

Como ya se mencionó, el sistema se alimenta por distintas aplicaciones de software, algunas de éstas las consideramos primarias, ya que por ellas entra la información de los desembolsos y los pagos, esto se puede observar en la figura 1.1.

Como ya se mencionó antes, la información recorre un largo camino donde experimenta diversos cambios. La figura 1.2. lo muestra de forma detallada.

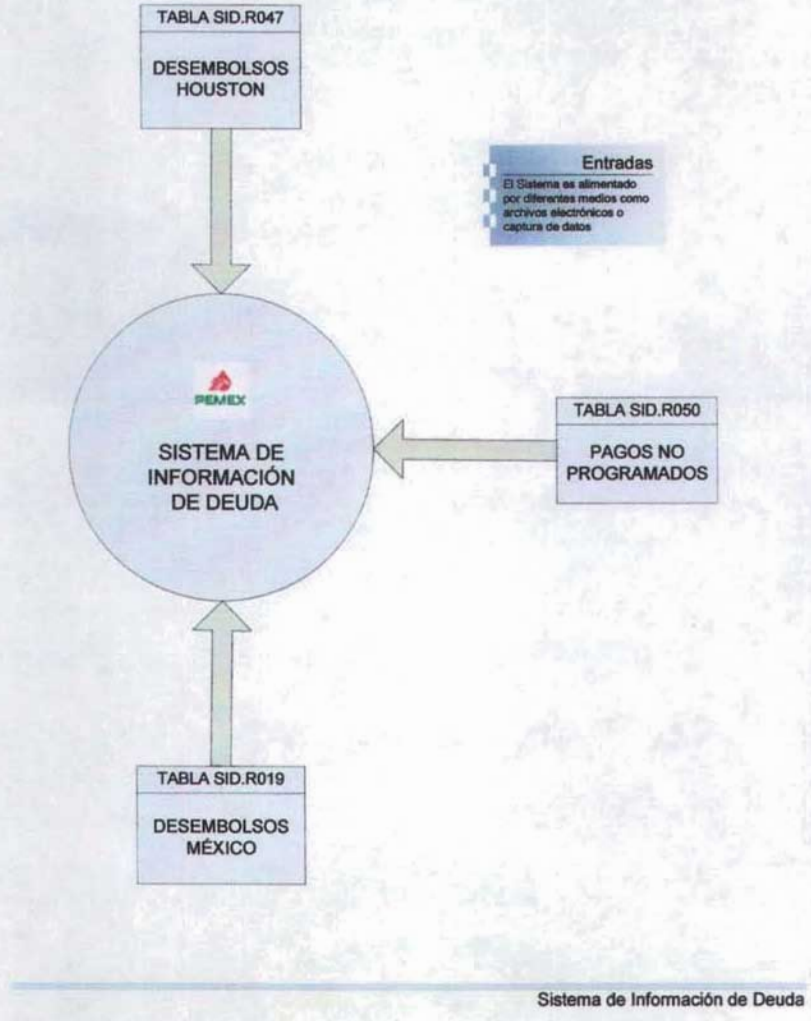


Figura 1.1 Diagrama de entradas al sistema

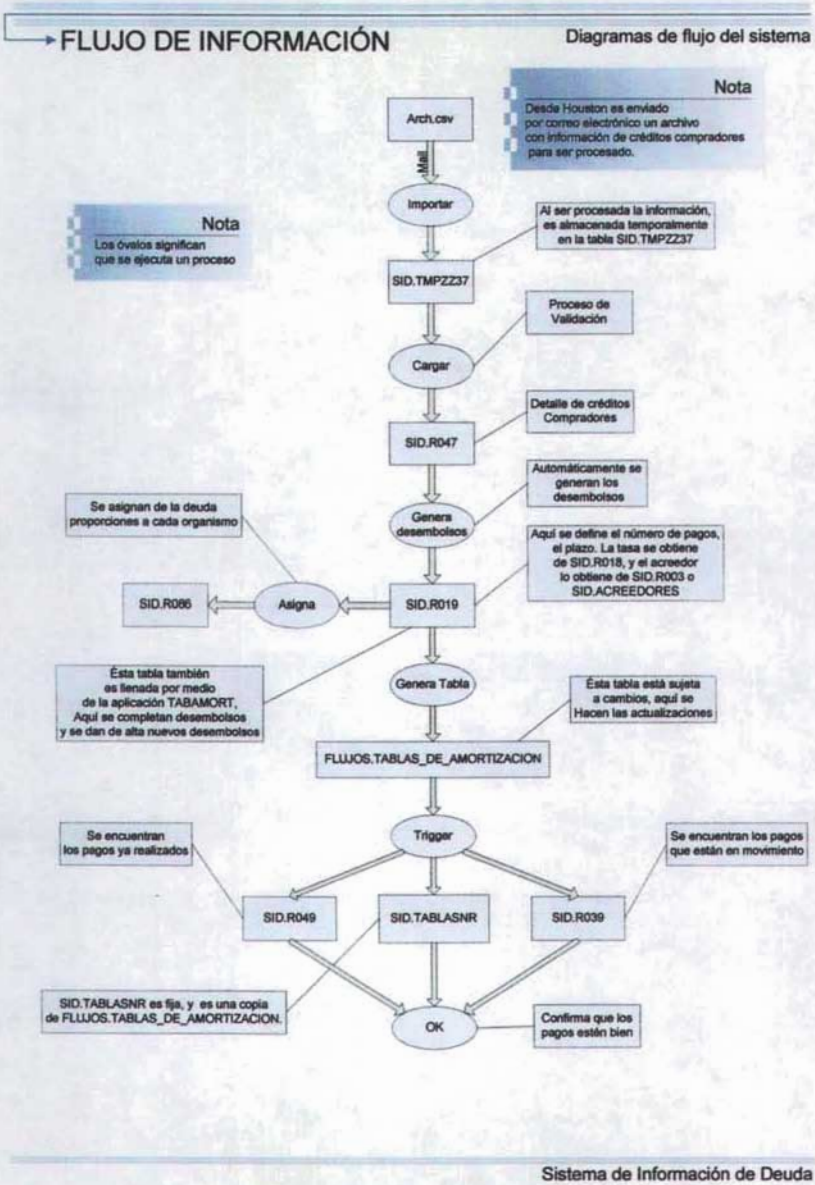


Figura 1.2 Recorrido de la información del Sistema de Información de Deuda

1.2 Definición del problema

El problema consiste en migrar las aplicaciones que se desarrollaron en Oracle Developer a la tecnología .NET de Microsoft, y como actualmente el usuario ha incrementado sus necesidades, se agregarán nuevos requerimientos a las aplicaciones en base a dichas necesidades. También esta tecnología será utilizada para desarrollar una nueva aplicación. Dado que .NET ofrece herramientas para el fácil y eficiente desarrollo de aplicaciones, a medida de lo posible, se explotarán al máximo para facilitar la programación y ofrecerle al usuario facilidad de uso, funcionalidad y rapidez en la operabilidad de las aplicaciones.
Cabe mencionar que las aplicaciones realizadas en Power Builder, por motivos de tiempo, por el momento no serán migradas.

1.3 Motivos de la migración

Para poder comprender el porqué de la migración, se hará una breve descripción de los antecedentes del Sistema de Información de Deuda de PEMEX.

1.3.1 Antecedentes de software en la Gerencia de Financiamientos y Análisis de Mercado de PEMEX.

A continuación mostraremos la cronología de cómo fue evolucionando el Sistema de Información de Deuda y las aplicaciones de software a partir de 1990:

1990: La Base de Datos se encontraba en Oracle 4.0. Surgen las primeras aplicaciones de software utilizando SQL Forms 2.0 de Oracle. Algunas aplicaciones fueron hechas en lenguaje Shell de UNIX.
1994: Se utilizó un convertidor para migrar la información de Oracle 4.0 a Oracle 7.0 y las aplicaciones de SQL Forms a Oracle Developer. Los nuevos desarrollos se hicieron en Oracle Developer.
1995: Comienza a usarse Power Builder 3.5. Se migraron las aplicaciones que estaban en Shell a esta herramienta y se hicieron desarrollos nuevos. Por falta de tiempo ya no se migraron las aplicaciones de Oracle Developer. La Base de Datos se migró a Oracle 7.3.

1996: Se migra de Power Builder 3.5 a Power Builder 6.0.

2000: Se suspende el uso de Oracle Developer, y aunque no se autorizó estandarizar Power Builder, se siguió apoyando su uso debido a su funcionalidad.

2003: Se deja de usar Power Builder para desarrollar aplicaciones, pero las aplicaciones anteriores ahí realizadas se continúan utilizando. Se comienza a utilizar Visual Basic 6.0 para desarrollos nuevos.

2004: La Gerencia de Informática de PEMEX[1] decide estandarizar Visual Basic .NET como herramienta de desarrollo de software. Se comienzan a migrar las aplicaciones desarrolladas en Oracle Developer y Visual Basic 6 usando esta nueva tecnología. Los nuevos desarrollos también se desarrollan en esta herramienta. Las aplicaciones que estaban en Power Builder por el momento se siguen usando.

La estandarización de la tecnología .NET en PEMEX se hizo gracias a su facilidad de desarrollo y a que las herramientas de Microsoft son sencillas de usar y están estandarizadas en la mayoría de las instituciones de Gobierno. Además la tecnología .NET cuenta con diversos elementos interesantes que más adelante describiremos.

Es por ello que se decidió migrar las aplicaciones que estaban en Oracle Developer y en Visual Basic 6.0 a tecnología .NET.

1.4 Descripción de las aplicaciones a migrar

A continuación, describiremos cada una de las aplicaciones a migrar, así como los requerimientos funcionales y no funcionales de cada una de ellas. Primeramente, enlistamos los requerimientos funcionales y no funcionales que son comunes a todas las aplicaciones.

1.4.1 Requerimientos funcionales

R1. Autenticarse antes de entrar

R2. Abrir cualquier aplicación desde la pantalla principal

R3. Consultar la información respectiva

R4. Enviar a una hoja de Excel la información consultada

R5. Insertar nuevos registros

R6. Borrar registros

[1] Departamento que rige a PEMEX en todo lo referente a computación.

R7. Actualizar registros ya existentes

R8. Deshacer los cambios realizados antes de guardar

R9. Guardar los cambios en la Base de Datos

R10. Salir a la pantalla principal

Cabe aclarar que únicamente para la aplicación Asignación de créditos a los organismos R5 a R9 no aplican.

1.4.2 Requerimientos no funcionales

Requerimientos de ejecución

- Las aplicaciones deberán ser ejecutadas en máquinas con sistema operativo WIN32 y plataforma .NET.

Requerimientos operacionales

- El único requerimiento es que la máquina donde se instalen las aplicaciones esté conectada al servidor donde se encuentra la Base de Datos.

Facilidad de uso

- La consulta y manipulación de datos se deberá hacer en una sola ventana de manera multiregistro.
- Cada pantalla deberá contener una barra de estado en la parte inferior donde se muestre el total de los registros y el número de registro en el cuál el usuario tiene el cursor. Dicha barra también deberá mostrar mensajes de instrucciones o sugerencias para el usuario dependiendo dónde se encuentre el cursor.
- Las tablas de la Base de Datos deben contener su información en letras mayúsculas, así que en caso de que el usuario no lo haga, las aplicaciones deben de hacer la conversión a mayúsculas de forma automática.
- Cada que el usuario cometa un error en altas, bajas o cambios, las aplicaciones deberán mostrar un diálogo con el mensaje correspondiente con una explicación clara del error.
- Los componentes gráficos de cada aplicación (botones, menús, etiquetas, etc.) deben ser claros para que el usuario fácilmente pueda realizar sus diferentes tareas.

- La consulta de datos deberá poderse hacer por diversas formas de búsqueda, dependiendo de la tabla de la Base de Datos a la que hagamos referencia.

Requerimientos de interfaz gráfica

- Habrá una pantalla principal la cual tendrá menús y barras de herramientas para acceder a cada una de las aplicaciones. Se podrán abrir más de una aplicación a la vez, y se podrá cambiar de una a otra sin necesidad de cerrarlas.
- Por motivos de estética, en la ventana de cada aplicación el usuario podrá escoger una imagen desde un archivo para colocarla en el fondo de la aplicación.

Requerimientos de seguridad

- No permitirle al usuario que haga movimientos a la Base de Datos que no sean correctos.
- Si el usuario hace alguna modificación importante, el sistema deberá preguntarle al usuario la confirmación del cambio.
- Cuando el usuario salga de alguna aplicación, se le deberá preguntar si desea guardar los cambios antes de salir.

Requerimientos legales

- Que estén al día las licencias del software a utilizar.

Tiempo de respuesta

- Que las consultas y actualizaciones a la Base de Datos se realicen de manera eficiente, para ello se recomienda usar al máximo las bibliotecas de ADO.NET.

1.4.3 Catálogo de Contratos de Financiamiento

Recordemos que para que PEMEX pueda pedir un financiamiento a un acreedor, primero necesita abrir un contrato de financiamiento. Es por ello que es necesario tener un registro completo de cada uno de los contratos de financiamiento que tiene PEMEX con los acreedores, esta información es controlada en la aplicación llamada Catálogo de Contratos de Financiamiento.

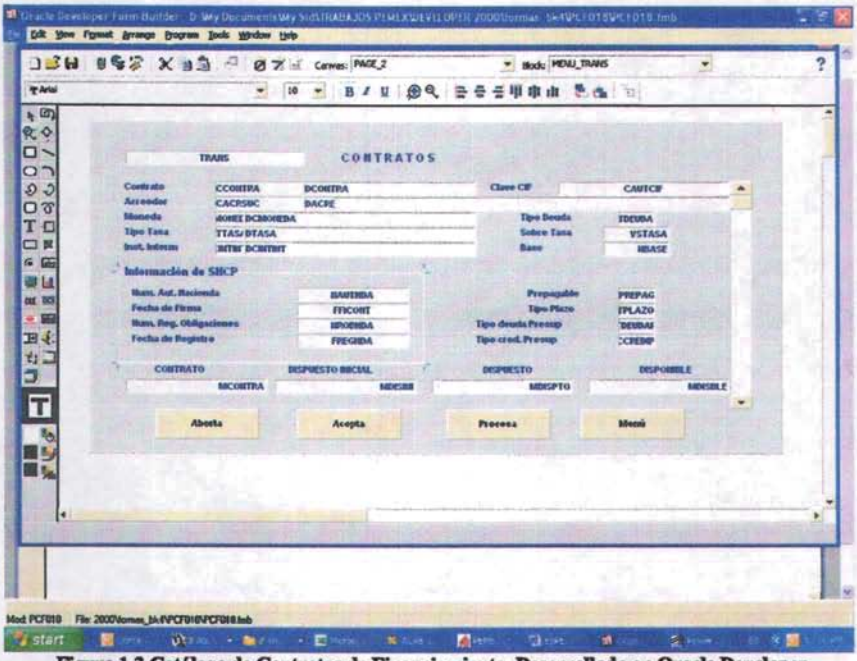


Figura 1.3 Catálogo de Contratos de Financiamiento. Desarrollado en Oracle Developer

Sus requerimientos funcionales y no funcionales son los que mencionamos para todas las aplicaciones a migrar, excepto para los requerimientos no funcionales en cuanto a seguridad.

1.4.3.1 Requerimientos no funcionales (Requerimientos de seguridad)

Como es común en las aplicaciones de software, la inserción, actualización y borrado de datos necesita ciertas validaciones. El Catálogo de Contratos de Financiamiento no es la excepción, ya que cada campo requiere cumplir ciertas reglas para poder insertar, actualizar o borrar; de no ser así habría inconsistencia en los datos, y esto en la práctica provocaría graves consecuencias por el tipo de información que se está manejando.

Para que esto no ocurra se necesitan incluir las validaciones de tal forma que el usuario únicamente tenga opción de insertar, modificar o eliminar datos de manera correcta. Y es por ello que detallaremos los requerimientos no funcionales para esta aplicación en cuanto a la seguridad (validaciones):

Validaciones generales

El usuario debe poder teclear en cada campo del catálogo el tipo de dato especificado en la tabla de la Base de Datos y a lo más el número de caracteres permitido así como cumplir con su restricción en caso de que exista, cualquier otra validación en particular de cada campo será aclarada de manera explícita.

La tabla en la Base de Datos que contiene la información del Catálogo de Contratos es SID.R018, aunque contiene campos que actualmente ya no son utilizados, a continuación se muestran los campos que se utilizarán así como sus especificaciones:

Campo	Nombre en SID.R018	Restricciones	Tipo de dato
Contrato	CCONTRA	NOT NULL	VARCHAR2(7)
Clave Cif	CAUTCIF		VARCHAR2(11)
Acreedor	CACRSUC		NUMBER(7)
Moneda	CMONEDA		VARCHAR2(1)
Tipo Deuda	TDEUDA		VARCHAR2(1)
Tipo Tasa	TTASA		VARCHAR2(2)
Sobre Tasa	VSTASA		NUMBER(7,5)
Institución Intermediaria	CINTINT		VARCHAR2(2)
Base	NBASE		NUMBER(3)
Número de autorización de Hacienda	NAUTHDA		VARCHAR2(16)
Fecha de Firma	FFICONT		VARCHAR2(8)
Número de Registro de Obligaciones	NROBHDA		VARCHAR2(10)
Fecha de Registro	FREGHDA		VARCHAR2(8)
Prepagable	IPREPAG		VARCHAR2(1)
Tipo Plazo	TPLAZO		NUMBER(7)
Tipo de deuda presupuestal	TDEUDAP		VARCHAR2(1)
Tipo de crédito presupuestal	CCREDIP		VARCHAR2(2)

La llave primaria es CCONTRA. Vale la pena aclarar que para la inserción de datos es conveniente respetar el orden anterior para la propia comodidad del usuario.

Validaciones particulares de cada campo

Los campos que no especifiquen validación particular, únicamente deberán cumplir con su especificación en la tabla.

Contrato

- Únicamente debe admitir teclear 7 caracteres, no más ni menos.
- Los dos primeros caracteres son letras mayúsculas y deben estar en la tabla SID.R007 campo CCREDI. En caso de no ser letras mayúsculas, hacer automáticamente la conversión.
- Los cinco caracteres siguientes deben ser números.
- Dado que este campo es la llave primaria, no se debe teclear una clave que ya esté registrada.

Clave Cif

- En caso que el campo contrato inicie con "AP", no podrá ser vacío.

Acreedor

- Solamente puede contener la clave de un acreedor ya registrado, es decir solamente valores del campo CACRSUC de la tabla SID.ACREEDORES. Para ello se colocará un Combo Box que contenga la lista de los nombres de los acreedores, y al seleccionar el nombre del acreedor, su clave quede insertada en el campo. Ésta forma será la única para seleccionar el acreedor, no se permitirá teclear la clave para evitar errores.
- Este campo no podrá ser vacío.

Moneda

- Contendrá únicamente la clave de una moneda ya registrada, dichas claves se localizan en el campo CMONEDA de la tabla SID.R004. Al igual que en Acreedor, la forma de seleccionar será con un Combo Box que contendrá el nombre de la moneda, y al seleccionarla llenará la celda con su clave.
- Este campo no podrá ser vacío.

Tipo Deuda

- Solamente tendrá dos opciones: "I" ó "E". Tendrá un Combo Box donde se pueda seleccionar "Interna" o "Externa". Al seleccionar "Interna" la celda se llenará con "I", y al seleccionar "Externa" con "E". En caso que el país del acreedor sea México, por omisión el campo se llenará con I. Para ello hacer la consulta correspondiente en la tabla SID.ACREEDORES.
- Este campo no podrá ser vacío.

Tipo Tasa

- Este campo tendrá un Combo Box que contendrá los tipos de tasas registradas. El campo será llenado con la clave del tipo de tasa al seleccionar el nombre. La información correspondiente se encuentra en la tabla SID.R034.
- Este campo no podrá ser vacío.

Sobre Tasa

- Este campo por omisión será llenado con 0.

Institución Intermediaria

- Sus opciones serán dos: "PU" ó "PR". Tendrá un Combo Box donde se podrá seleccionar "Pública" para "PU" y "Privada" para "PR".
- Este campo no podrá ser vacío.

Base

- Tendrá un Combo Box con cuatro opciones a seleccionar: 30, 31, 360, 365.
- Este campo no podrá ser vacío.

Fecha de Firma

- Su formato de llenado será *ddmmyyyy*, y para garantizar esto, se colocará sobre la celda un Date Time Picker, donde se podrá elegir la fecha a llenar.
- Por omisión se llenará con la fecha actual.
- Estará acotada entre tres meses atrás y tres meses adelante respecto a la fecha actual.

Número de Registro de Obligaciones

- Este campo no podrá ser vacío

Fecha de Registro

- Tendrá las mismas validaciones que el campo Fecha de Firma

Prepagable

- Contendrá un Combo Box con opciones de "SI" o "NO", al seleccionar "SI" el campo se llenará con "S" y al seleccionar "NO" con "N".

Tipo Plazo

- Tendrá un Combo Box con tres opciones: "Corto", "Mediano" o "Largo". Sus valores a llenar en la celda serán "C", "M" o "L" respectivamente.

Tipo de deuda presupuestal

- Llevará un Combo Box con las opciones: "R", "N" o "T".

Tipo de crédito presupuestal

- Tendrá un Combo Box donde las opciones a escoger serán las del campo DCREDI de la tabla SID.R007, y el valor que pondrá en la celda será el campo CCREDI.

Validaciones para bajas o cambios

Para poder eliminar un registro o cambiarle algún dato tiene que cumplir la condición de no tener desembolsos registrados. Para ello solamente se revisa que la clave del contrato no se encuentre registrada en la tabla SID.R019.

1.4.4 Catálogo Interfase Bancos Houston-México

En la ciudad de Houston Texas, PEMEX tiene oficinas corporativas donde realiza movimientos financieros. Esta información llega a México en archivos vía correo electrónico, y se requiere poder asociar algunos de estos datos a los contratos de financiamiento catalogados dentro del Sistema. Para lograrlo se utiliza el Catálogo Interfase Bancos Houston-México.

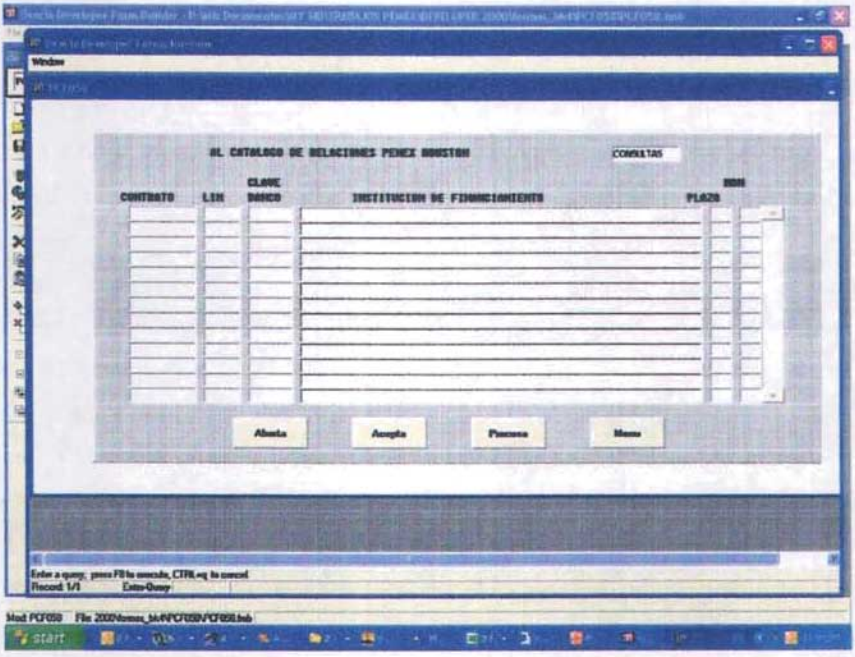


Figura 1.4 Catálogo Interfase Bancos Houston-México. Desarrollado en Oracle Developer

1.4.4.1 Requerimientos de seguridad

Validaciones generales

La tabla en la Base de Datos que contiene la información del Catálogo interfase bancos Houston-México es SID.R055. A continuación se muestran los campos que se utilizarán así como sus especificaciones:

Campo	Nombre en SID.R055	Restricciones	Tipo de dato
Contrato	CCONTRA	NOT NULL	VARCHAR2(7)
Clave del Banco	CVEBCO	NOT NULL	VARCHAR2(4)
Nombre	NOMBRE		VARCHAR2(50)
Plazo	PLAZO	NOT NULL	VARCHAR(2)
Moneda	CMONEDA	NOT NULL	VARCHAR(2)
Linea	CLINEA	NOT NULL	VARCHAR2(4)

La llave primaria son los campos: CCONTRA, CVEBCO, PLAZO, CMONEDA Y CLINEA.

Validaciones particulares de cada campo

Contrato

- Usar un Combo Box cuya fuente de datos será la lista de contratos registrados.

Clave del Banco

- Este campo solo podrá ser llenado con cuatro caracteres.

Nombre

- Este campo será de sólo lectura y se llena con el nombre del acreedor al que pertenece el contrato.

Plazo

- Tendrá un Combo Box con tres opciones: "Corto", "Mediano" o "Largo". Sus valores a llenar en la celda serán "C", "M" o "L" respectivamente.

Moneda

- Se usará un Combo Box que contendrá el nombre de la moneda, y al seleccionarla se llenará el campo con su clave.

Validaciones para bajas o cambios

Existe libertad para borrar, y actualizar. Recordemos que para insertar únicamente se debe revisar que el registro no exista.

1.4.5 Catálogo de Conceptos de Pago

Los pagos que hace PEMEX a los distintos acreedores con los que tiene contratos abiertos son de diferentes tipos (diversas comisiones, intereses, intereses devengados, diversos gastos, pagos por anticipado, primas, etc.), así que para clasificar los pagos se utiliza el Catálogo de Conceptos de Pago.

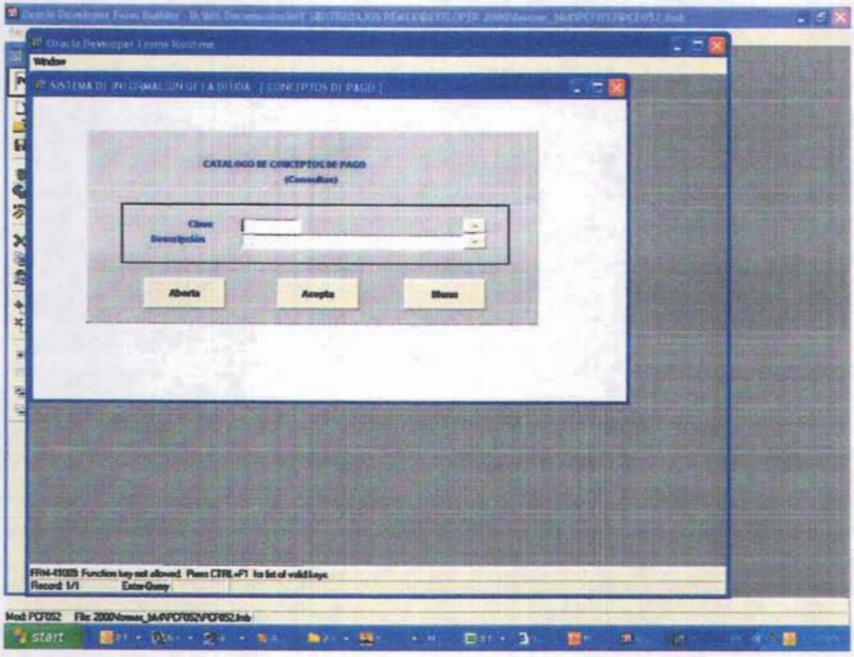


Figura 1.5 Catálogo de Conceptos de Pago. Desarrollado en Oracle Developer

1.4.5.1 Requerimientos de seguridad

Validaciones generales

La tabla fuente de este catálogo es SID.R052, mostremos los campos y sus especificaciones:

Campo	Nombre en SID.R052	Restricciones	Tipo de dato
Concepto de pago	CPAGO	NOT NULL	VARCHAR2(3)
Descripción del Pago	DPAGO		VARCHAR2(30)
Tipo de movimiento	TMOVIM		VARCHAR2(3)
Devenga	DEVENGA		VARCHAR(2)

Validaciones particulares de cada campo

Concepto de pago

- Dado que este campo es la llave primaria, únicamente hay que revisar que la clave no exista.

Tipo de movimiento

- Este campo no es actualizable.

Devenga

- Únicamente tendrá 3 opciones: "SI", "NO" o "SN", para ello se utilizará el Combo Box correspondiente.

Validaciones para bajas o cambios

Se podrá actualizar o borrar registros, siempre y cuando el campo CPAGO del la tabla correspondiente no exista en el campo CPAGO de las tablas SID.R050 ó SID.R051.

1.4.6 Catálogo de tipos de cambio de Cierre

Al final de cada mes, se efectúa un cierre financiero mensual. Para ello es necesario conocer cuánto vale el tipo de cambio para cada una de las monedas internacionales respecto al peso mexicano, y para ello se utiliza el Catálogo de Tipos de Cambio de Cierre.

1.6 Catálogo de Tipos de Cambio de Cierre. Desarrollado en Oracle Developer

1.4.6.1 Requerimientos de seguridad

Validaciones generales

La tabla fuente de este catálogo es SID.TCAMBIO_CIE, mostremos los campos y sus especificaciones:

Campo	Nombre en SID.TCAMBIO_CIE	Restricciones	Tipo de dato
Fecha de Cambio	FCAM	NOT NULL	VARCHAR2(8)
Moneda	CMONEDA	NOT NULL	VARCHAR2(1)
Tipo de Cambio en Moneda Nacional	VCAMMN	NOT NULL	VARCHAR2(22,15)

La llave primaria son los campos FCAM y CMONEDA de la tabla.

Es necesario también indicar el valor de cambio en Dólar USA. Para ello se agregará un campo en la ventana el cual no pertenecerá a la tabla de la Base de Datos. El cálculo lo hará de manera automática una vez que se encuentre registrado el tipo de cambio en Dólar en la fecha correspondiente.

Validaciones particulares de cada campo

Fecha de cambio

- Al insertarla, se deberá verificar que sea mayor a la fecha del cierre anterior. El día tendrá que ser el último del mes que se desea cerrar. Esto lo hará de manera automática, es decir, al posicionarse en el campo correspondiente. Por omisión pondrá el último día del mes que se cerrará. Para validar esto, se colocará un Date Time Picker para la elección de la fecha.
- La fecha podrá ser adelantada respecto a la fecha actual.

Moneda

- Se colocará el Combo Box correspondiente con opción de elegir el nombre de la moneda, al elegir el campo se llenará con la respectiva clave.

Validaciones para bajas o cambios

Se podrá borrar o cambiar un registro si su fecha de tipo de cambio no ha cerrado. Para ello se requerirá hacer la consulta correspondiente para conocer la fecha del último cierre.

1.4.7 Pagos no programados

Cuando PEMEX hace sus movimientos financieros con los distintos acreedores, surgen pagos por anticipado que deben cubrirse. Su registro se hace en la aplicación llamada Pagos No Programados.

1.7 Aplicación Pagos No Programados. Desarrollado en Oracle Developer

1.4.7.1 Requerimientos de seguridad

Validaciones generales

La tabla fuente es la SID.R050, y su especificación es la siguiente:

Campo	Nombre en SID.R050	Restricciones	Tipo de dato
Contrato	CCONTRA	NOT NULL	VARCHAR2(7)
Desembolso	NDESEMB	NOT NULL	NUMBER(4)
Número de pago	NPAGO	NOT NULL	NUMBER

Situación del pago	ISITPA		VARCHAR2(1)
Concepto de pago	CPAGO	NOT NULL	VARCHAR2(3)
Devenga	DEVENGA		VARCHAR2(2)
Fecha valor	FVALVEN	NOT NULL	VARCHAR2(8)
Fecha inicio	FREPANT	NOT NULL	VARCHAR2(8)
Fecha fin	FREPSIG		VARCHAR2(8)
Monto	MMONTO		NUMBER(15,2)
Financiado	FINANCIADO		VARCHAR2(2)
Contrato que Financia	CCONTRA_FINAN		VARCHAR2(7)
Desembolso que Financia	NDESEMB_FINAN		NUMBER(4)
Fecha Tipo de Cambio	FTCPAGO		VARCHAR2(8)
Tasa ISR	VTASA_ISR		NUMBER(18,15)

La llave primaria son los campos CCONTRA, NDESEMB, FVALVEN, CPAGO y FREPANT. Se requiere también agregar campos que no pertenecen a la Base de Datos. Esto con el fin de mostrarle información adicional necesaria para el usuario. Los campos a agregar son: Acreedor, Expediente, Total, Moneda, Tasa Piramidal y Monto Piramidal.

Validaciones particulares de cada campo

Contrato

- Deberá tener un Combo Box con los contratos ya registrados.

Desembolso

- También deberá tener un Combo Box para elegir el desembolso. Al elegir el contrato, dicho Combo Box se deberá llenar con los desembolsos asociados al contrato elegido.

Acreedor

- Al elegir el contrato, en este campo se mostrará el acreedor con el que fue abierto dicho contrato. Para extraer el dato se utiliza el campo SID.R003.DACRE.
- Este campo no es modificable.

Expediente

- Al elegir el desembolso se deberá hacer la consulta correspondiente en la tabla de desembolsos para encontrar el número de expediente para posteriormente mostrarlo al usuario. Para extraer el dato se utiliza SID.R019.CEXPE.
- Este campo no es modificable

Número de pago

- Este campo es llenado automáticamente por un Data Base Trigger, pero al momento de insertar el registro, por omisión se llenará con "0".

Situación del pago

- Habrá un Combo Box con opciones: "INICIADO" y "REAL"; y el campo será llenado con "I" y "R" respectivamente.

Concepto de Pago

- Se podrán elegir los conceptos de pago que se encuentran tabla SID.R052, donde el usuario podrá elegir en un Combo Box la descripción del concepto de pago, pero el campo se llenará con la clave al momento de hacer la elección.
- Al insertar un nuevo registro, por omisión se llenará con la clave "DII" la cual hace referencia a un pago de interés devengado.

Devenga

- Se llena automáticamente en función del concepto de pago.
- En caso de que no sea determinado por el concepto de pago, por omisión tendrá la opción "NO" y tendrá la opción de cambiar.

Fecha Valor

- En el caso de que el concepto de pago se a "DII" o "PII", esta fecha determina la agrupación de pagos por familia, ya que se dice que dos pagos son de una misma familia si el concepto de pago es "DII" o "PII" y su Contrato, Desembolso y Fecha Valor son iguales.
- Si esta fecha es cambiada a un pago de la familia, tendrá que cambiarse a todos los pagos de la familia.

- Si es el primer pago de la familia esta fecha puede ser hasta un año antes. Si ya existen pagos en la familia, ésta tiene que ser mayor al cierre y no menor a la Fecha Inicio del último pago de la familia.
- En caso de que el concepto de pago no sea "DII" o "PII", los pagos ya no serán agrupados por familia, y entonces únicamente esta fecha tiene que ser mayor al cierre.

Fecha Inicio

- En caso de que sea un pago que pertenece a una familia, esta fecha tiene que ser menor a Fecha Valor y a Fecha Fin.
- Tiene que ser del mismo mes que la Fecha Fin.
- También tiene que ser un mes más adelante que la Fecha Fin del pago anterior.
- Por omisión será el penúltimo día del mes actual.
- Si esta fecha ya cerró, no se puede actualizar todo el registro.
- En caso que el pago no pertenezca a ninguna familia, únicamente deberá ser menor a la fecha fin, y por omisión será un día menor a la fecha Fin.

Fecha Fin

- En caso de ser pago de alguna familia, tiene que ser menor a Fecha Valor y mayor a Fecha Inicio.
- Tiene que ser de un mes después de la Fecha Fin del pago anterior de la familia.
- En caso que Fecha Valor = Fecha Inicio = Fecha Fin, la familia se cierra y se introduce automáticamente un pago concepto "PII" con las mismas fechas, el cual contiene el monto de la suma de los pagos de la familia.
- En caso de que no sea elemento de familia, únicamente debe ser menor a la Fecha Valor.
- Por omisión será igual a la Fecha Valor.

Monto

- Deberá aceptar únicamente cantidades mayores a 0.

Total

- En este campo se calcula el total de los montos acumulados de los pagos que se muestran en la pantalla. Recordemos que este campo no pertenece a la Base de Datos.
- No es modificable.

Moneda

- Este campo es llenado automáticamente con el tipo de moneda con el cual se abrió el contrato.
- No es modificable.

Financia

- Llevará un Combo Box con opciones de "SI" o "NO".
- Por omisión el valor será "NO".
- En caso de elegir "SI", los dos campos siguientes serán llenados por omisión con el mismo contrato y desembolso de los campos anteriores.

Contrato que Financia

- Por omisión será nulo.
- Será llenado automáticamente en caso de que el campo Financia sea llenado con opción "SI".

Desembolso que Financia

- Al igual que el campo anterior, por omisión será nulo y será llenado automáticamente en caso que el campo Financia se llenado con SI.

Fecha Tipo de Cambio

- Por omisión será nula
- Tendrá un Date Time Picker y la fecha y será mayor o igual a la Fecha Valor.

Tasa ISR

- Deberá aceptar sólo cantidades mayores a 0.

Tasa Piramidal

- Esta cantidad se calcula a partir del Monto y la Tasa ISR, dicho cálculo se efectúa con una función de la Base de Datos, así que el campo es llenado automáticamente.
- Este campo no es modificable.

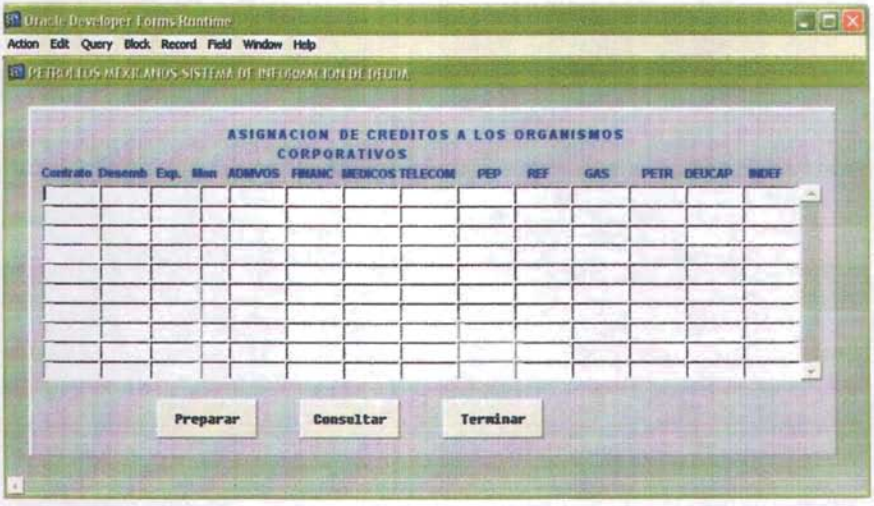
Monto Piramidal

- Su validación es similar que la Tasa Piramidal.

1.4.8 Asignación de Créditos a los organismos corporativos

Cada que PEMEX hace un desembolso, se tiene que definir la proporción que pagará cada organismo. En esta aplicación se puede consultar qué proporción le toca pagar a cada organismo respecto a los desembolsos.

A continuación mostramos la pantalla de Asignación de Créditos a los Organismos Corporativos:



1.8 Aplicación Asignación de Créditos a los Organismos Corporativos. Desarrollado en Oracle Developer

Dado que esta aplicación es sólo para consulta, únicamente se requiere conocer la especificación de la tabla de la Base de Datos, la cual es una vista:

Campo	Nombre en SID.R052	Restricciones	Tipo de dato
Contrato	CONTRATO	NOT NULL	VARCHAR2(7)
Desembolso	DESEMBOLSO	NOT NULL	NUMBER(4)
Expediente	EXP_PAGO		VARCHAR2(6)
Moneda	MONEDA		VARCHAR2(1)
Corporativo	CORP		NUMBER
Exploración	PEP		NUMBER
Refinación	REF		NUMBER
Gas y petroquímica básica	GAS		NUMBER
Gas y petroquímica secundaria	PETR		NUMBER
Deuda capitalizable	DEUDOC		NUMBER
Indefinido	INDEF		NUMBER

1.4.9 Catálogo de Acreedores

Esta aplicación ya está desarrollada en tecnología .NET. Únicamente se tiene que integrar con el resto de las aplicaciones, hacerle ajustes para unificar la interfaz gráfica.

1.5 Desarrollo de aplicación para Monitoreo de flujo de información de pagos y desembolsos

Como ya se mencionó en el apartado 1.1.2, el sistema la información va sufriendo cambios al ser procesada y al ser retroalimentada por diferentes aplicaciones. Es natural que esto tarde cierto tiempo dependiendo del proceso que se ejecute y del tipo de cambio que se le haga. Y por lo regular el usuario requiere con urgencia los resultados de dicha información cuando ésta se encuentra en proceso.

Se requiere una aplicación para monitorear todos estos procesos y conocer el estado de la información en cualquier instante. Esta aplicación la utilizarán los administradores del Sistema, y su objetivo es hacer las consultas al sistema de una manera más rápida, ya que anteriormente esto se tenía que hacer tecleando el código de las consultas. De esta forma el monitoreo se hará de una forma más amigable, sencilla, rápida y eficiente.

1.5.1 Requerimientos funcionales

R1. Revisar estado y trayectoria de los desembolsos en el Sistema así como todos sus atributos relevantes.
R2. Revisar estado y trayectoria de los pagos en el Sistema así como todos sus atributos relevantes.

1.5.2 Requerimientos no funcionales

Requerimientos de ejecución

- La aplicación deberá ser ejecutada en máquinas con sistema operativo WIN32 y plataforma .NET.

Requerimientos operacionales

- El único requerimiento es que la máquina donde se instalen las aplicaciones esté conectada al servidor donde se encuentra la Base de Datos.

Facilidad de uso

- Deberán estar enlistadas las tablas donde se revisarán los pagos y desembolsos, para elegir dónde hacer la revisión.
- La revisión de datos se hará de forma multiregistro.
- Los componentes gráficos de cada aplicación (botones, menús, listas de elementos, etc.) deben ser claros para que el usuario fácilmente pueda realizar sus diferentes tareas.
- La consulta para revisión de pagos y desembolsos deberá poderse hacer por diversas formas de búsqueda, dependiendo de las tablas de la Base de Datos a la que hagamos referencia.

Requerimientos de interfaz gráfica

- El usuario deberá tener a la vista y de forma ordenada los elementos a utilizar para delimitar sus consultas.

Requerimientos de seguridad

- Únicamente los administradores pueden acceder a esta aplicación.
- Esta aplicación únicamente es para consulta, ya que los que hacen las actualizaciones son los usuarios. Así que no habrá forma de modificar datos.

Requerimientos legales

- Que estén al día las licencias del software a utilizar.

Tiempo de respuesta

- Se recomienda usar al máximo las bibliotecas de ADO.NET para que la manipulación de la Base de Datos se haga en el menor tiempo posible.

CAPÍTULO 2

DESCRIPCIÓN DE LA PLATAFORMA .NET

La plataforma .NET es un ambiente utilizado para desarrollar aplicaciones y ejecutarlas en sistemas operativos WIN32. El desarrollo de aplicaciones en esta plataforma no depende de ningún lenguaje, es decir, se puede desarrollar de forma multilenguaje.

2.1 El Framework de .NET

Es la parte medular de la plataforma .NET, es una infraestructura que se utiliza para integrar y ejecutar aplicaciones basadas en Microsoft Windows. Se compone principalmente de los siguientes elementos:

- Un conjunto de compiladores de diversos lenguajes
- Biblioteca de clases
- Máquina virtual

2.1.1 Los compiladores de .NET

La plataforma .NET soporta múltiples lenguajes de programación, y a pesar de que cada lenguaje contiene características propias, se puede desarrollar cualquier aplicación que corra en un sistema operativo WIN32 en cualquiera de estos lenguajes. En la actualidad se han adaptado más de 30 lenguajes a la plataforma .NET, los más comunes son: C#, Visual Basic y C++.

Al ser compilado un código fuente en cualquier lenguaje soportado por .NET, se traduce a un código intermedio llamado MSIL (Microsoft Intermediate Languaje). Cabe mencionar que este código no se puede ejecutar directamente, ya que no es lenguaje máquina; y las reglas necesarias para generar dicho código se encuentran en el Common Lenguaje Specification (CLS).

Por eso se dice que la plataforma .NET es independiente del lenguaje, ya que cuando se compila un código fuente, siempre será traducido a MSIL independientemente del lenguaje en el que se encuentre dicho código fuente.

2.1.2 Bibliotecas de clases de .NET

Al programar en un lenguaje de programación, por lo regular tratamos de aprovechar al máximo las bibliotecas del lenguaje para reducir el tiempo de programación. Para el caso de .NET las bibliotecas son comunes para todos sus lenguajes. Dichas bibliotecas son un conjunto de clases agrupadas en un espacio de nombres jerárquico (namespaces). De esta forma podemos acceder a estas clases desde cualquier lenguaje de la plataforma usando la misma sintaxis de invocación.

Para que todo esto sea posible, el Framework posee un sistema de tipos universal, denominado Common Type System (CTS). Este sistema permite que el programador pueda hacer interactuar las clases de las bibliotecas .NET con las creadas por él mismo, y utilizar clases en cierto lenguaje desde un lenguaje distinto. De esta forma se aprovechan las ventajas de la programación orientada a objetos.

Las Bibliotecas de clases de .NET también son llamadas modelo unificado de clases base.

2.1.3 La máquina virtual de la plataforma .NET

Es llamada Common Languaje Runtime (CLR), y es el núcleo del Framework de .NET. Básicamente se encarga de compilar dinámicamente código MSIL a código máquina para poder ser ejecutado; esto se logra gracias a un compilador llamado Just In-Time Compiler (JIT). El CLR maneja la memoria de una forma eficiente, por ejemplo cuenta con un recolector de basura para hacer más fácil para el programador el manejo de memoria. Todo esto en general está implementado para todos los lenguajes, pero hay excepciones.

2.1.3.1 Código manejado y no manejado

Se llama código manejado (Managed Code) al que el CLR administra y ejecuta, y sobre el cual tiene absoluto control. Pero ¿cuándo podría suceder lo contrario? En el caso de Visual Basic, nunca, ya que sólo puede producir ejecutables que corran sobre la plataforma y, por tanto, que estén ligados al CLR. Pero hay dos excepciones que podemos nombrar: C# puede tener secciones de código marcadas como "unsafe" o inseguras, en las cuales es posible usar punteros (pointers) que no estén regulados por el

CLR. Por omisión C++ no crea ejecutables para la plataforma .NET, por lo que puede prescindir del CLR (aunque esto sí puede hacer, mediante las Manager Extensions para C++). Ésta es una de las grandes ventajas del uso de lenguaje Visual Basic para aplicaciones que corran en la plataforma .NET, ya que todo el manejo de memoria es controlado por el CLR.

2.2 Visual Studio .NET

Visual Studio .NET es una aplicación que se utiliza para el desarrollo de aplicaciones que son ejecutadas en plataforma .NET. Su función principal es disminuir el tiempo de programación y permitir la edición de código de una manera más fácil y organizada.

2.2.1 Diseñador de interfaces gráficas

Una de las principales ventajas del Visual Studio es la facilidad que ofrece para la construcción de interfaces gráficas mediante los Windows Forms. Cualquier componente visual de Windows se puede hacer con este diseñador. Con el ratón de la computadora se construye cualquier ventana con sus respectivos controles así como asociar propiedades a cada control en el panel de propiedades.
Una vez construida una interfaz gráfica, el diseñador genera el código respectivo el cual es mostrado para poder ser modificado. Esto reduce grandemente el tiempo de programación.
Todo esto permite que sin muchos conocimientos de programación se puedan desarrollar aplicaciones de manera cómoda y rápida.

2.2.2 Facilidad de desarrollo

Visual Studio cuenta con muchas herramientas para facilitar la programación, y aunque no vamos a entrar en detalles, podemos mencionar algunas:

- Capacidad de completar código al iniciar una estructura de control
- Generación de código al elegir eventos, métodos y propiedades asociadas a los componentes Windows.
- Cuenta con asistentes para elaboración de componentes comunes en las aplicaciones.

- Tiene diversas herramientas de depuración.
- Fácil integración y distribución de proyectos.
- Señalamientos de errores de compilación en tiempo de edición de código.
- Fácil navegación en las clases del proyecto, métodos, propiedades, eventos, etc.

Todo esto permite que la edición de código se reduzca de gran manera haciendo así que el desarrollo de un proyecto se realice en menor tiempo.

2.3 ADO .NET

ADO (ActiveX Data Objects) .NET es el nombre de la tecnología que utiliza la plataforma .NET para el acceso a bases de datos. Como en cualquier lenguaje que permite conexiones a bases de datos, en .NET se puede acceder directamente a bases de datos utilizando los respectivos comandos SQL mediante una conexión.

2.3.1 Adaptadores de datos (Data Adapters) y conjuntos de datos (Data Sets)

La principal ventaja de la tecnología ADO .NET es el trabajo con bases de datos en forma desconectada.

Es común que cuando se utiliza una aplicación de software que hace consultas y actualizaciones a una Base de Datos, se mantengan conexiones con el servidor de manera prolongada y constante. Esto es poco eficiente ya que consume demasiados recursos tanto en el Servidor como en el Cliente. Ahora con la tecnología ADO .NET de Microsoft esto se ha reducido mediante el uso de los adaptadores y conjuntos de datos.

Un adaptador de datos (Data Adapter) es un enlace entre la Base de Datos y un conjunto de datos que se encuentra en la memoria de la computadora donde se está ejecutando la aplicación. Funciona de la siguiente forma: Se crea un adaptador de datos y se define la conexión a la Base de Datos, se construye la consulta (con lenguaje SQL) con la que trabajará el adaptador; y una vez creado el adaptador, automáticamente se crean comandos SQL de consulta, inserción, actualización, y borrado de datos. Posteriormente se crea un conjunto de datos (Data Set) a partir del adaptador; el conjunto de datos es un conjunto de tablas que serán llenadas por los datos arrojados de la consulta con la que se generó el adaptador. Para llenar las tablas, únicamente se invoca el método correspondiente y la conexión se abre mientras las tablas son llenadas, y una vez hecho esto, ya se pueden insertar, actualizar y borrar datos en las tablas del conjunto de datos.

Todo esto se hace en memoria, es decir, que la conexión con el servidor está cerrada. Se lleva un control de qué registros han sido actualizados y qué nuevos registros se han insertado, y al ejecutar el método que actualiza en la Base de Datos, la conexión se abre; y mediante los comandos de inserción, actualización y borrado, la Base de Datos es actualizada en función de los cambios que anteriormente se habían realizado en el conjunto de datos.

Esto es muy eficiente, ya únicamente se abre la conexión para traer los datos y para actualizar los cambios hechos, y se ejecutan los métodos correspondientes de llenado y actualización, ya que se trabaja sobre los datos en memoria, no en el servidor.

2.3.2 Data Bindings

Muchos de los controles de los WinForms despliegan información para ser elegida, agregada o solamente mostrada (ListBox, Combo Box, DataGrid, etc.). Una forma rudimentaria de suministrarles la información, es accediendo por medio del código a sus propiedades e insertarles dato por dato. Pero la tecnología ADO .NET permite que la obtengan directamente de la tabla de un conjunto de datos. Y mediante los Data Bindings, al modificar la información en alguno de estos controles, el cambio se verá reflejado en la tabla del conjunto de Datos.

Todo lo anterior facilita el manejo de aplicaciones que manipulan información de la Base de Datos, ya que no hay que estar trayendo constantemente información de la Base de Datos, sino que solo lo hacemos una vez y la suministramos a los distintos controles que la requieran.

CAPÍTULO 3

DESCRIPCIÓN DEL DISEÑO

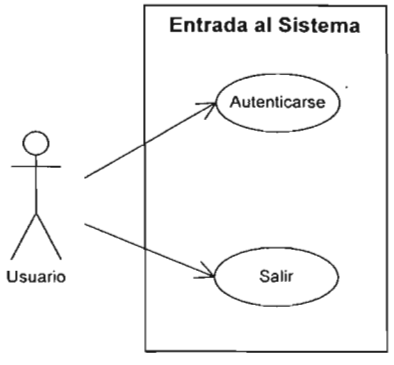
En este capítulo se mostrará el modelado de las aplicaciones para de esta forma conocer de cómo serán implementadas. Para ello se utilizarán diagramas de casos de uso y de clases, ya que consideramos que éstos son los principales diagramas utilizados en UML.

3.1 Diagramas de casos de uso

En el capítulo 1 se enlistaron los requerimientos funcionales, a partir de ellos hemos extraído los siguientes casos de uso:

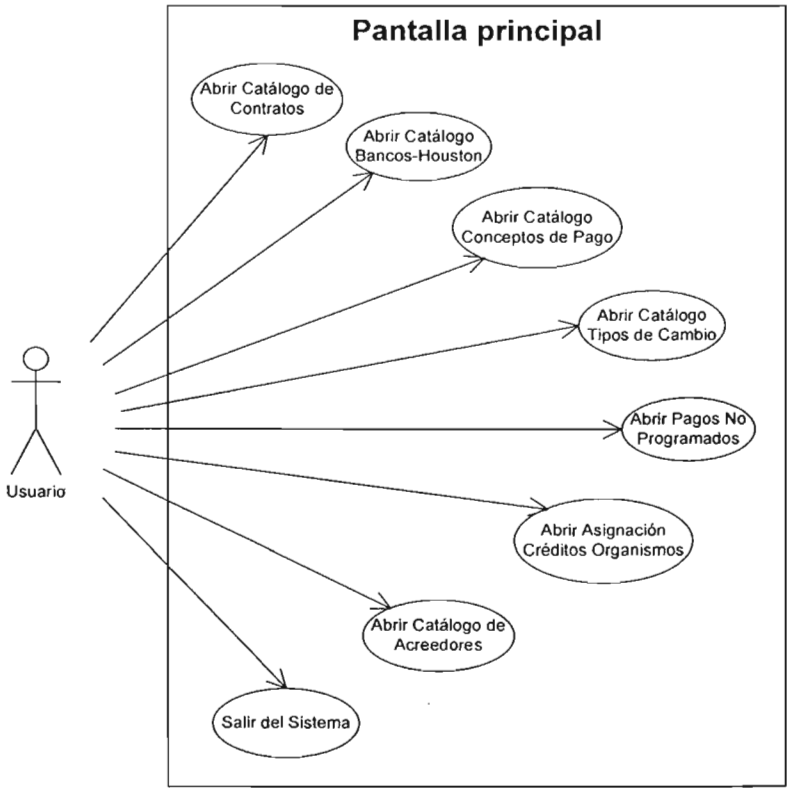
3.1.1 Diagramas de casos de uso de la entrada a la aplicación

Al iniciar el sistema, el usuario puede realizar dos tareas: autenticarse y salir del sistema.



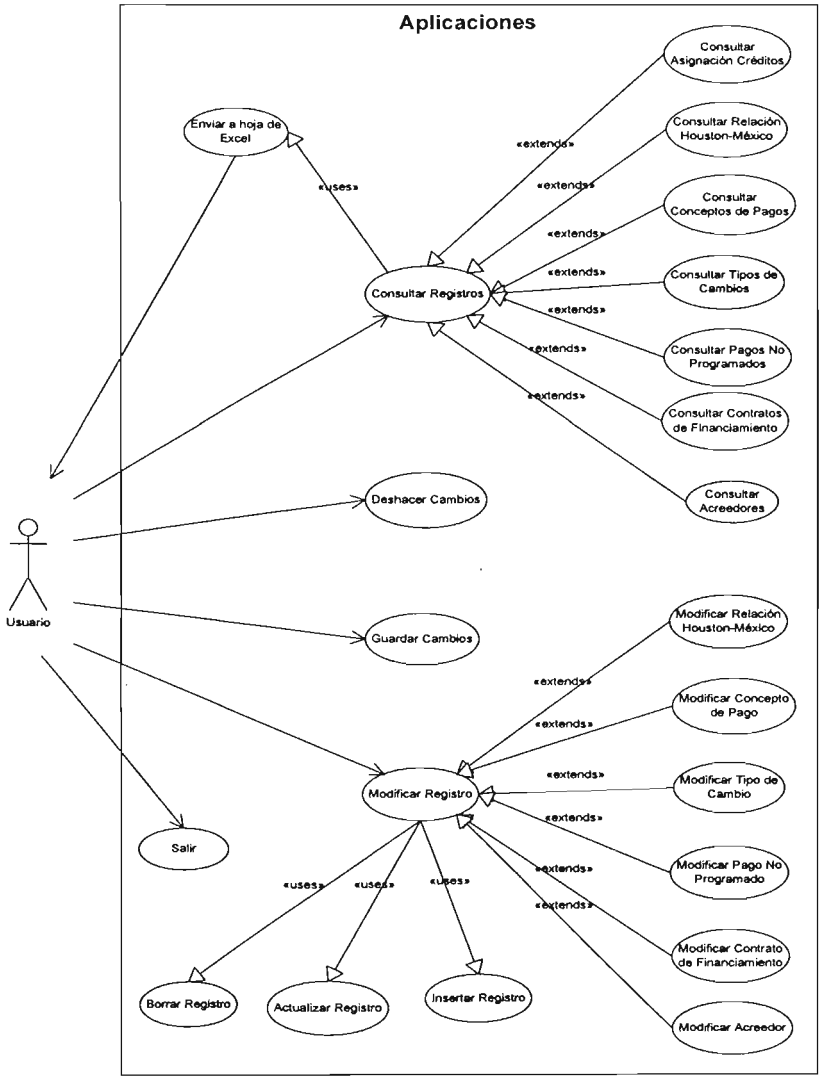
3.1.2 Diagramas de casos de uso de la pantalla principal

Una vez que el usuario se ha autenticado, tiene opción de abrir las aplicaciones:



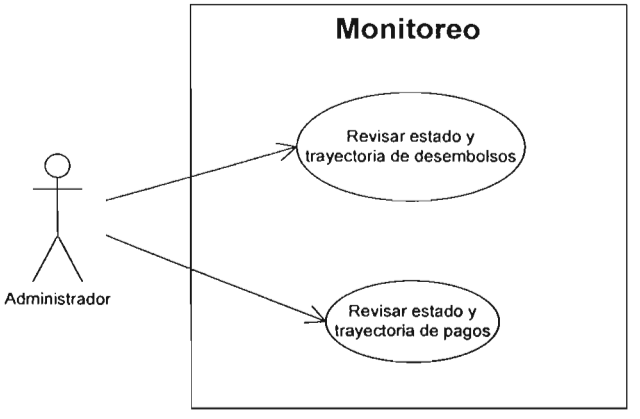
3.1.3 Diagramas de casos de uso de las aplicaciones a migrar

Para cada aplicación el usuario puede realizar diversas tareas de las cuales la mayoría son comunes para todas las aplicaciones, y todo esto lo hemos englobado el siguiente diagrama de casos de casos de uso:



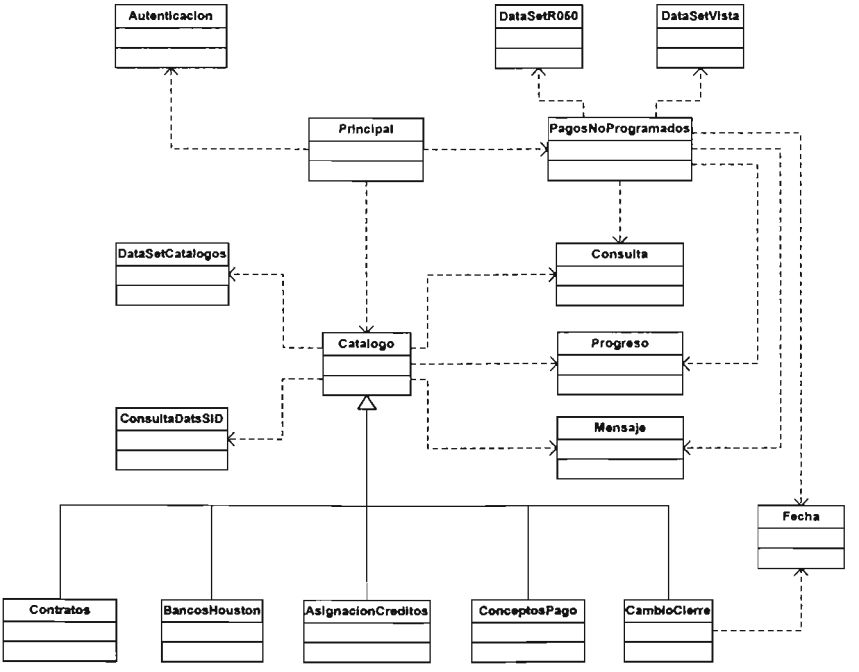
3.1.4 Diagrama de casos de uso de la aplicación monitoreo de flujo de información de pagos y desembolsos

Recordemos que el administrador del sistema, tiene dos tareas primordiales: monitorear pagos y monitorear desembolsos.



3.2 Diagrama de Clases de las aplicaciones a migrar

Las clases utilizadas y sus respectivas relaciones las mostramos a continuación:



En el Apéndice B se describe a detalle qué atributos, métodos y eventos contiene cada clase y para qué son utilizados.

3.3 Descripción de las clases de las aplicaciones a migrar

Durante este apartado mencionaremos una justificación de porqué fueron utilizadas estas clases y que función desempeñan. El detalle de atributos, métodos y eventos se muestra en el apéndice.

3.3.1 Descripción de la clase Mensaje

Esta clase fue implementada para poder enviar mensajes de todo tipo al usuario, como son preguntas, información, mensajes de error etc. Hereda de `System.Windows.Forms.Form`.

3.3.2 Descripción de la clase Consulta

Esta clase es utilizada por la mayoría de de las clases, ya que es la que permite hacer consultas a la Base de Datos.

3.3.3 Descripción de la clase ConsultaDatosSID

En el Sistema de Información de Deuda de la Gerencia de Financiamientos y Análisis de Mercado de PEMEX, existe información que es consultada muy repetidamente. Y esta clase fue creada para el fácil acceso a los datos más utilizados. Principalmente se utiliza para llenar de información los Combo Box.

3.3.4 Descripción de la clase Progreso

Su funcionamiento radica en mostrar al usuario el avance de ciertos procesos cuando éstos tardan en ejecutarse un tiempo considerable. Hereda de `System.Windows.Forms.Form`.

3.3.5 Descripción de la clase Autenticación

Se encarga de hacer la autenticación de usuario. Despliega una ventana donde el usuario ingresa su usuario y contraseña. Hereda de `System.Windows.Forms.Form`.

3.3.6 Descripción de la clase Catalogo

Esta es la clase padre de todos los catálogos. Aquí es donde se implementa la solución para la mayoría de los requerimientos funcionales en común para todos los catálogos. Hereda de `System.Windows.Forms.Form`.

3.3.7 Descripción de la clase Contratos

Es esencialmente la aplicación Catálogo de Contratos. Aquí se maneja todo el funcionamiento del catálogo y su interfaz gráfica. Hereda de la clase Catalogo.

3.3.8 Descripción de la clase BancosHouston

Se encarga de controlar el funcionamiento de la aplicación Catálogo Internase Bancos Houston-México. Hereda de Catalogo.

3.3.9 Descripción de la clase ConceptosPago

Controla todo el funcionamiento de la aplicación Catálogo de Conceptos de Pago. Hereda de la clase Catalogo.

3.3.10 Descripción de la clase CambioCierre

Es la clase se encarga de todo el funcionamiento de la aplicación Tipos de Cambio de Cierre. Hereda de la clase Catalogo.

3.3.11 Descripción de la clase AsignacionCreditos

Su función es controlar el funcionamiento de la aplicación Asignación de créditos a los organismos corporativos. Hereda de la clase Catalogo.

3.3.12 Descripción de la clase PagosNoProgramados

En esta clase es implementada la aplicación Pagos No Programados. Hereda de `System.Windows.Forms.Form`.

3.3.13 Descripción de la clase Principal

Ésta es la clase que controla a la pantalla principal, dicha pantalla permite que sean ejecutadas las aplicaciones por medio de los menús y las barras de herramientas. Hereda de `System.Windows.Forms.Form`.

3.3.14 Descripción de la clase FormAcreMovs y FormTransac

Recordemos que el Catálogo de Acreedores ya estaba implementado y sólo había que integrarlo. Las clases que lo conforman son FormAcreMovs y FormTransac. Únicamente serán utilizadas por la clase Principal.

3.3.15 Manejo de Data Sets y Autogeneración de clases

Al generar un Data Set por medio de un Data Adapter. Visual Studio genera una clase con el nombre del Data Set. Esta clase permite manejar de manera automática hacer consultas y cambios a la Base de Datos sin necesidad de usar código SQL. Consideramos que no vale la pena mostrar la estructura de dichas clases, solo las enlistaremos, ya que anteriormente se mencionó para qué eran utilizadas: DataSetCatalogos, DataSetR050, DataSetVista, DataSetMonitoreo.

3.4 Diagrama de Clases de la aplicación de monitoreo de pagos y desembolsos

Las clases utilizadas y sus respectivas relaciones las mostramos a continuación:

3.5 Descripción de Clases de la aplicación de monitoreo de pagos y desembolsos

De igual forma que para las aplicaciones a migrar, describiremos las clases utilizadas.

3.5.1 Descripción de la clase Autenticación

Esta clase es la misma que en las aplicaciones a migrar.

3.5.2 Descripción de la clase Monitoreo

En ésta clase se hace todo le funcionamiento del monitoreo de información de pagos y desembolsos. Hereda de `System.Windows.Forms.Form`.

CAPÍTULO 4

ESTRATEGIA DE IMPLEMENTACIÓN

A lo largo de este capítulo se describirá el orden de implementación, integración y pruebas de las aplicaciones de la migración.

4.1 Clase Catalogo

Durante la fase de diseño primeramente se enlistaron los requerimientos funcionales de los catálogos, y al hacerlo se vio que la mayoría de los requerimientos funcionales eran comunes. Así que al hacer el diseño de clases se pensó que habría una clase padre llamada *Catalogo* de la cual heredarían las clases correspondientes de los catálogos a implementar.

Figura 5.1 Pantalla del Catálogo de Contratos de Financiamiento

Dicha clase Catalogo fue la primera clase que se implementó, el objetivo de la clase fue que cumpliera con los requerimientos funcionales comunes a todos los catálogos. Esta clase utiliza la tecnología ADO .NET, así que contiene un Data Adapter y un Data Set donde se hacen las consultas y actualizaciones a la Base de Datos, y cada clase que

hereda de Catalogo hace referencia a su respectivo Data Adapter y Data Set de Catalogo. Recordemos que al construir un Data Adapter y un Data Set, Visual Studio .NET genera una clase que se encarga del manejo de datos del Data Set. Para probar la clase Catalogo se construyó una clase auxiliar de prueba que heredara de Catalogo. Recordemos que Catalogo hereda de `System.Windows.Forms.Form`, así que para mayor rapidez, se utilizó el diseñador del Visual Studio para programar la interfaz gráfica.

Figura 5.2 Pantalla de Catálogo de Conceptos de Pago

Una vez terminada la clase auxiliar, se ejecutó la ventana y se revisó que cumpliera con los requerimientos funcionales comunes para los catálogos (consulta de datos, inserción, modificación, etc.). Posteriormente se prosiguió a implementar otras clases a utilizar y las clases que heredan de Catalogo.

4.2 Otras clases

Para la implementación de los catálogos y la aplicación Pagos No Programados se requirieron otras clases: Consulta, ConsultaDatosSID, Fecha, Mensaje y Progreso. Estas clases fueron implementadas inmediatamente después de la clase Catalogo y fueron probadas conforme eran utilizadas por las clases que heredan de Catalogo.

Figura 5.3 Ejemplo de mensaje mandado al usuario

Figura 5.4 Ventana que muestra al usuario el progreso de un proceso

4.3 Clases que heredan de la clase Catalogo

Una vez probada la clase Catalogo, se implementó y probó la clase Contratos y se prosiguió a implementar los requerimientos de seguridad, es decir, las validaciones de inserción, modificación y borrado de registros.

Figura 5.5 Pantalla del Catálogo Interfase Bancos Houston-Mexico

Posteriormente se procedió a implementar las clases: ConceptosPago, BancosHouston, AsignacionCreditos y CambiosCierre, las cuales se implementaron de manera similar a la clase Contrato.

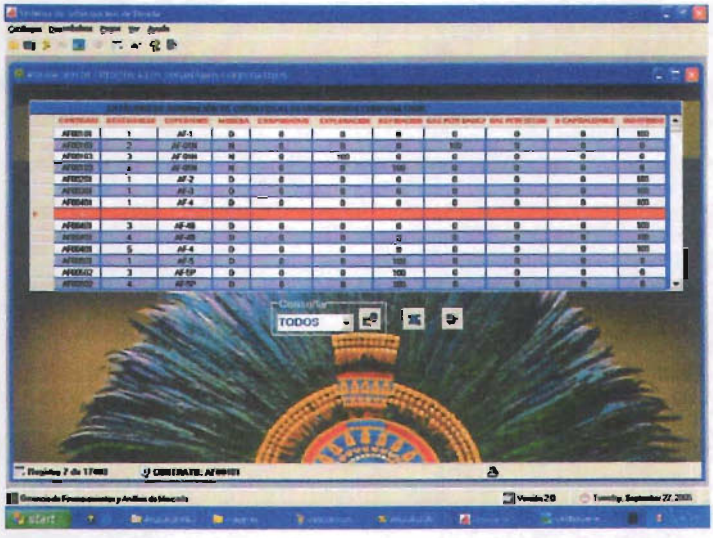


Figura 5.6 Pantalla de la aplicación Asignación de Créditos a los Organismos Corporativos

Cabe mencionar, que al igual que en la clase Catalogo, para la implementación de estas clases también se utilizó el diseñador de Visual Studio para programar el resto de la interfaz gráfica; y para la inserción de nuevos registros, se utilizaron los Data Bindings de ADO .NET.

4.4 Clase PagosNoProgramados

Debido a que esta clase no hereda de Catalogo, su implementación se hizo terminando las clases anteriores, también se utilizó el diseñador de Visual Studio y ADO .NET para la implementación. Aquí se generaron dos Data Sets, uno para consulta y otro para actualizaciones. La consulta y actualización de datos se manejan de manera similar a los catálogos. Las validaciones de inserción y actualizacón de datos, al ser más estrictas que en los catálogos, tardaron más en implementarse.

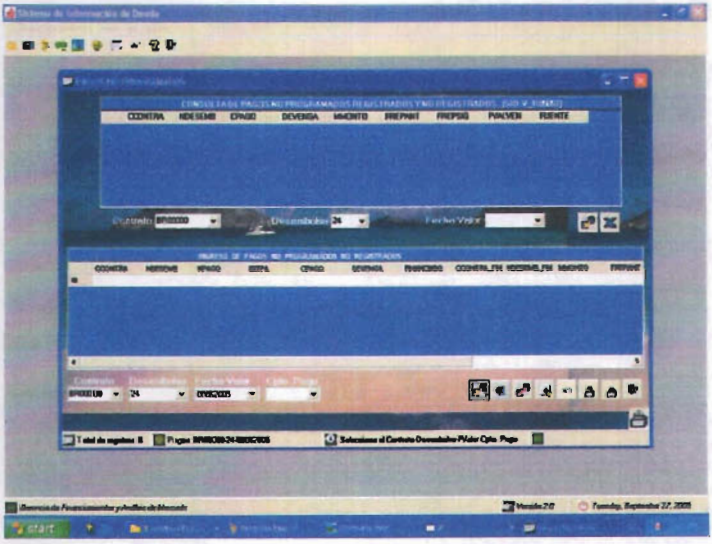


Figura 5.7 Pantalla de la Aplicación Pagos No Programados

5.5 Integración

Las pruebas de los catálogos y la aplicación de Pagos No Programados, se hicieron de manera independiente, es decir, se corría cada aplicación por sí sola. Una vez que funcionaron correctamente se procedió a implementar una ventana que integrara a cada

una de las aplicaciones. Para ello primeramente se implementó la clase Autenticacion, únicamente hace una consulta a la Base de Datos para revisar que el usuario se encuentre registrado.

Figura 5.8 Pantalla del Catálogo de Tipos de Cambio de Cierre

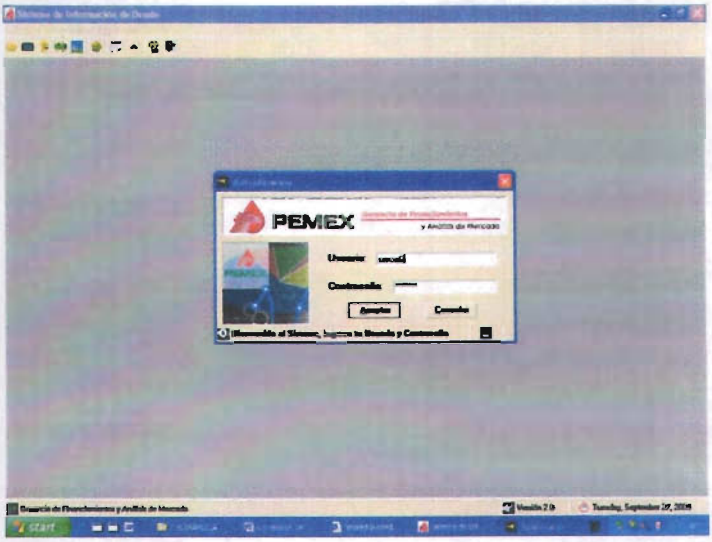


Figura 5.9 Pantalla de autenticación

Posteriormente se implementó la clase Principal, esta clase contiene menús y barras de herramientas que se utilizan para acceder a cada una de las aplicaciones, y para ello se utilizó el diseñador del Visual Studio. Una vez construida la interfaz gráfica, se le asignaron los eventos correspondientes a los controles para el acceso a las aplicaciones.

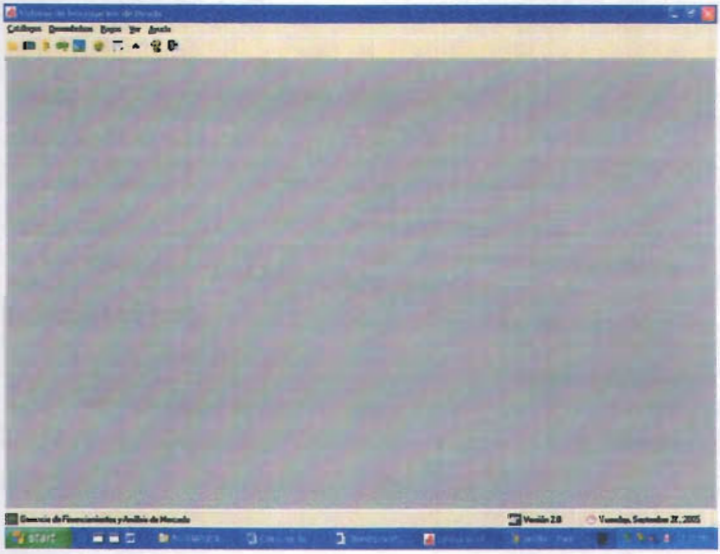


Figura 5.10 Pantalla principal

5.6 Clase Monitoreo

La aplicación del monitoreo de pagos y desembolsos, al ser para el administrador del sistema, se implementó de manera independiente a las demás. Todo su funcionamiento se encuentra en la clase Monitoreo, consta básicamente de controles para la selección de consultas y diversos Data Adapters y Data Sets que permiten la consulta de datos. Dichos datos son vistos en un control Data Grid de .NET. El Data Grid es un control que permite visualizar datos en forma de tabla.

5.7 Conclusiones

De esta forma, la migración fue concluida de forma satisfactoria. Las aplicaciones nuevas realizan lo que las anteriores y los requerimientos nuevos, con la diferencia de que su uso es más ágil.

Una vez concluido el desarrollo de las aplicaciones en .NET, se hicieron pruebas con los usuarios que utilizaron por muchos años las aplicaciones hechas en Oracle Developer. Únicamente se les explicó a los usuarios de manera muy general el manejo de dichas aplicaciones, ya que la forma de uso de cada aplicación era clara a primera vista.

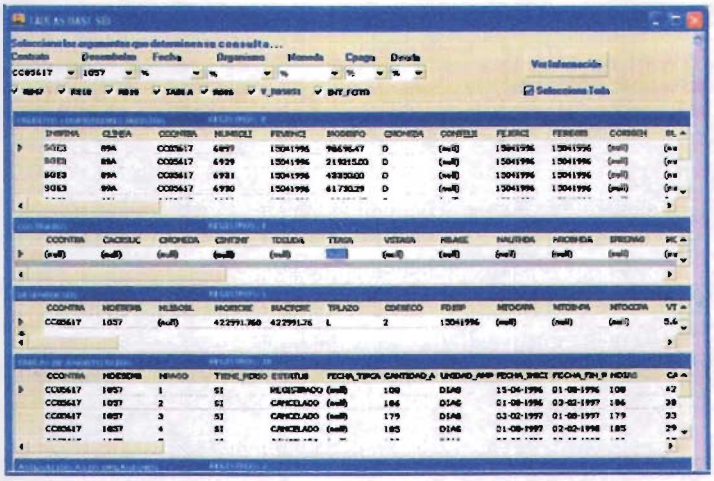


Figura 5.11 Pantalla de la aplicación Monitoreo de Pagos y Desembolsos

CONCLUSIONES

Una vez concluida la migración de las aplicaciones, el objetivo final del proyecto y las necesidades del usuario fueron cumplidas, ya que los usuarios realizan sus tareas que hacían con las aplicaciones anteriores, sólo que ahora lo hacen de una forma más eficiente. También el usuario puede realizar tareas nuevas que con las aplicaciones anteriores no podía hacer.

Esto queda demostrado al realizar una prueba con los usuarios la cual consiste en lo siguiente: Se eligen a dos usuarios y tres tareas a ejecutar y posteriormente se mide el tiempo que tardan los usuarios en ejecutar dichas tareas en las aplicaciones anteriores, y se compara con el tiempo que tardan en ejecutar las tareas en las aplicaciones migradas. En promedio el tiempo de ejecución de tareas con las aplicaciones migradas se optimiza en un 70%.

Consideramos que esto se debe a que .NET permite facilitarle al usuario el manejo de aplicaciones gracias a su gran variedad de controles que contiene para la creación de interfaces gráficas para el manejo de grandes cantidades de información.

Con ayuda de Visual Studio, se pueden desarrollar aplicaciones (al menos para cuestiones financieras y administrativas) de manera rápida y eficiente. Consideramos que esto se logra debido a las siguientes ventajas:

- *Programación visual.* Con Visual Studio, el programador puede crear su interfaz gráfica y parte del funcionamiento de la misma utilizando el diseñador, ya que el mismo diseñador genera el código que crea la interfaz gráfica.

- *Uso de asistentes y herramientas para el manejo de Bases de Datos.* El desarrollador puede programar consultas y actualizaciones a una Base de Datos utilizando asistentes para creación de Data Adapters y Data Sets, todo esto sin necesidad de escribir el código que crea estos controles. Únicamente se utilizan los métodos correspondientes para actualización de datos.

- *Facilidad para la Programación Orientada a Objetos en la plataforma .NET.* Visual Studio provee de diversas ayudas al programador al momento de escribir código, ya que muestra sugerencias para la creación de clases, objetos, métodos, atributos, etc.

Las ventajas anteriores permiten que se reduzca el tiempo usado en la programación de aplicaciones de software. Pero consideramos necesario que el programador conozca cómo opera Visual Studio, es decir, conocer las bibliotecas de clases utilizadas por el diseñador, cómo funciona el código que es generado, etc. Ya que todo programador debe saber justificar y documentar el código de la aplicación de software que desarrolló.

Al estudiar y descubrir las facilidades que provee Microsoft para el desarrollo de aplicaciones con .NET y Visual Studio, nos atrevemos a afirmar que .NET es buena opción para desarrollar aplicaciones para sistemas operativos Win32.

En esta ocasión sólo nos enfocamos a ciertas necesidades del usuario en un departamento de PEMEX, pero con base en lo obtenido se observó que la plataforma .NET puede ser utilizada para desarrollo de aplicaciones en cualquier institución o empresa que requiera de aplicaciones de éste tipo, asi como migrar aplicaciones de tecnologías obsoletas.

Durante el proceso de la migración, se presentaron ciertos problemas al utilizar la tecnología ADO .NET y el control DataGrid, ya que consideramos que actualmente dichos elementos todavía no cuentan con ciertas propiedades que también serían útiles para el programador. Para ello hubo que buscar soluciones alternativas, y al hacerlo se pudo culminar los requerimientos del usuario para cada una de las aplicaciones.

Como todo software, las aplicaciones migradas requieren de mantenimiento, ya que con el tiempo el usuario requerirá realizar nuevas tareas. Para ello por el momento se seguirá utilizando la tecnología .NET y sus novedades que presente con el paso del tiempo.

APÉNDICE A

DESCRIPCIÓN DE CAMPOS DE LAS TABLAS UTILIZADAS EN LA BASE DE DATOS

CAMPO	DESCRIPCIÓN
Acreedor	Banco con el que PEMEX tiene contratos de financiamiento
Base	Intervalo de tiempo (en dias) por el cual se rige la tasa de interés
Clave del banco	Clave asignada a los bancos del Catálogo Interfase bancos Houston-México
Clave CIF	Clave de institución financiera relacionada con un egreso
Concepto de pago	Clave que se le asignan a los pagos para agruparlos
Contrato	Convenio de PEMEX con un acreedor para poder generar desembolsos
Contrato que financia	Contrato que se encarga de financiar algún pago
Corporativo	Organismo que conforma PEMEX
Desembolso	Préstamo otorgado a PEMEX mediante un contrato de financiamiento
Desembolso que financia	Desembolso que se encarga de financiar algún pago
Descripción del pago	Nombre que se le asigna a los pagos para agruparlos
Devenga	Campo que indica si un pago devenga o no
Deuda capitalizable	Monto de deuda capitalizada al Gobierno Federal
Expediente	Clave que identifica a un conjunto de contratos
Exploración	Organismo que conforma PEMEX
Fecha de firma	Fecha de firma de un contrato
Fecha de registro	Fecha de registro un contrato
Fecha fin	Fecha donde culmina el pago de un interés devengado

Fecha inicio	Fecha donde inicia el pago de un interés devengado
Fecha tipo de cambio	Fecha de tipo de cambio a moneda nacional de cierto pago
Fecha valor	Fecha en que debe cubrirse un desembolso
Financiado	Campo que indica si un pago es financiado o no
Gas y petroquímica básica	Organismo que conforma PEMEX
Gas y petroquímica secundaria	Organismo que conforma PEMEX
Indefinido	Se refiere a cuando un crédito no se asigna ningún organismo de PEMEX
Institución intermediaria	Institución que intercede en un contrato
Linea	Clave para agrupar tipos de negociaciones con los acreedores
Moneda	Tipo de moneda extranjera
Monto	Cantidad del pago
Nombre	Nombre del banco
Número de autorización de Hacienda	Clave que asigna la Secretaría de Hacienda a un contrato abierto con un acreedor
Número de pago	Identificador que se le asigna a cierto conjunto de pagos
Número de registro de obligaciones	Es proporcionado por la Secretaría de Hacienda cuando se obtiene un financiamiento
Plazo	Se refiere al plazo a pagar de desembolsos en un contrato de financiamiento, puede ser corto, mediano o largo.
Prepagable	Indica si un pago puede ser prepagable o no
Refinación	Organismo que conforma PEMEX
Situación del pago	Estado del pago
Sobre tasa	Valor para cálculo de intereses
Tasa ISR	Tasa especial para impuestos sobre la renta
Tipo de cambio en moneda nacional	Valor que indica cuánto vale cierta moneda extranjera en moneda nacional
Tipo de crédito presupuestal	Clave que define el impacto presupuestal del

	financiamiento
Tipo de deuda presupuestal	Clave de deuda presupuestal
Tipo de movimiento	Se refiere a si un pago es un interés devengable, o un gasto
Tipo deuda	Puede tomar valores de "interna" o "externa"
Tipo tasa	Tipo de tasa que tendrán los desembolsos pertenecientes a algún contrato de financimiento
Tipo plazo	Tipos de plazos a pagar de los desembolsos, puede ser a corto, mediano o largo plazo

APÉNDICE B

DESCRIPCIÓN DE ATRIBUTOS, MÉTODOS Y EVENTOS DE LAS CLASES IMPLEMENTADAS

Notación a utilizar

Aquí se detallará los atributos, métodos y eventos utilizados. Los eventos en la plataforma .NET son métodos que se ejecutan cuando ocurre una acción en la aplicación. Cada evento define a qué acción es a la que responde.

La notación que se utilizará será la siguiente:

+ para indicar que un atributo o método es público (public)

- para privado (private)

para protegido (protected)

Para el caso de los atributos:

<+, - ó #> <atributo> : <Tipo de dato o clase>

Para el caso de los métodos:

<+, - ó #> <Nombre del método(nombre de los parámetros : tipo de dato)> : <Tipo de dato de regreso>

Para el caso de los eventos:

<+, - ó #> <Nombre del método asignado al evento(nombre de los parámetros: tipo de dato) => Objeto.Evento

Clase Mensaje

Atributos

`- components : System.ComponentModel.IContainer`
Se refiere al contenedor de los controles. Los controles son elementos gráficos colocados en las ventanas de las interfaces gráficas, dichos controles son los que interactúan con el usuario.

`- btnNo : System.Windows.Forms.Button`
Botón con que el usuario responde a una pregunta de manera negativa.

`- btnCancelar : System.Windows.Forms.Button`
Botón que se utiliza para cancelar lo que se le pide al usuario en el mensaje.

`- btnSi : System.Windows.Forms.Button`
Botón con que el usuario responde a una pregunta de manera afirmativa.

`- lblMensaje : System.Windows.Forms.Label`
Etiqueta donde va escrito el mensaje que se despliega al usuario.

- picWarning : System.Windows.Forms.PictureBox
Imagen que mostrará al usuario cuando el mensaje sea de alerta.

- picError : System.Windows.Forms.PictureBox
Imagen que mostrará al usuario cuando el mensaje sea de error.

- picPregunta : System.Windows.Forms.PictureBox
Imagen que mostrará al usuario cuando el mensaje sea una pregunta al usuario.

- picInformacion : System.Windows.Forms.PictureBox
Imagen que mostrará al usuario cuando el mensaje sea una información para el usuario.

- respuesta : MsgBoxResut
Tipo de respuesta que da el usuario, las cuales podrán ser las siguientes: MsgBoxResult.No, MsgBoxResult.Yes y MsgBoxResult.Cancel.

- xCentral : Integer = 192
Indica la posición horizontal del botón que irá en el centro del mensaje cuando únicamente haya un botón.

- xIzquierdo : Integer = 104
Cuando haya tres botones a escoger, ésta será la posición horizontal del botón que irá a la izquierda.

- xDerecho : Integer = 280
Cuando haya tres botones a escoger, ésta será la posición horizontal del botón que irá a la derecha.

- xCentralIzquierdo : Integer = 152
Cuando haya dos botones a escoger, ésta será la posición horizontal del botón que irá a la izquierda.

- xCentralDerecho : Integer = 240
Cuando haya dos botones a escoger, ésta será la posición horizontal del botón que irá a la derecha.

- yGeneral : Integer = 80
Posición vertical que tendrán todos los botones.

Métodos

+ New()
Constructor de la clase

Dispose(disposing : Boolean)
Destruye al objeto

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del mensaje.

+ respuesta() : MsgBoxResult
Regresa lo que responde el usuario al mensaje. Dicha respuesta se asigna dependiendo el botón que oprima el usuario: MsgBoxResult.No si el usuario oprime el botón de respuesta negativa, MsgBoxResult.Yes si el usuario oprime el botón de aceptar o de respuesta afirmativa y MsgBoxResult.Cancel si el usuario oprime el botón de cancelar.

+ mandaMensaje(mensaje : String)
Manda un mensaje al usuario donde solo se puede oprimir aceptar.
mensaje : Mensaje a desplegar.

+ mandaInformacion(titulo : String, mensaje : String, : textoBoton :String)
Manda un mensaje informativo acompañado de un icono de exclamación.
titulo: Texto del encabezado de la ventana.
mensaje: Texto del mensaje.
textoBoton: Texto del botón para aceptar.

`+ mandaError(titulo : String, mensaje : String, : textoBoton : String)`
Manda un mensaje de error acompañado de un icono de error.
`titulo:` Texto del encabezado de la ventana.
`mensaje:` Texto del mensaje.
`textoBoton:` Texto del botón para aceptar.

`+ mandaErrorCancela(titulo : String, mensaje : String, TextoBotonAceptar : String, textoBotonCancelar : String) : MsgBoxResult`
Similar al anterior, solo que aquí el usuario puede aceptar o cancelar.
`titulo:` Texto del encabezado de la ventana.
`mensaje:` Texto del error.
`textoBotonAceptar:` Texto del botón para aceptar.
`textoBotonCancelar:` Texto del botón para cancelar.

`+ mandaWarning(titulo : String, mensaje : String, textoBoton : String)`
Manda un mensaje de alerta al usuario con su respectivo icono.
`titulo:` Texto del encabezado de la ventana.
`mensaje:` Texto del mensaje de alerta.
`textoBoton:` Texto del botón para aceptar.

`+ mandaPregunta(titulo : String, mensaje : String, textoBotonSi : String, textoBotonNo : String) : MsgBoxResult`
Despliega una pregunta al usuario con su respectivo icono.
`titulo:` Texto del encabezado de la ventana.
`mensaje:` Texto de la pregunta.
`textoBotonSi:` Texto del botón para respuesta afirmativa.
`textoBotonNo:` Texto del botón para respuesta negativa.

`+ mandaPreguntaCancela(titulo : String, mensaje : String, textoBotonSi : String, textoBotonNo : String, textoBotonCancela : String) : MsgBoxResult`
Este método es similar al anterior, solo que aquí el usuario no solo puede responder si o no, sino que también puede cancelar.
`titulo:` Texto del encabezado de la ventana.
`mensaje:` Texto de la pregunta.
`textoBotonSi:` Texto del botón para respuesta afirmativa.
`textoBotonNo:` Texto del botón para respuesta negativa.
`textoBotonCancela:` Texto del botón para cancelar.

`- exclamacion(titulo : String, mensaje : String, textoBoton : String)`
Es usado por los métodos anteriores para acomodar los componentes de la ventana cuando el mensaje no es pregunta.

`- pregunta(titulo : String, mensaje : String, textoBotonSi : String, textoBotonNo : String, textoBotonCancela : String) : Boolean`
También es usado por los anteriores para acomodar los componentes de la ventana para cuando es una pregunta.

`- iconoInfo()`
Coloca el icono de información en la ventana del mensaje.

`- iconoError()`
Coloca el icono de error en la ventana del mensaje.

`- iconoWarning()`
Coloca el icono de alerta en la ventana del mensaje.

- botonesExclamacion()
Acomoda los botones en la ventana cuando el mensaje es de error, información o alerta.

- botonesPregunta()
Acomoda los botones en la ventana cuando el mensaje es de pregunta.

- botonesErrorCancela()
Acomoda los botones en la ventana cuando el mensaje es de error y el usuario puede cancelar.

- botonesPreguntaCancela()
Acomoda los botones en la ventana cuando el mensaje es de pregunta y el usuario puede cancelar.

Eventos

- btnSi_Click(sender : System.Object, e : System.EventArgs) => btnSi.Click
Se dispara cuando el usuario oprime el botón de aceptar o afirmación en algún mensaje.

- btnNo_Click(sender : System.Object, e : System.EventArgs) => btnNo.Click
Se dispara cuando el usuario oprime el botón de negación en los mensajes de pregunta.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando el usuario oprime el botón de cancelar en algún mensaje.

Clase Consulta

Atributos

- conexion : OleDb.OleDbConnection
La conexión a la Base de Datos.

- comando : OleDb.OleDbCommand
El comando a ser ejecutado en la Base de Datos.

- resultados : OleDb.OleDbDataReader
Colección de resultados obtenidos de la Base de Datos

- mensaje : Mensaje
Este objeto se utiliza para enviar los mensajes.

- numero : Integer
Cuando se hace una consulta a la Base de Datos y ésta regresa un escalar, dicho escalar es asignado a éste entero.

- cadena : String
Se le asigna el resultado de la consulta a la Base de Datos.

Métodos

+ New()
Constructor de la clase

+ extraeDato(query : String) : String
Nos regresa el dato a consultar.
Query: Sentencia SQL a ejecutar

+ hayDato(query : String) : Boolean
Indica si existe o no el dato a consultar.
Query: Sentencia SQL a ejecutar.

+ extraeEscalar(query : String) : Integer
Extrae un dato numérico entero de la Base de Datos.
Query: Sentencia SQL a ejecutar.

Clase ConsultaDatosSID

Atributos

- conexion : OleDbConnection
La conexión a la Base de Datos

- dataSetConsultas : New DataSet
El conjunto de Data Tables donde se almacenarán los datos

- adaptador : OleDbDataAdapter
El adaptador para los Data Tables

- nuevo : DataRow
La fila para insertar a los Data Tables

- dataTContratos : String = "CONTRATOS"
El nombre del Data Table para los Contratos de Financiamiento

- dataTContratosConsulta : String = "CONTRATOSCONSULTA"
Es similar al anterior, sólo que aquí existen más datos para cuestiones de búsqueda

- vistaContra : String = "CCONTRA"
Campo de la clave de los contratos

- dataTDevenga : String = "DEVENGA"
Nombre del Data Table que indica si un pago es devengable

- vistaDev : String = "DEVENGA"
Campo que indica el si devenga o no un pago

- dataTAcreedores : String = "ACREEDORES"
Nombre del Data Table de Acreedores

- vistaAcree : String = "DACRE"
Campo indica la descripción los acreedores.

- valorAcree : String = "CACRSUC"
Campo indica la clave de los acreedores.

- dataTMoneda : String = "MONEDA"
Nombre del Data Table de tipos de monedas.

- vistaMon : String = "DMONEDA"
Campo indica la descripción de las monedas.

- valorMon : String = "CMONEDA"
Campo indica la clave de las monedas.

- dataTTipoDeuda : String = "TIPODEUDA"
Nombre del Data Table de los tipos de deuda.

- vistaDeu : String = "DDEUDA"
Campo indica la descripción de los tipos de deuda.

- valorDeu : String = "CDEUDA"
Campo indica la clave de los tipos de deuda.

- dataTTipoTasa : String = "TIPOTASA"
Nombre del Data Table de los tipos de tasa.

- vistaTipoTas : String = "DTASA"
Campo indica la descripción de los tipos de tasa.

- valorTipoTas : String = "CTASA"
Campo indica la clave de los tipos de tasa.

- dataTInstInterm : String = "INSTINTERM"
Nombre del Data Table de los tipos de institución intermediaria.

- vistaInst : String = "DINSTINTERM"
Campo indica la descripción tipos de institución intermediaria.

- valorInst : String = "CINSTINTERM"
Campo indica la clave de los tipos de institución intermediaria.

- dataTBase : String = "BASE"
Nombre del Data Table de las diferentes bases.

- valorBas : String = "CBASE"
Campo que indica la clave de las diferentes bases

- dataTPrepagable : String = "PREPAGABLE"
Nombre del Data Table cuya información indica si es prepagable o no.

- vistaPrepag : String = "DPREPAGABLE"
Campo que indica la descripción de si es prepagable o no.

- valorPrepag : String = "CPREPAGABLE"
Campo que indica la clave de si es prepagable o no.

- dataTTipoPlazo : String = "TIPOPLAZO"
Nombre del Data Table cuya información indica el tipo de plazo

- vistaTipoPlaz : String = "DTIPOPLAZO"
Campo que indica la descripción del tipo de plazo.

- valorTipoPlaz : String = "CTIPOPLAZO"
Campo que indica la clave del tipo de plazo.

- dataTTipoDeudaPresup : String = "TIPODEUDAPRESUP"
Nombre del Data Table cuya información contiene los tipos de Deuda Presupuestal

- vistaTipoDeudaPres : String = "DTIPODEUDAPRESUP"
Campo que indica la descripción del tipo de Deuda Presupuestal

- valorTipoDeudaPres : String = "CTIPODEUDAPRESUP"
Campo que indica la clave del tipo de Deuda Presupuestal

- dataTTipoCredPresup : String = "TIPOCREDPRESUP"
Nombre del Data Table cuya información contiene los tipos de Crédito Presupuestal

- vistaTipoCredPres : String = "DCREDI"
Campo que indica la descripción del tipo de Crédito Presupuestal

- valorTipoCredPres : String = "CCREDI"
Campo que indica la clave del tipo de Crédito Presupuestal

Métodos

`+ New()`
Constructor de la clase.

`+ vistaContratos() : String`
Regresa el campo de las claves de los contratos.

`+ vistaAcreedores() : String`
Regresa el campo de la descripción de los acreedores.

`+ valorAcreedores() : String`
Regresa el campo de la clave de los acreedores.

`+ vistaMoneda() : String`
Regresa el campo de la descripción de las monedas.

`+ valorMoneda() : String`
Regresa el campo de la clave de las monedas.

`+ vistaDeuda() : String`
Regresa el campo de la descripción de los tipos de deuda.

`+ valorDeuda() : String`
Regresa el campo de la clave de los tipos de deuda.

`+ vistaTipoTasa() : String`
Regresa el campo de la descripción de los tipos de tasas.

`+ valorTipoTasa() : String`
Regresa el campo de la clave de los tipos de tasas.

`+ vistaInstInterm() : String`
Regresa el campo de la descripción de los tipos de institución intermediaria.

`+ valorInstInterm() : String`
Regresa el campo de la clave de la descripción de los tipos de institución intermediaria.

`+ valorBase() : String`
Regresa el campo de los tipos de bases.

`+ vistaPrepagable() : String`
Regresa el campo que indica la descripción de si es prepagable o no.

`+ valorPrepagable() : String`
Regresa el campo que indica la clave de si es prepagable o no.

`+ vistaTipoPlazo() : String`
Regresa el campo de la descripción de los tipos de plazos.

`+ valorTipoPlazo() : String`
Regresa el campo de la clave de los tipos de plazos.

`+ vistaTipoDeudaPresup() : String`
Regresa el campo de la descripción de los tipos de deuda presupuestal.

`+ valorTipoDeudaPresup() : String`
Regresa el campo de la clave de los tipos de deuda presupuestal.

`+ vistaTipoCredPresup() : String`
Regresa el campo de la descripción de los tipos de crédito presupuestal.

`+ valorTipoCredPresup() : String`
Regresa el campo de la clave de los tipos de crédito presupuestal.

`+ vistaDevenga() : String`
Regresa el campo de la descripción de si devenga o no.

`+ dataTableContratosConsulta() : DataTable`
Regresa el Data Table con la lista de contratos e información adicional para consulta.

`+ dataTableContratos() : DataTable`
Regresa el Data Table con la lista de contratos.

`+ dataTableAcreedores() : DataTable`
Regresa el Data Table con la lista de acreedores.

`+ dataTableMoneda() : DataTable`
Regresa el Data Table con la lista de monedas.

`+ dataTableDeuda() : DataTable`
Regresa el Data Table con la lista de tipos de deuda.

`+ dataTableDevenga() : DataTable`
Regresa el Data Table con la lista de opciones de si devenga o no.

`+ dataTableTipoTasa() : DataTable`
Regresa el Data Table con la lista de tipos de tasa.

`+ dataTableInstInterm() : DataTable`
Regresa el Data Table con la lista de instituciones intermediarias.

`+ dataTableBase() : DataTable`
Regresa el Data Table con la lista de tipos de bases.

`+ dataTablePrepagable() : DataTable`
Regresa el Data Table con la lista de opciones de si es prepagable o no.

`+ dataTableTipoPlazo() : DataTable`
Regresa el Data Table con la lista de tipos de plazos.

`+ dataTableTipoDeudaPresup() : DataTable`
Regresa el Data Table con la lista de tipos de deuda presupuestal.

`+ dataTableTipoCredPresup() : DataTable`
Regresa el Data Table con la lista de tipos de crédito presupuestal.

Clase Fecha

Métodos

`+ <<Shared>> parseaFecha(fecha : String) : Date`
Regresa a fecha una cadena.
`fecha:` La cadena a convertir, será en formato ddmmyyyy.

`+ <<Shared>> ultimoCierre() : Date`
Nos regresa la fecha del último mes cerrado.

`+ <<Shared>> fechaCerrada(fecha : String) : Boolean`
Nos dice si una fecha ya cerró o no.
`Fecha:` Cadena en formato ddmmyyyy.

`+ <<Shared>> fechaCerrada(fecha : Date) : Boolean`
Nos dice si una fecha ya cerró o no.

`Fecha:` La fecha a revisar.

`+ <<Shared>> esMayorA(fecha1 : Date, fecha2 : Date) : Boolean`
Nos dice si una fecha es mayor a otra. Compara `fecha1` con `fecha2`.

`+ <<Shared>> esMenorA(fecha1 : Date, fecha2 : Date) : Boolean`
Nos dice si una fecha es menor a otra. Compara `fecha1` con `fecha2`.

`+ <<Shared>> esMenorOIgual(fecha1 : Date, fecha2 : Date) : Boolean`
Nos dice si una fecha es menor o igual a otra. Compara `fecha1` con `fecha2`.

`+ <<Shared>> esMayorOIgual(fecha1 : Date, fecha2 : Date) : Boolean`
Nos dice si una fecha es mayor o igual a otra. Compara `fecha1` con `fecha2`.

`+ <<Shared>> toDate(fecha : Date) : String`
Convierte a cadena un objeto Date.

Clase Progreso

Atributos
`- components : System.ComponentModel.IContainer`
Contenedor de los controles.

`- avance : System.Windows.Forms.ProgressBar`
La barra que indica al usuario el progreso

Métodos
`+ New()`
Constructor de la clase

`- InitializeComponent()`
Coloca y acomoda todos los componentes en la ventana del progreso.

`# Dispose(disposing : Boolean)`
Destruye al objeto

Clase Autenticacion

Atributos

`- components : System.ComponentModel.IContainer`
Contenedor de los controles.

`- lblUsuario : System.Windows.Forms.Label`
Etiqueta con leyenda "Usuario:"

`- lblContrasena : System.Windows.Forms.Label`
Etiqueta con leyenda "Contraseña:"

`- txtUsuario : System.Windows.Forms.TextBox`
Caja de texto donde se ingresa el usuario.

`- txtContrasena : System.Windows.Forms.TextBox`
Caja de texto donde se ingresa la contraseña.

`- btnAceptar : System.Windows.Forms.Button`
Botón de Aceptar.

`- btnCancelar : System.Windows.Forms.Button`
Botón de Cancelar.

- stBienvenida : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde la da la bienvenida al usuario.

- stCandado : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde aparece el icono de un candado.

- barEstado : System.Windows.Forms.StatusBar
Barra de estado la cual aparece en la parte inferior de la ventana y contiene paneles de información.

- picGrafica : System.Windows.Forms.PictureBox
Espacio para colocar una imagen.

- picLogo : System.Windows.Forms.PictureBox
Espacio para colocar una imagen.

- mensaje : Mensaje
Para mandar mensajes al usuario.

- consulta : Consulta
Para hacer consultas a la Base de Datos.

- autenticado : Boolean
Indica si el usuario y contraseña son correctos.

- cancela : Boolean
Indica si fue cancelada la autenticación.

- intentos : Integer = 3
Número de intentos que se le da al usuario para autenticarse correctamente.

Métodos

+ New()
Constructor de la clase

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del progreso.

Dispose(disposing : Boolean)
Destruye al objeto

+ autentica()
Manda la ventana para que se autentica el usuario.

+ cancelado() : Bolean
Indica si fue cancelada la autenticación.

+ autorizado() : Bolean
Indica si el usuario es correcto o no.

Eventos

- Autenticacion_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- btnAceptar_Click(sender : System.Object, e : System.EventArgs) => btnAceptar.Click
Se dispara cuando el usuario oprime el botón de aceptar.

- Autenticacion_Closing(sender : Object, e : System.ComponentModel.CancelEventArgs) => MyBase.Closing

Se dispara cuando se está cerrando la ventana.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando el usuario oprime el botón de cancelar.

Clase Catalogo

Atributos

- components : System.ComponentModel.IContainer
Contenedor de los controles.

btnSalir : System.Windows.Forms.Button
Botón para cerrar la ventana del catálogo.
btnExcel : System.Windows.Forms.Button
Botón para enviar la información consultada a una hoja de Microsoft Excel.

btnDeshacer : System.Windows.Forms.Button
Botón que se utiliza para deshacer los cambios hechos antes de salvar los cambios realizados.

btnBorrar : System.Windows.Forms.Button
Botón que se utiliza para borrar un registro en particular.

btnActualizar : System.Windows.Forms.Button
Botón que al oprimirse se guardan los cambios en la Base de Datos.

btnInsertar : System.Windows.Forms.Button
Botón que se utilizar para permitir la inserción de un nuevo registro.

dataGrid : System.Windows.Forms.DataGrid
Es el Data Grid donde se hacen las consultas a la Base de Datos y se ven reflejados los cambios. Un Data Grid es una tabla con datos.

barEstado : System.Windows.Forms.StatusBar
Barra de estado la cual aparece en la parte inferior de la ventana y contiene paneles de información.

stRegistros : System.Windows.Forms.StatusBarPanel
Panel que indica el número total de registros consultados y el número de registro en que el usuario tiene el cursor.

stCandado : System.Windows.Forms.StatusBarPanel
Panel que indica con icono si el registro que tiene el cursor es modificable o no.

btnConsultar : System.Windows.Forms.Button
Botón que sirve para consultar la información requerida por el usuario.

- tooltips : System.Windows.Forms.ToolTip
Tooltips de los botones. Un tooltip es una leyenda que aparece cuando el usuario coloca el puntero del ratón sobre algún control.

- buscaImagen : System.Windows.Forms.OpenFileDialog
Ventana de diálogo donde se puede elegir algún archivo, en éste caso se elige una imagen para colocarla de fondo en la ventana.
- mnuFondo : System.Windows.Forms.ContextMenu
Menú contextual de la ventana donde el usuario elige la opcion de personalizar la imagen del fondo. Un menú contextual es un menú flotante que se activa oprimiendo el botón derecho del ratón.

- mnuImagen : System.Windows.Forms.MenuItem
Opción del menu contextual mnuFondo.

txtInicial : System.Windows.Forms.TextBox
Es la caja de texto donde el usuario comienza a insertar un nuevo registro.

DataGridTableStyle1 : System.Windows.Forms.DataGridTableStyle
Es el objeto que crea lo estilos del DataGrid. Los estilos son la apariencia física que muestra el Data Grid.

btnAgregar : System.Windows.Forms.Button
Botón que inserta al Data Grid el nuevo registro ya editado.

dataSetC : APLICACIONES.DataSetCatalogos
Es el Data Set donde se encuentran las tablas de los Catalogos. Recordemos un Data Set es un conjunto de Data Tables, y un Data Table es una tabla en memoria donde se mapea con una tabla en particular de la Base de Datos.

conexion : System.Data.OleDb.OleDbConnection
La conexión a la Base de Datos.

btnCancelar : System.Windows.Forms.Button
Botón que hace que se cancele la edición del nuevo registro a insertar.

stCelda : System.Windows.Forms.StatusBarPanel
Panel donde se muestra la información detallada sobre el campo del registro en que está el cursor.

mensaje : Mensaje
Para mandar mensajes al usuario.

consulta : Consulta
Se requiere de éste objeto para poder hacer consultas de ciertos datos.

consultaSID : ConsultaDatosSID
Para poder traer ciertos datos requeridos del Sistema de Información de Deuda.

mensajeBorrar : String
Cada que el usuario necesita borrar un registro, se le manda un mensaje de confirmación el cual tiene que ser asignado a éste atributo.

catalogo : String
Nombre del catálogo en el que se está ejecutando.

adaptadorCatalogo : OleDb.OleDbDataAdapter
Este es el adaptador de datos que permite las consultas y actualizaciones a la Base de Datos.

tablaDataSet : String
Nombre del Data Table con el que se está operando.

filaActual : Integer
Aquí se guarda el indice de fila del Data Grid donde se encuentra el cursor.

columnaActual : Integer
Aquí se guarda el indice de la columna del Data Grid donde se encuentra el cursor.

datoActual : Object
Aquí se almacena el dato que se encuentra en la celda del Data Grid donde se encuentra el cursor.

filaAnterior : Integer
Aquí se guarda el indice de la fila del Data Grid donde estuvo el cursor antes de que estuviera donde se encuentra actualmente.

columnaAnterior : Integer
Aquí se guarda el indice de la columna del Data Grid donde estuvo el cursor antes de que estuviera donde se encuentra actualmente.

datoAnterior : Object
Aquí se guarda el dato de la celda del Data Grid donde estuvo el cursor antes de que estuviera donde se encuentra actualmente.

cambioDatoCelda : Bolean
Nos dice se cambio el dato de la celda del Data Grid.

- progreso : Progreso
Se utiliza para indicar el avance del proceso de enviar los datos consultados a una hoja de Microsoft Excel.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del progreso.

Dispose(disposing : Boolean)
Destruye al objeto

- cargaDataSet()
Prepara el Data Set actual para ser llenado el respectivo Data Table.

- llenaDataSet(dataSet : DataSetCatalogos)
Llena el respectivo Data Table con los datos consultados de la Base de Datos.

- localizaCambiosDataSet()
Si hubo cambios en el Data Set, los registra para posteriormente actualizar la Base de Datos.

- actualizaDataSet(filasCambiadas : DataSetCatalogos)
Actualiza los cambios en la Base de Datos.

Eventos

- Catalogo_Load(sender : System.Object, e : System.EventArgs) =>
MyBase.Load
Se dispara al inicializar la ventana.

btnConsultar_Click(sender : System.Object, e : System.EventArgs) =>
btnConsultar.Click
Se dispara al oprimir el botón de consultar.

- btnActualizar_Click(sender : System.Object, e : System.EventArgs) =>
btnActualizar.Click
Se dispara al oprimir el botón de actualizar.

- btnDeshacer_Click(sender : System.Object, e : System.EventArgs) =>
btnDeshacer.Click
Se dispara al oprimir el botón de deshacer.

- btnExcel_Click(sender : System.Object, e : System.EventArgs) =>
btnExcel.Click
Se dispara al oprimir el botón de

- btnSalir_Click(sender : System.Object, e : System.EventArgs) =>
btnSalir.Click
Se dispara al oprimir el botón de salir.

- mnuImagen_Click(sender : System.Object, e : System.EventArgs) => mnuImagen.Click
Se dispara al elegir personalizar imagen del menú contextual.

- Catalogo_Closing(sender : Object, e : System.ComponentModel.CancelEventArgs) => MyBase.Closing
Se dispara cuando la ventana se está cerrando.

- btnBorrar_Click(sender : System.Object, e : System.EventArgs) => btnBorrar.Click
Se dispara cuando se oprime el botón borrar.

- btnInsertar_Click(sender : System.Object, e : System.EventArgs) => btnInsertar.Click
Se dispara cuando se oprime el botón de insertar nuevo registro.

- btnAgregar_Click(sender : System.Object, e : System.EventArgs) => btnAgregar.Click
Se dispara cuando se oprime el botón de agregar registro.

- dataGrid_CurrentCellChanged(sender : Object, e : System.EventArgs) => DataGrid.CurrentCellChanged
Se dispara cuando se hace cambio de celda en el Data Grid.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando se oprime el botón de cancelar edición de nuevo registro.

Clase Contratos

Atributos

- components : System.ComponentModel.IContainer
Contenedor de los controles.

- cnaContrato : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Contrato. Este objeto contiene las propiedades necesarias para mostrar los estilos de dicha columna. Los estilos de columna se refieren a la alineación, el color del encabezado, el tipo de letra, el tamaño, etc.

- cnaClaveCif : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Clave Cif.

- cnaAcreedor : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Acreedor.

- cnaMoneda : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Moneda.

- cnaTipoDeuda : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Tipo Deuda.

- cnaTipoTasa : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Tipo Tipo.

- cnaInstInterm : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid con encabezdo Inst. Interm.

- cnaSobreTasa : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Sobre Tasa.

- cnaNumAutHacienda : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Num. Aut. Hacienda.

- cnaFechaFirma : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Fecha Firma.

- cnaBase : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Base.

- cnaNumRegObligaciones : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Num. Reg. Obligaciones.

- cnaFechaRegistro : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Fecha Registro.

- cnaPrepagable : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Prepagable.

- cnaTipoPlazo : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Tipo Plazo.

- cnaTipoDeudaPresup : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Tipo Deuda Presup.

- cnaTipoCredPresup : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Tipo Cred Presup.

- cboConsulta : System.Windows.Forms.ComboBox
ComboBox que permite delimitar las consultas.
- cboAcreedores : System.Windows.Forms.ComboBox
ComboBox para editar el campo Acreedor.

- gbxConsulta : System.Windows.Forms.GroupBox
Marco que encierra al ComboBox usado para delimitar las consultas.

- lblAcreedor : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Acreedor"

- lblMoneda : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Moneda"

- lblTipoTasa : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo Tasa"

- lblInstInterm : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Inst. Interm".

- lblClaveCIF : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Clave Cif".

- lblTipoDeuda : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo Deuda"

- lblSobreTasa : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Sobre Tasa"

- lblBase : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Base"

- gbxSHCP : System.Windows.Forms.GroupBox
Marco que encierra a la información relacionada con la Secretaría de Hacienda.

- dtpFechaRegistro : System.Windows.Forms.DateTimePicker
Date Time Picker para editar la Fecha de Registro.

- txtNumRegObligaciones : System.Windows.Forms.TextBox
Caja de Texto para editar el campo Num. Reg. Obligaciones.

- lblFechaRegistro : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Fecha Registro"

- lblFechaFirma : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Fecha Firma"

- lblNumRegObligaciones : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Num. Reg. Obligaciones"

- lblNumAutHacienda : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Num. Aut. Hacienda"

- txtNumAutHacienda : System.Windows.Forms.TextBox
Caja de Texto para editar el Num. Aut. Hacienda.

- dtpFechaFirma : System.Windows.Forms.DateTimePicker
Date Time Picker para seleccionar la Fecha de Firma.

- txtClaveCIF : System.Windows.Forms.TextBox
Etiqueta cuya leyenda es "Clave Cif"

- cboMoneda : System.Windows.Forms.ComboBox
ComboBox para editar el tipo de Moneda.

- cboTipoDeuda : System.Windows.Forms.ComboBox
ComboBox para editar el tipo de deuda.

- cboTipoTasa : System.Windows.Forms.ComboBox
ComboBox para editar el campo Tipo Tasa.

- txtSobreTasa : System.Windows.Forms.TextBox
Caja de texto para editar el campo Sobre Tasa

- cboBase : System.Windows.Forms.ComboBox
ComboBox para editar el campo Base.

- cboInstInterm : System.Windows.Forms.ComboBox
ComboBox para editar el campo Inst. Interm.

- lblPrepagable : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Prepagable".

- lblTipoPlazo : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo Plazo".

- lblTipoDeudaPresup : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo Deuda Presup".

- lblTipoCredPresup : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo Cred. Presup."

- cboPrepagable : System.Windows.Forms.ComboBox
ComboBox para editar el campo Prepagable.

- cboTipoPlazo : System.Windows.Forms.ComboBox
ComboBox para editar el campo Tipo Plazo.

- cboTipoDeudaPresup : System.Windows.Forms.ComboBox
ComboBox para editar el campo Tipo Deuda Presup.

- cboTipoCredPresup : System.Windows.Forms.ComboBox
ComboBox para editar el campo Tipo Cred. Presup.

- adaptador : System.Data.OleDb.OleDbDataAdapter
Adaptador de datos para consultar y actualizar la tabla correspondiente en la Base de Datos.

- OleDbInsertCommand1 : System.Data.OleDb.OleDbCommand
Comando para la inserción a la tabla de la Base de Datos.

- OleDbUpdateCommand1 : System.Data.OleDb.OleDbCommand
Comando para la actualización a la tabla de la Base de Datos.

- OleDbDeleteCommand1 : System.Data.OleDb.OleDbCommand
Comando para el borrado de registros en tabla de la Base de Datos.

- commSelContratos : System.Data.OleDb.OleDbCommand
Comando para la consulta a la tabla de la Base de Datos.

- lblContrato : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Contrato"

- txtFechaFirma : System.Windows.Forms.Label
Etiqueta utilizada para ligar el dtpFechaFirma al Campo Fecha Firma

- txtFechaRegistro : System.Windows.Forms.Label
Etiqueta utilizada para ligar el dtpFechaRegistro al Campo Fecha Registro

- consultaContratos : String = "SELECT CCONTRA, CAUTCIF, CACRSUC, CMONEDA, TDEUDA, TTASA, VSTASA, CINTINT, NBASE, NAUTHDA, FFICONT, NROBHDA, FREGHDA, IPREPAG, TPLAZO, TDEUDAP, CCREDIP FROM R018"
Sentencia SQL para consultar los contratos existentes.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del Catálogo de Contratos.

Dispose(disposing : Boolean)
Destruye al objeto.

- llenaCombos()
Se llenan los ComboBox con su información respectiva y los liga al Data Table correspondiente.

- validaModificaciones()
Antes de insertar un registro previamente editado, se revisa que sea correcto.

- muestraEstados()
Cada que se hace un cambio de celda, se tiene que actualizar la información mostrada en la barra de estado.

- desactivaModificadores()
Cuando el cursor está en un registro que no es actualizable, los ComboBox, Date Time pickers y Cajas de texto se desactivan para no permitir al usuario su modificación.

- activaModificadores()
Cuando el cursor está en un registro que es actualizable, los ComboBox, Date Time pickers y Cajas de texto se activan para permitir al usuario su modificación.

- validaCampos() : Boolean
Revisa que el campo editado para la inserción esté correcto.

Eventos
- Contratos_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- Contratos_Closed(sender : Object, e : System.EventArgs) => MyBase.Closed
Se dispara cuando la ventana ya fue cerrada.

- cboConsulta_ValueMemberChanged(sender : Object, e : System.EventArgs) => cboConsulta.TextChanged, cboConsulta.SelectedValueChanged
Se dispara cuando el dato seleccionado del ComboBox para la consulta es cambiado.

- cboConsulta_LostFocus(sender : Object, e : System.EventArgs) => cboConsulta.LostFocus
Se dispara cuando el ComboBox de la consulta pierde el cursor.

- cboConsulta_KeyDown(sender : Object, e : System.Windows.Forms.KeyEventArgs) => cboConsulta.KeyDown
Se dispara cuando sobre el ComboBox se oprime alguna tecla del teclado.

- dataGrid_CurrentCellChanged(sender : Object, e : System.EventArgs) => dataGrid.CurrentCellChanged
Se dispara cuando el cursor cambia de celda en el Data Grid.

- btnInsertar_Click(sender : System.Object, e : System.EventArgs) => btnInsertar.Click
Se dispara cuando se oprime el botón de insertar nuevo registro.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando se cancela la inserción de un nuevo registro.

- btnAgregar_Click(sender : System.Object, e : System.EventArgs) => btnAgregar.Click
Se dispara cuando se oprime el botón para agregar el nuevo registro ya editado.

- dtpFechaFirma_ValueChanged(sender : Object, e : System.EventArgs) => dtpFechaFirma.ValueChanged
Se dispara cuando se cambia el valor del dtpFechaFirma

- dtpFechaRegistro_ValueChanged(sender : Object, e : System.EventArgs) => dtpFechaRegistro.ValueChanged
Se dispara cuando se cambia el valor del dtpValueChanged

Clase BancosHouston

Atributos

- components : System.ComponentModel.IContainer
Contenedor de los controles.

- cnaContrato : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Contrato.

- cnaClaveBanco : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Clave Banco.

- cnaNombre : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Institución de Financiamiento.

- cnaPlazo : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Plazo

- cnaMoneda : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Moneda.

- cnaLinea : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Linea.

- adaptador : System.Data.OleDb.OleDbDataAdapter
Adaptador de datos para consultar y actualizar la tabla correspondiente en la Base de Datos.

- OleDbInsertCommand1 : System.Data.OleDb.OleDbCommand
Comando para la inserción a la tabla de la Base de Datos.

- OleDbUpdateCommand1 : System.Data.OleDb.OleDbCommand
Comando para la actualización a la tabla de la Base de Datos.

- OleDbDeleteCommand1 : System.Data.OleDb.OleDbCommand
Comando para el borrado de registros en la tabla de la Base de Datos.

- gbxConsulta : System.Windows.Forms.GroupBox
Marco que encierra al ComboBox usado para delimitar las consultas.

- cboConsulta : System.Windows.Forms.ComboBox
ComboBox que permite delimitar las consultas

- commSelContratos : System.Data.OleDb.OleDbCommand
Comando para la actualización a la tabla de la Base de Datos.

- lblContrato : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Contrato"

- lblLinea : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Linea"

- cboTipoPlazo : System.Windows.Forms.ComboBox
ComboBox para editar el campo Tipo Plazo

- cboMoneda : System.Windows.Forms.ComboBox
ComboBox para editar el campo Moneda.

- txtLinea : System.Windows.Forms.TextBox
Caja de texto para editar el campo Linea.

- lblCveBanco : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Clave Banco".

- txtClaveBco : System.Windows.Forms.TextBox
Caja de texto paara editar el campo Clave Banco.

- lblPlazo : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Plazo".

- lblMoneda : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Moneda".

- cboContratos : System.Windows.Forms.ComboBox
ComboBox para editar el campo contrato.

- consultaContratos : String = "SELECT CLINEA, CMONEDA, PLAZO, NOMBRE, CVEBCO, CCONTRA FROM SID.R055"
Sentencia SQL para la consulta de todos los registros de la tabla correspondiente.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del catálogo.

Dispose(disposing : Boolean)
Destruye al objeto.

- llenaCombos()
Se llenan los combos con la información correspondiente.

- validaCampos() : Boolean
Revisa que el campo editado para la inserción esté correcto.

Eventos

- BancosHouston_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- dataGrid_CurrentCellChanged(sender : System.Object, e : System.EventArgs) => dataGrid.CurrentCellChanged
Se dispara cuando se hace cambio de celda en el Data Grid.

- cboConsulta_ValueMemberChanged(sender : Object, e : System.EventArgs)
=> cboConsulta.TextChanged, cboConsulta.SelectedValueChanged
Se dispara cuando cambia la información seleccionada en el ComboBox de para las consultas.

- cboConsulta_LostFocus(sender : Object, e : System.EventArgs) => cboConsulta.LostFocus
Se dispara cuando el ComboBox de la consulta pierde el cursor.

- cboConsulta_KeyDown(sender : Object, e : System.Windows.Forms.KeyEventArgs) => cboConsulta.KeyDown
Se dispara cuando sobre el ComboBox se oprime alguna tecla del teclado.

- BancosHouston_Closed(sender : Object, e : System.EventArgs) => MyBase.Closed
Se dispara cuando la ventana es cerrada.

- btnInsertar_Click(sender : System.Object, e : System.EventArgs) => btnInsertar.Click
Se dispara cuando se oprime el botón de insertar nuevo registro.

- btnAgregar_Click(sender : System.Object, e : System.EventArgs) => btnAgregar.Click
Se dispara cuando se oprime el botón de agregar el registro ya editado.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando se oprime el botón de cancelar la inserción del nuevo registro.

Clase ConceptosPago

Atributos

- components : System.ComponentModel.IContainer
contenedor de los controles

- cnaClavePago : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Clave Pago

- cnaDescripcionPago : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Descripción Pago

- cnaTipoMovimiento : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo tipo de movimiento

- cnaDevenga : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Devenga

- adaptador : System.Data.OleDb.OleDbDataAdapter
El adaptador de datos para consultar y actualizar la tabla correspondiente en la Base de Datos.

- OleDbSelectCommand1 : System.Data.OleDb.OleDbCommand
Comando para la consulta a la tabla de la Base de Datos.

- OleDbInsertCommand1 : System.Data.OleDb.OleDbCommand
Comando para la inserción a la tabla de la Base de Datos.

- OleDbUpdateCommand1 : System.Data.OleDb.OleDbCommand
Comando para la actualización a la tabla de la Base de Datos.

- OleDbDeleteCommand1 : System.Data.OleDb.OleDbCommand
Comando para el borrado a la tabla de la Base de Datos.

- lblClave : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Clave".

- lblDescripcion : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Descripción".

- txtDescripcion : System.Windows.Forms.TextBox
Caja de texto para la edición del campo Descripción.

- txtMovimiento : System.Windows.Forms.TextBox
Caja de texto para la edición del campo Tipo de Movimiento.

- lblMovimiento : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo de Movimiento"

- lblDevenga : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Devenga"

- cboDevenga : System.Windows.Forms.ComboBox
ComboBox para la edición del campo Devenga.

Métodos

`+ New()`
Constructor de la clase.

`- InitializeComponent()`
Coloca y acomoda todos los componentes en la ventana.

`# Dispose(disposing : Boolean)`
Destruye al objeto.

`- lenaCombos()`
Se llenan los ComboBox con su información respectiva y los liga al Data Table correspondiente.

`- validaCampos() : Boolean`
Revisa que el campo editado para la inserción esté correcto.

Eventos

`- ConceptosPago_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load`
Se dispara cuando la ventana se está iniciando

`- dataGrid_CurrentCellChanged(sender : System.Object, e : System.EventArgs) => dataGrid.CurrentCellChanged`
Se dispara cuando se hace cambio de celda en el Data Grid

`- ConceptosPago_Closed(sender : Object, e : System.EventArgs) => MyBase.Closed`
Se dispara cuando la ventana ya fue cerrada

`- btnAgregar_Click(sender : System.Object, e : System.EventArgs) => btnAgregar.Click`
Se dispara cuando el botón de agregar es oprimido

Clase CambioCierre

Atributos

`- components : System.ComponentModel.IContainer`
Contenedor de los controles.

`- cnaFechaCambio : System.Windows.Forms.DataGridTextBoxColumn`
Columna del Data Grid que se mapea con el campo Fecha de Cambio.

`- cnaMoneda : System.Windows.Forms.DataGridTextBoxColumn`
Columna del Data Grid que se mapea con el campo Moneda.

`- cnaCambioMN : System.Windows.Forms.DataGridTextBoxColumn`
Columna del Data Grid que se mapea con el campo Tipo de Cambio Moneda Nacional

`- cnaCambioDolar : System.Windows.Forms.DataGridTextBoxColumn`
Columna del Data Grid que se mapea con el campo Tipo de Cambio Dolar

`- adaptador : System.Data.OleDb.OleDbDataAdapter`
Adaptador de datos para consultar y actualizar la tabla correspondiente en la Base de Datos.

`- OleDbSelectCommand1 : System.Data.OleDb.OleDbCommand`
Comando para la consulta a la tabla de la Base de Datos.

`- OleDbInsertCommand1 : System.Data.OleDb.OleDbCommand`
Comando para la inserción a la tabla de la Base de Datos.

- OleDbUpdateCommand1 : System.Data.OleDb.OleDbCommand
Comando para la actualización a la tabla de la Base de Datos.

- OleDbDeleteCommand1 : System.Data.OleDb.OleDbCommand
Comando para el borrado a la tabla de la Base de Datos.
- lblFecha : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Fecha".

- lblMoneda : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Moneda".

- lblCambio : System.Windows.Forms.Label
Etiqueta cuya leyenda es "Tipo de Cambio M.N.".

- cboMoneda : System.Windows.Forms.TextBox
ComboBox para editar el tipo de Moneda

- dtpFecha : System.Windows.Forms.TextBox
Date Time Picker para editar la Fecha

- txtCambio : System.Windows.Forms.TextBox
Etiqueta para editar el Tipo de Cambio.

Métodos

- llenaCombos()
Se llenan los ComboBox con su información respectiva y los liga al Data Table correspondiente.

- calculaDolar()
Calcula los tipos de cambio consultados en Dólares U.S.A.

- validaCampos() : Boolean
Revisa que el campo editado para la inserción esté correcto.

- muestraEstados()
Cada que se hace un cambio de celda, se tiene que actualizar la información mostrada en la barra de estado.

Eventos

- CambioCierre_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- dataGrid_CurrentCellChanged(sender : Object, e : System.EventArgs) => dataGrid.CurrentCellChanged
Se dispara cuando el cursor cambia de celda en el Data Grid.

- CambioCierre_Closed(sender : Object, e : System.EventArgs) => MyBase.Closed
Se dispara cuando la ventana es cerrada.

- btnConsultarCambios_Click(sender : System.Object, e : System.EventArgs) => btnConsultar.Click
Se dispara cuando se oprime el botón de consultar.

- btnAgregar_Click(sender : System.Object, e : System.EventArgs) => btnAgregar.Click

Se dispara cuando se oprime el botón de agregar registro.

- btnCancelar_Click(sender : System.Object, e : System.EventArgs) => btnCancelar.Click
Se dispara cuando se oprime el botón de cancelar edición de nuevo registro.

Clase AsignacionCreditos

Atributos

- components : System.ComponentModel.IContainer
Contenedor de los controles.

- cnaContrato : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Contrato.

- cnaDesembolso : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Desembolso.

- cnaExpediente : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Expediente.

- cnaMoneda : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Moneda.

- cnaCorporativo : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Coorporativo.

- cnaExploracion : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Exploración.

- cnaRefinacion : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Refinación.

- cnaGasPetroquimicaBasica : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Gas y Petroquímica Básica.

- cnaGasPetroquimicaSecundaria : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Gas y Petroquímica Secundaria.

- cnaDeudaCapitalizable : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Deuda Capitalizable.

- cnaIndefinido : System.Windows.Forms.DataGridTextBoxColumn
Columna del Data Grid que se mapea con el campo Indefinido.

- adaptador : System.Data.OleDb.OleDbDataAdapter
Adaptador de datos para consultar y actualizar la tabla correspondiente en la Base de Datos.

- gbxConsulta : System.Windows.Forms.GroupBox
Marco que encierra al ComboBox usado para delimitar las consultas.

- cboConsulta : System.Windows.Forms.ComboBox
ComboBox que permite delimitar las consultas.

- commSelContratos : System.Data.OleDb.OleDbCommand
Comando para la consulta a la tabla de la Base de Datos.

- consultaContratos : String = "SELECT CLINEA, CMONEDA, PLAZO, NOMBRE, CVEBCO, CCONTRA FROM SID.R055"
Sentencia SQL para la consulta de datos.

Métodos

`+ New()`
Constructor de la clase

`- InitializeComponent()`
Coloca y acomoda todos los componentes en la ventana del progreso.

`# Dispose(disposing : Boolean)`
Destruye al objeto

`- llenaCombos()`
Se llenan los ComboBox con su información respectiva y los liga al Data Table correspondiente.

`- muestraEstados()`
Cada que se hace un cambio de celda, se tiene que actualizar la información mostrada en la barra de estado.

Eventos

`- AsignacionCreditos_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load`
Se dispara cuando la ventana se está iniciando.

`- AsignacionCreditos_Closed(sender : Object, e : System.EventArgs) => MyBase.Closed`
Se dispara cuando la ventana es cerrada.

`- dataGrid_CurrentCellChanged(sender : Object, e : System.EventArgs) => dataGrid.CurrentCellChanged`
Se dispara cuando el cursor cambia de celda en el Data Grid.

`- cboConsulta_ValueMemberChanged(sender : Object, e : System.EventArgs) => cboConsulta.TextChanged, cboConsulta.SelectedValueChanged`
Se dispara cuando cambia la información seleccionada en el ComboBox de para las consultas.

`- cboConsulta_LostFocus(sender : Object, e : System.EventArgs) => cboConsulta.LostFocus`
Se dispara cuando el ComboBox de la consulta pierde el cursor.

`- cboConsulta_KeyDown(sender : Object, e : System.Windows.Forms.KeyEventArgs) => cboConsulta.KeyDown`
Se dispara cuando sobre el ComboBox se oprime alguna tecla del teclado.

Clase PagosNoProgramados

Atributos

`- components : System.ComponentModel.IContainer`
El contenedor de los controles.

`- btnConsultar : System.Windows.Forms.Button`
Botón de Consulta de Datos.

`- btnInsertar : System.Windows.Forms.Button`
Botón de Inserción de Datos

`- lblContrato : System.Windows.Forms.Label`
Etiqueta con leyenda "Contrato".

- dataGridVista : System.Windows.Forms.DataGrid
Data Grid para consulta de pagos por familia.

- dataGridR050 : System.Windows.Forms.DataGrid
Data Grid para la inserción de Datos

- btnSalir : System.Windows.Forms.Button
Botón de salida.

- btnActualizar : System.Windows.Forms.Button
Botón para actualizar en la Base de Datos.

- btnBorrar : System.Windows.Forms.Button
Botón de borrado de registros.

- btnDeshacer : System.Windows.Forms.Button
Botón para deshacer cambios.

- barra : System.Windows.Forms.StatusBar
Barra de estado de la ventana.

- stRegistros : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado que indica el número de registros y el índice del registro donde se encuentra el cursor.

- stFamilia : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado que indica a qué familia pertenece el pago que se está insertando.

- stInstrucciones : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado que muestra mensajes y sugerencias al usuarios.

- stTotalDevengados : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado que indica la suma total de los pagos de la familia actual.

- stCandado : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde aparece el icono de un candado.

- adaptadorVista : System.Data.OleDb.OleDbDataAdapter
Adaptador de datos para consultar la tabla correspondiente en la Base de Datos.

- OleDbSelectCommand1 : System.Data.OleDb.OleDbCommand
Comando para la consulta a la tabla de la Base de Datos.

- conexion : System.Data.OleDb.OleDbConnection
Conexión a la Base de Datos

- adaptadorR050 : System.Data.OleDb.OleDbDataAdapter
Adaptador de datos para actualizar la tabla correspondiente en la Base de Datos.

- OleDbSelectCommand2 : System.Data.OleDb.OleDbCommand
Comando para la consulta a la tabla de la Base de Datos.

- OleDbInsertCommand2 : System.Data.OleDb.OleDbCommand
Comando para la inserción de Datos a la tabla de la Base de Datos.

- OleDbUpdateCommand1 : System.Data.OleDb.OleDbCommand
Comando para la actualización de Datos.

- OleDbDeleteCommand1 : System.Data.OleDb.OleDbCommand
Comando para el borrado de Datos.

- dataSetPVista : DataSetVista
Data Set para la consulta de Datos.

- dataSetPR050 : DataSetR050
Data Set para la inserción de Datos.

- cboContratoConsulta : System.Windows.Forms.ComboBox
ComboBox para editar el campo contrato.

- cboDesembolsoConsulta : System.Windows.Forms.ComboBox
ComboBox para editar el campo Desembolso.
- dtpFechaValorConsulta : System.Windows.Forms.DateTimePicker
Date Time Picker para editar el campo Fecha Valor.

- toolTips : System.Windows.Forms.ToolTip
Tool tips para los botones.

- imagen : System.Windows.Forms.ContextMenu
Menú contextual de la ventana.

- MenuItem1 : System.Windows.Forms.MenuItem
Opción del menú contextual para la elección de la imagen de fondo.

- buscaImagen : System.Windows.Forms.OpenFileDialog
Ventana de diálogo para la elección del archivo de imagen para el fondo.

- cboContratoVista : System.Windows.Forms.ComboBox
ComboBox para delimitar las consultas por contrato.

- btnConsultarVista : System.Windows.Forms.Button
Botón para la consulta de Datos.

- btnExcelVista : System.Windows.Forms.Button
Botón para enviar los datos consultados a una hoja de Excel.

- cboDesembolsoVista : System.Windows.Forms.ComboBox
ComboBox para delimitar la consulta por desembolso.

- lblDia : System.Windows.Forms.Label
Etiqueta con leyenda "Dia"

- cboDia : System.Windows.Forms.ComboBox
ComboBox para delimitar la consulta por el día de la fecha valor.

- lblMes : System.Windows.Forms.Label
Etiqueta con leyenda "Mes"

- cboMes : System.Windows.Forms.ComboBox
ComboBox para delimitar la consulta por el mes de la fecha valor.

- lblAnio : System.Windows.Forms.Label
Etiqueta con leyenda "Año"

- cboAnio : System.Windows.Forms.ComboBox
ComboBox para delimitar la consulta por el año de la fecha valor.

- GroupBox1 : System.Windows.Forms.GroupBox
Marco que encierra los controles de consulta por fecha valor.

- rbValida : System.Windows.Forms.RadioButton
Radio Button que activa las validaciones de la actualización de pagos.

- rbNoValida : System.Windows.Forms.RadioButton
Radio Button que desactiva las validaciones de la actualización de pagos.

- PictureBox1 : System.Windows.Forms.PictureBox
Marco para colocar una imagen con leyenda "Contrato".

- PictureBox2 : System.Windows.Forms.PictureBox
Marco para colocar una imagen con leyenda "Desembolso".

- PictureBox3 : System.Windows.Forms.PictureBox
Marco para colocar una imagen con leyenda "Fecha Valor".

- PictureBox4 : System.Windows.Forms.PictureBox
Marco para colocar una imagen con leyenda "Contrato".

- PictureBox5 : System.Windows.Forms.PictureBox
Marco para colocar una imagen con leyenda "Desembolso".

- prog : Progreso
Se utiliza para indicar el avance del proceso de enviar los datos consultados a una hoja de Microsoft Excel.

- mensaje : Mensaje
Para mandar mensajes al usuario.

- dsCombo : New DataSet
Data Set utilizado para almacenar los datos que llenan los ComboBox.

- dsCboContrato : New DataSet
Data Set utilizado para almacenar los contratos para llenar el ComboBox para la edición del campo contrato.

- dsCboDesembolso : New DataSet
Data Set utilizado para almacenar los desembolsos para llenar el ComboBox para la edición de desembolsos.

- dsCboDesembolsoVista : New DataSet
Data Set utilizado para almacenar los contratos para llenar el ComboBox de la consulta por contrato.

- dsCboDesembolsoDataGrid : New DataSet
Data Set utilizado para almacenar los desembolsos para llenar el ComboBox para la edición del campo contrato.

- estilo : New DataGridTableStyle
Estilos del Data Grid donde se actualizan registros.

- estiloVista : New DataGridTableStyle
Estilos del Data Grid para consulta.

- consulta : New consulta
Para consultar ciertos datos a la Base de Datos.

- cambioConsulta : Boolean
Indica si se ha cambiado algún valor en los controles utilizados para delimitar las consultas.

- fechaCierre : Date
Indica la fecha del último cierre.

- cambioValor : Boolean
Indica si el dtpFechaValor ha cambiado de valor.

- cambioFin : Boolean
Indica si el dtpFechaFin ha cambiado de valor.

- cambioInicio : Boolean
Indica si el dtpFechaInicio ha cambiado de valor.

- dsComboFuera : New DataSet
Data Set utilizado para almacenar los datos que llenan los Combo Box para la consulta por Fecha Valor.

- dsCboContratoVista : New DataSet
Data Set utilizado para almacenar los datos que llena el Combo Box utilizado para delimitar la consulta por contratos.

- cboAcreedor : New ComboBox
Combo Box utilizado para la edición del campo Acreedor.

- cboSitPago : New ComboBox
Combo Box utilizado para la edición del campo Acreedor.

- dtpFechaValor : New DateTimePicker
Date Time Picker utilizado para la edición del campo Fecha Valor.

- dtpFechaInicio : New DateTimePicker
Date Time Picker utilizado para la edición del campo Fecha Inicio.

- dtpFechaFin : New DateTimePicker
Date Time Picker utilizado para la edición del campo Fecha Fin.

- cboConceptoPago : New ComboBox
Combo Box utilizado para la edición del campo Concepto de Pago.

- cboDevenga : New ComboBox
Combo Box utilizado para la edición del campo Devenga.

- cboMoneda : New ComboBox
Combo Box utilizado para la edición del campo Moneda.

- cboFinanciado : New ComboBox
Combo Box utilizado para la edición del campo Financiado.

- cboContrato : New ComboBox
Combo Box utilizado para la edición del campo Contrato.

- cboDesembolso : New ComboBox
Combo Box utilizado para la edición del campo Desembolso.

- dtpFTCambio : New DateTimePicker
Date Time Picker utilizado para la edición del campo Fecha de Tipo de Cambio.

- columnaAnterior : Integer
Indice de la columna anterior del Data Grid a la que actualmente tiene el cursor.

- filaAnterior : Integer
Índice de la fila anterior del Data Grid a la que actualmente tiene el cursor.

- paraGuardar : String
Dato de la celda anterior del Data Grid a la que actualmente tiene el cursor.

- primerCambioCelda : Boolean = True
Indica si es la primera vez que el usuario cambia de Celda en el Data Grid.

- valida : Boolean = True
Indica si están activadas las validaciones o no.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana del catálogo.

Dispose(disposing : Boolean)
Destruye al objeto.

- estilos()
Inicializa los estilos de los Data Grids.

- combosConsulta()
Llena con información los Combos utilizados para delimitar la consulta.

- combosDataGrid()
Llena con información los Combos utilizados para editar información.

- insertaEnFamilia(fechaInicio : Date, fechaFin : Date)
Inserta un pago cuando ya pertenece a alguna familia.
fechaInicio: El dato del campo Fecha Inicio
fechaFin: El dato del campo Fecha Fin

- llenaColumnas()
Hace el llenado de los campos que no están en la Base de Datos. Básicamente son cálculos numéricos.

- validacionGeneral()
Revisa si se puede o no actualizar el registro actual.

- muestraEstados()
Cada que se hace un cambio de celda, se tiene que actualizar la información mostrada en la barra de estado.

- muestraCombos()
Se llenan los ComboBox con su información respectiva y los liga al Data Table correspondiente.

- fechaFinMaxima() : Date
Dada una familia de pagos, regresa el campo Fecha Fin máximo de la familia de pagos.

- fechaFinMinima() : Date
Dada una familia de pagos, regresa el campo Fecha Fin mínima de la familia de pagos.

- fechaInicioMinima() : Date
Dada una familia de pagos, regresa el campo Fecha Inicio mínima de la familia de pagos.

- fechaFinMaximaParcial(fecha1 : Date) : Date
Dada una familia de pagos, regresa el campo Fecha Fin máximo de la familia de pagos sin contar la Fecha se que da como parámetro.

- fechaInicioMaximaParcial(fecha1 : Date) : Date
Dada una familia de pagos, regresa el campo Fecha Inicio máximo de la familia de pagos sin contar la Fecha se que da como parámetro.

- validaNulo()
Revisa si el campo que se está insertando es nulo.

- `validaCampo()`
Revisa si el campo que se está insertando tiene valor correcto.

- `insertaPII()`
Inserta pago que cierra la familia de pagos ya existentes.

- `esPimpon() : Bolean`
Indica si el pago que se está editando contiene valor "PII" en el campo Concepto de Pago.

- `esDIIQueCierra() : Bolean`
Indica si el pago que se inserta cierra la familia de pagos y contiene valor "DII" en el campo Concepto de Pago.

- `esPIIQueCierra() : Bolean`
Indica si el pago que se inserta cierra la familia de pagos y contiene valor "PII" en el campo Concepto de Pago.

Eventos

- `PagosNoProgramados_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load`
Se dispara cuando la ventana se está iniciando.

- `btnInsertar_Click(sender : System.Object, e : System.EventArgs) => btnInsertar.Click`
Se dispara cuando se oprime el botón de insertar registro.

- `btnConsultar_Click(sender : System.Object, e : System.EventArgs) => btnConsultar.Click`
Se dispara cuando se oprime el botón de consultar datos.

- `btnActualizar_Click(sender : System.Object, e : System.EventArgs) => btnActualizar.Click`
Se dispara cuando se oprime el botón de actualizar Base de Datos.

- `btnBorrar_Click(sender : System.Object, e : System.EventArgs) => btnBorrar.Click`
Se dispara cuando se oprime el botón de borrar registro.

- `btnDeshacer_Click(sender : System.Object, e : System.EventArgs) => btnDeshacer.Click`
Se dispara cuando se oprime el botón de deshacer cambios.

- `btnSalir_Click(sender : System.Object, e : System.EventArgs) => btnSalir.Click`
Se dispara cuando se oprime el botón de salir.

- `dataGridR050_CurrentCellChanged(sender : Object, e : System.EventArgs) => dataGridR050.CurrentCellChanged`
Se dispara cuando se cambia de celda en el Data Grid.

- `cboContratoConsulta_SelectedValueChanged(sender : Object, e : System.EventArgs) => cboContratoConsulta.SelectedValueChanged`
Se dispara cuando se cambia el valor del ComboBox usado para delimitar la consulta por contrato.

- `cambioDatosConsulta(sender : Object, e : System.EventArgs) => cboDesembolsoConsulta.SelectedValueChanged, dtpFechaValorConsulta.ValueChanged`
Se dispara cuando se cambia el valor en el ComboBox para la edición de desembolsos o el valor del Date Time Picker para la edición de la fecha valor.

- dataGridR050_Paint(sender : Object, e : System.Windows.Forms.PaintEventArgs) => dataGridR050.Paint
Se dispara cada el Data Grid llama al método Paint.

- cboAcreedor_DropDown(sender : Object, e : System.EventArgs) => cboAcreedor.DropDown
Se dispara cada que se hace DropDown en el ComboBox de acreedores.

- cboAcreedor_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboAcreedor.SelectedIndexChanged
Se dispara cada que se cambia el índice del valor en el ComboBox de acreedores.

- cboSitPago_DropDown(sender : Object, e : System.EventArgs) => cboSitPago.DropDown
Se dispara cada que se hace DropDown en el ComboBox de situación de pago.

- cboSitPago_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboSitPago.SelectedIndexChanged
Se dispara cada que se cambia el índice del valor en el ComboBox de situación de pago.

- cboConceptoPago_DropDown(sender : Object, e : System.EventArgs) => cboConceptoPago.DropDown
Se dispara cada que se hace DropDown en el ComboBox de situación de pago.

- cboConceptoPago_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboConceptoPago.SelectedIndexChanged
Se dispara cada que se cambia el índice del valor en el ComboBox del concepto de pago.

- cboDevenga_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboDevenga.SelectedIndexChanged
Se dispara cada que se cambia el índice del valor en el ComboBox que indica si devenga o no el pago.

- dtpFechaValor_ValueChanged(sender : Object, e : System.EventArgs) => dtpFechaValor.ValueChanged
Se dispara cada que se cambia el valor del dtpFechaValor.

- dtpFechaValor_CloseUp(sender : Object, e : System.EventArgs) => dtpFechaValor.CloseUp
Se dispara cada que se cierran las opciones del dtpFechaValor

- dtpFechaInicio_ValueChanged(sender : Object, e : System.EventArgs) => dtpFechaInicio.ValueChanged
Se dispara cada que se cambia el valor del dtpFechaInicio.

- dtpFechaInicio_CloseUp(sender : Object, e : System.EventArgs) => dtpFechaInicio.CloseUp
Se dispara cada que se cierran las opciones del dtpFechaInicio

- dtpFechaFin_ValueChanged(sender : Object, e : System.EventArgs) => dtpFechaFin.ValueChanged
Se dispara cada que se cambia el valor del dtpFechaInicio.

- dtpFechaFin_CloseUp(sender : Object, e : System.EventArgs) => dtpFechaFin.CloseUp
Se dispara cada que se cierran las opciones del dtpFechaFin

- cboMoneda_DropDown(sender : Object, e : System.EventArgs) => cboMoneda.DropDown
Se dispara cada que se hace DropDown en el ComboBox del tipo de moneda.

- cboMoneda_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboMoneda.SelectedIndexChanged
Se dispara cada que se cambia el valor del ComboBox del tipo de moneda.

- cboFinanciado_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboFinanciado.SelectedIndexChanged
Se dispara cada que se cambia el valor del cboFinanciado.

- cboContrato_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboContrato.SelectedIndexChanged
Se dispara cada que se cambia el valor del cboContrato.

- cboDesembolso_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboDesembolso.SelectedIndexChanged
Se dispara cada que se cambia el valor del cboDesembolso.

- dtpFTCambio_ValueChanged(sender : Object, e : System.EventArgs) => dtpFTCambio.ValueChanged
Se dispara cada que se cambia el valor del dtpFTCambio.

- MenuItem1_Click(sender : System.Object, e : System.EventArgs) => MenuItem1.Click
Opción del menú contextual para seleccionar la imagen de Fondo.

- cboContratoVista_SelectedIndexChanged(sender : Object, e : System.EventArgs) => cboContratoVista.SelectedIndexChanged
Se dispara cada que se cambia el valor del cboContratoVista.

- btnConsultarVista_Click(sender : System.Object, e : System.EventArgs) => btnConsultarVista.Click
Se dispara cuando se oprime el botón de solo consulta de pagos.

- btnExcelVista_Click(sender : System.Object, e : System.EventArgs) => btnExcelVista.Click
Se dispara cuando se oprime el botón de enviar a Excel la información consultada.

- PagosNoProgramados_Closing(sender : Object, e : System.ComponentModel.CancelEventArgs) => MyBase.Closing
Se dispara cuando la ventana se está cerrando.

- PagosNoProgramados_Closing(sender : Object, e : System.ComponentModel.CancelEventArgs) => MyBase.Closing
Se dispara cuando la ventana se ya fue cerrada.

- rbValida_CheckedChanged(sender : System.Object, e : System.EventArgs) => rbValida.CheckedChanged
Se dispara cuando se cambia el valor del rbValida.

Clase Principal

Atributos

- components : System.ComponentModel.IContainer
El contenedor de los controles.

- barEstado : System.Windows.Forms.StatusBar
Barra de estado donde se le muestra diversa información al usuario.

- mnuPrincipal : System.Windows.Forms.MainMenu
Contenedor de los menús.

- mnuCatalogos : System.Windows.Forms.MenuItem
Menú que contiene como opciones cada uno de los catálogos.

- mnuDesembolsos : System.Windows.Forms.MenuItem
Menú que contiene la opción de abrir la ventana de Asignación de Créditos a los Organismos Corporativos.

- mnuPagos : System.Windows.Forms.MenuItem
Menú que contiene la opción de abrir la ventana de Pagos No Programados.

- mnuVer : System.Windows.Forms.MenuItem
Menú que las opciones de vista de ventanas.

- mnuAyuda : System.Windows.Forms.MenuItem
Menú que contiene la opción de acerca del sistema.

- popFondo : System.Windows.Forms.ContextMenu
Menú contextual de la pantalla principal.

- popBarra : System.Windows.Forms.ContextMenu
Menú contextual de la barra de herramienta.

- lstIconosBarra : System.Windows.Forms.ImageList
Conjunto de iconos de la barra de herramientas.

- stLeyenda : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde se muestra el texto "Gerencia de Financiamientos y Análisis de Mercado.

- stVersion : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde se muestra el texto "Versión 2.0".

- stFecha : System.Windows.Forms.StatusBarPanel
Panel de la barra de estado donde se muestra la fecha.

- mnuAcreedores : System.Windows.Forms.MenuItem
Opción del mnuCatalogos que abre la ventana del Catálogo de Acreedores.

- mnuVentanas : System.Windows.Forms.MenuItem
Opción del mnuVer que contiene opciones de vista de las ventanas.

- mnuCascada : System.Windows.Forms.MenuItem
Opción del mnuVentanas que permite visualizar las ventanas en cascada.

- mnuHorizontal : System.Windows.Forms.MenuItem
Opción del mnuVentanas que permite visualizar las ventanas de forma horizontal.

- mnuVertical : System.Windows.Forms.MenuItem
Opción del mnuVentanas que permite visualizar las ventanas de forma vertical.

- mnuBarras : System.Windows.Forms.MenuItem
Opción del mnuVer que contiene opciones de vista de los botones de la barra de herramientas.

- mnuVerDesembolsos : System.Windows.Forms.MenuItem
Opción del mnuBarras permite visualizar los botones de desembolsos en la barra de herramientas.

- mnuVerPagos : System.Windows.Forms.MenuItem
Opción del mnuBarras que permite visualizar los botones de desembolsos en la barra de herramientas.

- mnuVerVista : System.Windows.Forms.MenuItem
Opción del mnuBarras que permite visualizar los botones de vista en la barra de herramientas.

- mnuVerOpciones : System.Windows.Forms.MenuItem
Opción del mnuBarras que permite visualizar los botones de opciones en la barra de herramientas.

- mnuAcercaDe : System.Windows.Forms.MenuItem
Opción del mnuAyuda que envía un mensaje al usuario con información de la aplicación.

- popCascada : System.Windows.Forms.MenuItem
Opción del popFondo que permite visualizar las ventanas abiertas en cascada.

- popAcercaDe : System.Windows.Forms.MenuItem
Opción del popFondo que envía un mensaje al usuario con información de la aplicación.

- popSalir : System.Windows.Forms.MenuItem
Opción del popFondo permite salir de la aplicación.

- popCatalogos : System.Windows.Forms.MenuItem
Opción de popBarra que permite visualizar los botones de la barra de herramientas que se refiere a los catálogos.

- popDesembolsos : System.Windows.Forms.MenuItem
Opción de popBarra que permite visualizar los botones de la barra de herramientas que se refiere a desembolsos.

- popPagos : System.Windows.Forms.MenuItem
Opción de popBarra que permite visualizar los botones de la barra de herramientas que se refiere a pagos.

- popVista : System.Windows.Forms.MenuItem
Opción de popBarra que permite visualizar los botones de la barra de herramientas que se refiere a vista.

- popOpciones : System.Windows.Forms.MenuItem
Opción de popBarra que permite visualizar los botones de la barra de herramientas que se refiere a opciones.

- btnCascada : System.Windows.Forms.ToolBarButton
Botón de la barra de herramientas cuya función es visualizar en cascada las ventanas abiertas.

- btnAcercaDe : System.Windows.Forms.ToolBarButton
Botón de la barra de herramientas cuya función es mostrarle al usuario información de la aplicación.

- btnSalir : System.Windows.Forms.ToolBarButton
Botón de la barra de herramientas cuya función es salir de las aplicaciones.

- mnuVerCatalogos : System.Windows.Forms.MenuItem
Opción del mnuBarras permite visualizar los botones de catálogos en la barra de herramientas.

- btnSepCatalogos : System.Windows.Forms.ToolBarButton
Separación para agrupar botones en la barra de herramientas.

- btnSepDesembolsos : System.Windows.Forms.ToolBarButton
Separación para agrupar botones en la barra de herramientas.

- btnSepPagos : System.Windows.Forms.ToolBarButton
Separación para agrupar botones en la barra de herramientas.

- btnSepCascada : System.Windows.Forms.ToolBarButton
Separación para agrupar botones en la barra de herramientas.

- barHerramientas : System.Windows.Forms.ToolBar
Barra de herramientas.

- sepPopFondo : System.Windows.Forms.MenuItem
Separador de opciones en PopFondo.

- sepPopBarra : System.Windows.Forms.MenuItem
Separador de opciones popBarra.

- sepVerBarras : System.Windows.Forms.MenuItem
Separador de opciones mnuBarras.

- mnuContratos : System.Windows.Forms.MenuItem
Opción del mnuCatálogos que permite abrir la pantalla del Catálogo de Contratos.

- mnuConceptosPago : System.Windows.Forms.MenuItem
Opción del mnuCatálogos que permite abrir la pantalla del Catálogo de Conceptos de Pago.

- mnuAltasAcreedores : System.Windows.Forms.MenuItem
Opción del mnuCatálogos que permite abrir la pantalla del Catálogo de Acreedores.

- mnuTransaccionesAcreedores : System.Windows.Forms.MenuItem
Opción del mnuCatálogos que permite abrir la pantalla de transacciones de acreedores.

- mnuBancosHouston : System.Windows.Forms.MenuItem
Opción del mnuCatálogos que permite abrir la pantalla del Catálogo Bancos-Houston.

- mnuNoProgramados : System.Windows.Forms.MenuItem
Opción del mnuPagos que permite abrir la pantalla de Pagos No Programados.

- btnContratos : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana del Catálogo de Contratos.

- btnCambioCierre : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana del Catálogo Tipos de Cambio de Cierre.

- btnConceptosPago : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana del Catálogo de Conceptos de pago.

- btnAcreedores : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana del Catálogo de Acreedores.

- btnBancosHouston : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana del Catálogo Bancos-Houston.

- btnNoProgramados : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana de Pagos No Programados.

- mnuCambioCierre : System.Windows.Forms.MenuItem
Opción del mnuCatalogos que permite abrir la pantalla del catálogo de Tipos de Cambio de Cierre.

- mnuAsignacion : System.Windows.Forms.MenuItem
Opción del mnuDesembolsos que permite abrir la pantalla de Asignación de Créditos a los Organismos Corporativos.

- btnAsignacion : System.Windows.Forms.ToolBarButton
Botón de la barra de Herramientas que permite abrir la ventana de Asignación de Créditos a los Organismos Corporativos.

- autentica : Autenticación
Para abrir la ventana de autenticación para entrar al sistema.

- mensaje : mensaje
Para enviar mensajes al usuario.

- cambios : CambioCierre
Para la ventana del Catálogo de Tipos de Cambio de Cierre.

- asignacion : AsignacionCreditos
Para la ventana de Asignación de créditos a los Organismos Corporativos.

- conceptos : ConceptosPago
Para la ventana del Catálogo de Conceptos de Pago.

- contrato : Contratos
Para la ventana del Catálogo de Contratos.

- houston : BancosHouston
Para la ventana del Catálogo Bancos-Houston.

- pagos : PagosNoProgramados
Para la ventana de Pagos No Programados.

- acreedores : FormAcreMovs
Para la ventana de Catálogo de Acreedores.

- transacciones : FormTransac
Para la ventana de transacciones entre acreedores.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana.

Dispose(disposing : Boolean)
Destruye al objeto.

Eventos

- Principal_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- mnuContratos_Click(sender : System.Object, e : System.EventArgs) => mnuContratos.Click
Se dispara cuando se selecciona mnuContratos.

- mnuCambiosCierre_Click(sender : System.Object, e : System.EventArgs) => mnuCambioCierre.Click
Se dispara cuando se selecciona mnuCambiosCierre.

- mnuConceptosPago_Click(sender : System.Object, e : System.EventArgs) => mnuConceptosPago.Click
Se dispara cuando se selecciona mnuConceptosPago.

- mnuAltasAcreedores_Click(sender : System.Object, e : System.EventArgs) => mnuAltasAcreedores.Click
Se dispara cuando se selecciona mnuAltasAcreedores.

- mnuTransaccionesAcreedores_Click(sender : System.Object, e : System.EventArgs) => mnuTransaccionesAcreedores.Click
Se dispara cuando se selecciona mnuTransaccionesAcreedores.

- mnuBancosHouston_Click(sender : System.Object, e : System.EventArgs) => mnuBancosHouston.Click
Se dispara cuando se selecciona mnuBancosHouston.

- mnuAsigancion_Click(sender : System.Object, e : System.EventArgs) => mnuAsignacion.Click
Se dispara cuando se selecciona mnuAsignacion.

- mnuNoProgramados_Click(sender : System.Object, e : System.EventArgs) => mnuNoProgramados.Click
Se dispara cuando se selecciona mnuNoProgramados.

- mnuCascada_Click(sender : System.Object, e : System.EventArgs) => mnuCascada.Click, mnuHorizontal.Click, mnuVertical.Click
Se dispara cuando se selecciona mnuCascada, mnuHorizontalo o mnuVertical.

- mnuVerCatálogos_Click(sender : System.Object, e : System.EventArgs) => mnuVerCatalogos.Click, popCatalogos.Click
Se dispara cuando se selecciona mnuVerCatalogos o popCatalogos
- mnuVerDesembolsos_Click(sender : System.Object, e : System.EventArgs) => mnuVerDesembolsos.Click, popDesembolsos.Click
Se dispara cuando se selecciona mnuVerDesembolsos o popDesembolsos.

- mnuVerPagos_Click(sender : System.Object, e : System.EventArgs) => mnuVerPagos.Click, popPagos.Click
Se dispara cuando se selecciona mnuVerPagos o popPagos.

- mnuVerVista_Click(sender : System.Object, e : System.EventArgs) => mnuVerVista.Click, popVista.Click
Se dispara cuando se selecciona mnuVerVista o popVista.

- mnuVerOpciones_Click(sender : System.Object, e : System.EventArgs) => mnuVerOpciones.Click, popOpciones.Click
Se dispara cuando se selecciona mnuVerOpciones o popOpciones.

- mnuAcercaDe_Click(sender : System.Object, e : System.EventArgs) => mnuAcercaDe.Click
Se dispara cuando se selecciona mnuAcercaDe.

- barHerramientas_ButtonClick(sender : System.Object, e : System.Windows.Forms.ToolBarButtonClickEventArgs) => barHerramientas.ButtonClick
Se dispara cuando se oprime algún botón de la barra de herramientas.

- Principal_Closing(sender : Object, e : System.ComponentModel.CancelEventArgs) => MyBase.Closing
Se dispara cuando se está cerrando la ventana.

- popAcercaDe_Click(sender : System.Object, e : System.EventArgs) => popAcercaDe.Click
Se dispara cuando se selecciona popAcercaDe.

- popCascada_Click(sender : System.Object, e : System.EventArgs) => popCascada.Click
Se dispara cuando se selecciona popCascada.

- popSalir_Click(sender : System.Object, e : System.EventArgs) => popSalir.Click
Se dispara cuando se selecciona popSalir.

Clase Monitoreo

Atributos

- components : System.ComponentModel.IContainer
Contenedor de controles.

- gbxConsultas : System.Windows.Forms.GroupBox
Marco que encierra los elementos para la consulta de información.

- lblDeuda : System.Windows.Forms.Label
Etiqueta con leyenda "Deuda".

- cboDeuda : System.Windows.Forms.ComboBox
ComboBox con información de los tipos de deuda.

- chkR047 : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla R047.

- chkR018 : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla R018.

- chkIntFoto : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla Int_Foto.

- chkR05051 : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla V_R5051.

- chkR086 : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla R086.

- chkTabla : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla Tablas_de_amortizacion.

- chkR019 : System.Windows.Forms.CheckBox
CheckBox para indicar si la búsqueda también se hará en la tabla R019.

- cboFechas : System.Windows.Forms.ComboBox
ComboBox con las fechas posibles con base en la elección de contrato y desembolso.

- cboConceptoPago : System.Windows.Forms.ComboBox
ComboBox con los tipos de conceptos de pago.

- lblConceptoPago : System.Windows.Forms.Label
Etiqueta con leyenda "Concepto de Pago".

- lblDesembolso : System.Windows.Forms.Label
Etiqueta con leyenda "Desembolso".

- lblContrato : System.Windows.Forms.Label
Etiqueta con leyenda "Contrato".

- cboMonedas : System.Windows.Forms.ComboBox
ComboBox con la lista de tipos de monedas.

- lblMoneda : System.Windows.Forms.Label
Etiqueta con leyenda "Moneda".

- cboOrganismos : System.Windows.Forms.ComboBox
ComboBox con los distintos organismos que existen.

- lblOrganismo : System.Windows.Forms.Label
Etiqueta con leyenda "Organismo".

- lblFecha : System.Windows.Forms.Label
Etiqueta con leyenda "Fecha".

- cboContratos : System.Windows.Forms.ComboBox
ComboBox con la lista de contratos.

- cboDesembolsos : System.Windows.Forms.ComboBox
ComboBox con la lista de desembolsos.

- chkTodo : System.Windows.Forms.CheckBox
CheckBox para seleccionar todas las tablas posibles.

- btnConsultar : System.Windows.Forms.Button
Botón para hacer las consultas.

- dataGridR018 : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla R018.

- dataGridR047 : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla R047.

- dataGridV_R05051 : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla V_5051.

- dataGridIntFoto : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla Int_Foto.

- dataGridR086 : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla R086.

- dataGridTablas : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla Tablas_de_amortizacion.

- dataGridR019 : System.Windows.Forms.DataGrid
DataGrid con la información requerida de la tabla R019.

- dataSetMonitoreo : DataSet
DataSet donde se almacenarán las consultas realizadas.

- queryR047 : String
Sentencia SQL para la búsqueda en la tabla R047.

- queryR018 : String
Sentencia SQL para la búsqueda en la tabla R018.

- queryR019 : String
Sentencia SQL para la búsqueda en la tabla R019.

- queryTabla : String
Sentencia SQL para la búsqueda en la tabla tablas_de_amortizacion.

- queryR086 : String
Sentencia SQL para la búsqueda en la tabla R086.

- queryR05051 : String
Sentencia SQL para la búsqueda en la tabla V_R05051.

- queryIntFoto : String
Sentencia SQL para la búsqueda en la tabla Int_Foto.

Métodos

+ New()
Constructor de la clase.

- InitializeComponent()
Coloca y acomoda todos los componentes en la ventana.

Dispose(disposing : Boolean)
Destruye al objeto.

- iniciaCombos()
Llena los ComboBox con información inicial.

- llenaDesembolsos()
Llena de información el ComboBox de los desembolsos.

- llenaFechas()
Llena de información el ComboBox de las fechas posibles.

Eventos

- Monitoreo_Load(sender : System.Object, e : System.EventArgs) => MyBase.Load
Se dispara cuando la ventana se está iniciando.

- btnConsultar_Click(sender : System.Object, e : System.EventArgs) => btnConsultar.Click
Se dispara cuando btnConsultar es oprimido.

- chkR047_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkR047.CheckedChanged
Se dispara cuando chkR047 es marcado.

- chkR018_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkR018.CheckedChanged
Se dispara cuando chkR018 es marcado.

- chkR019_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkR019.CheckedChanged
Se dispara cuando chkR019 es marcado.

- chkTabla_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkTabla.CheckedChanged
Se dispara cuando chkTabla es marcado.

- chkR086_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkR086.CheckedChanged
Se dispara cuando chkR086 es marcado.

- chkR05051_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkR05051.CheckedChanged
Se dispara cuando chkR05051 es marcado.

- chkIntFoto_CheckedChanged(sender : System.Object, e : System.EventArgs) => chkIntFoto.CheckedChanged
Se dispara cuando chkIntFoto es marcado.

- cboContratos_SelectedIndexChanged(sender : System.Object, e : System.EventArgs) => cboContratos.SelectedIndexChanged, cboContratos.TextChanged
Se dispara cuando se cambia el valor o el texto de cboContratos.

- cboDesembolsos_SelectedIndexChanged(sender : System.Object, e : System.EventArgs) => cboDesembolsos.SelectedIndexChanged, cboDesembolsos.TextChanged
Se dispara cuando se cambia el valor o el texto de cboDesembolsos.

GLOSARIO

Combo Box: Objeto perteneciente a las bibliotecas de .NET el cual es utilizado en las interfaces gráficas para seleccionar un elemento de una lista.

Data Base Trigger: Es una unidad de programa en PL/SQL almacenado en la base de datos asociado a eventos de una tabla específica de la base de datos.

Data Grid: Objeto perteneciente a las bibliotecas de .NET el cual es utilizado en las interfaces gráficas para mostrar un conjunto de datos en forma de tabla. Aquí se puede modificar la información mostrada y agregar nueva.

Date Time Picker: Objeto perteneciente a las bibliotecas de .NET el cual es utilizado en las interfaces gráficas para seleccionar una fecha.

Requerimiento funcional: Es aquel que debe realizar un sistema de software en base a las necesidades del usuario, generalmente se deriva de un caso de uso.

Requerimiento no funcional: Es aquel que se refiere a la forma en que un sistema de software da solución a los requerimientos funcionales.

Stored Procedure: Es un bloque de instrucciones en PL/SQL para realizar una tarea específica sobre la base de datos.

BIBLIOGRAFÍA

Larman Craig. UML y Patrones, Introducción al análisis y diseño orientado a objetos. Mexico: Prentice Hall. 1999.

http://msdn.microsoft.com/

http://es.wikipedia.org/wiki/.NET_de_Microsoft

http://www.dcc.uchile.cl/~psalinas/uml/casosuso.html

http://www.dcc.uchile.cl/~psalinas/uml/modelo.html

Birnios Mariano N. Visual Basic .NET. Argentina: MP Ediciones. 2002.

http://www.desarrolloweb.com/articulos/1328.php?manual=48