UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

APLICACIÓN DEL LENGUAJE UNIFICADO DE MODELADO UML PARA EL DISEÑO DE SOFTWARE. CASO DE ESTUDIO: SISTEMA DE INFORMACIÓN FINANCIERA EN UNA COMPAÑÍA ASEGURADORA.

T R A B A J O D E E X P E R I E N C I A P R O F E S I O N A L

QUE PARA OBTENER EL TÍTULO DE:

ACTUARIO

P R E S E N T A :

CARLOS ALBERTO GUTIÉRREZ CRUZ

TUTORA: DRA. ELISA VISO GUROVICH

2006

Hola de Datos del Jurado

1. Datos del alumno Gutiérrez Cruz Carlos Alberto 56 84 26 31 Universidad Nacional Autónoma de México Facultad de Ciencias Actuaría 081380437
2. Datos del tutor Dra. Elisa Viso Gurovich
3. Datos del sinodal 1 M. en C. José de Jesús Galaviz Casas
4. Datos del sinodal 2 M. en I.O. María de Luz Gasca Soto
5. Datos del sinodal 3 Act. Fernando Alonso Pérez-Tejada López
6. Datos del sinodal 4 Lic. en C. C. Francisco Lorenzo Solsona Cruz
7. Datos del trabajo escrito Aplicación del Lenguaje Unificado de Modelado UML para el Diseño de Software. Caso de Estudio: Sistema de Información Financiera en una Compañía Aseguradora. 46 p 2006

CONTENIDO

LISTA DE FIGURAS

INTRODUCCIÓN

El presente trabajo es el producto de la experiencia profesional obtenida durante mi desempeño en el desarrollo de sistemas de información, el cual está basado en la elaboración de un Sistema de Información Financiera para una compañía aseguradora.

El trabajo consta de tres capítulos, el primero describe los antecedentes del proyecto, la problemática a resolver y la visión general del sistema a desarrollar.

El capítulo II, muestra cómo se desarrollaron las fases de Análisis y Diseño para el proyecto, el uso del Lenguaje Unificado de Modelado UML para ir modelando y visualizando los requerimientos del sistema, y cómo la documentación generada se utiliza durante toda la vida del proyecto.

El capítulo III describe las etapas de Desarrollo e Implementación, en las cuales se comenta cómo se fue implementando el sistema para estas etapas, es decir, cómo se crearon algunos de los algoritmos utilizados para la codificación y construcción del sistema, haciendo énfasis en que durante estas etapas del proyecto se requiere un nuevo análisis para poder detectar irregularidades que el diseño no puede visualizar y que en ocasiones se basan en el tipo de lenguaje o tecnología de desarrollo ha utilizar.

Posteriormente, se proporcionan las conclusiones del presente trabajo como producto de la experiencia obtenida, sus alcances y efectos después de la realización del proyecto, y finalmente una serie de recomendaciones que puedan ser útiles para futuros casos.

En el apéndice se hace referencia a los documentos y diagramas de la fase de análisis y diseño, el código generado en la fase de desarrollo e implementación y el diagrama de la base de datos.

Finalmente, se cita la bibliografía que sirvió de base y referencia en la elaboración de este trabajo.

OBJETIVO GENERAL

Presentar un trabajo como producto de conocimientos y experiencias profesionales, el cual está basado en un proyecto de desarrollo de sistemas a fin de mostrar los procesos y actividades realizadas desde su concepción, análisis y diseño con UML conforme a su versión 1.5, y finalmente su desarrollo e implementación.

JUSTIFICACIÓN

El caso de estudio está basado en un proyecto realizado durante el periodo de Marzo de 2004 a Febrero de 2005 para la compañía aseguradora Generali de México, S.A., en el cual participé como líder de proyecto, desde la concepción del sistema hasta su implementación y liberación.

Este trabajo responde a la necesidad de contar con un sistema central que permita automatizar y apoyar la operación administrativa y contable de los egresos que la compañía genera en cuanto al pago a proveedores, siniestros y comisiones de agentes, a fin de agilizar el trámite de pago de los mismo, un mejor control sobre dichas operaciones, maximizar los recursos financieros, contar con una distribución de gastos más eficiente por Centro de Costos y que deriven en la generación de Estados Financieros mas confiables.

En el caso de estudio mencionado se ejemplifica de manera práctica el uso del UML en el desarrollo de sistemas.

CAPÍTULO I

ENTORNO DEL PROBLEMA

Descripción General y Propósito del Proyecto.

Generalí de México desea contar con un sistema integral de gestión central de todos los egresos de la compañía que son generados por cada Centro de Costos. Dicho sistema debe permitir el control de la operación y autorización de dicha gestión.

Mediante niveles de seguridad el sistema deberá permitir el correcto acceso a funciones y módulos destinados para su operación, y deberán tener controles que auditen su utilización.

Para cumplir estos objetivos, las autorizaciones de pago registradas en el sistema se realizarán de manera central y se tendrán procesos automatizados que permitan registrar confiablemente todos los movimientos contables y realizar las debidas operaciones de conciliación bancaria contra los estados de cuenta de la compañía. Esto permitirá de manera más oportuna detectar cualquier desviación que se pueda presentar en el flujo de la operación.

Alcances o Requerimientos del sistema.

1. **Seguridad.**

Se debe contar con un módulo de Seguridad, para la identificación de todos los usuarios para la operación del sistema.

Se deberá contar con una bitácora de las operaciones realizadas por los usuarios dentro del sistema.

2. **Administración de Catálogos.**

El módulo de Administración de Catálogos deberá manejar todos los catálogos del sistema: control de usuarios, asegurados o beneficiarios, proveedores, bancos e información contable.

3. Registro de Órdenes de Pago.

Este módulo deberá presentar las siguientes capacidades:

Registro de Órdenes de Pago y Anticipos de Gastos por Usuario Solicitante y Centro de Costos

Registro automático de Pagos de Siniestros como Órdenes de Pago los cuales son generados en los Sistemas de Información de los Ramos de Daños, Vida y Gastos Médicos.

Poder integrar funcionalidad para el manejo de multimoneda, para permitir que una orden de pago pueda ser tramitada en otro tipo de moneda, como por ejemplo dólares, aún cuando el registro de sus importes se realice en pesos.

Ser capaces de realizar la afectación contable de manera automática en el Sistema de Contabilidad sobre órdenes de pago autorizadas.

4. Gestión Central de Órdenes de Pago.

Este módulo debe poder:

Centralizar los egresos de la Compañía en el nuevo sistema con la información de egresos que se genere con la operación día a día.

Realizar la clasificación dinámica de la información para facilitar su control y autorización por parte del Departamento de Tesorería.

Proporcionar una interfaz con los sistemas bancarios que utiliza la compañía para la generación de pagos por transferencia bancaria.

Realizar la impresión y reimpresión de cheques con control de asignación automática de la numeración por oficinas.

Automatizar los procesos de conciliación bancaria sobre la operación de pagos realizados vía cheque y transferencias electrónicas, contra los estados de cuenta bancarios de la Compañía.

Durante esta primera fase del proyecto se establecieron de manera general todos los requerimientos deseables y factibles del sistema. Esto es importante para cualquier inicio de un proyecto de sistemas, ya que permitirá visualizar con mayor claridad sus alcances y proporcionará el entorno sobre el que se elaborarán las fases siguientes.

En el siguiente capítulo presentamos el análisis de estos requerimientos y el diseño del sistema que los satisface.

CAPÍTULO II

ANÁLISIS Y DISEÑO

Modelando el Comportamiento del Sistema.

Durante esta fase del proyecto, se pretende modelar el comportamiento del sistema, es decir, la funcionalidad que debe proporcionar. Para ello se crean los Casos de Uso y los siguientes diagramas: de Casos de Uso, de Actividades, de Paquetes y de Clases.

Creación de los Casos de Uso.

Los modelos o diagramas de Casos de Uso forman parte de la información que es utilizada para representar el comportamiento del sistema, es decir, la funcionalidad que debe proporcionar el sistema. El papel más importante de los diagramas de Casos de Uso es el de la comunicación, proporcionando un medio para que los clientes o usuarios finales y los desarrolladores discutan la funcionalidad y el comportamiento del sistema.

Identificación de Actores.

Para una correcta identificación de los actores del sistema, cabe señalar que los actores no son parte del sistema, ellos representan a cualquier persona o algo que debe interactuar con el sistema y las siguientes preguntas pueden ser útiles para ayudar a identificarlos.

- ¿Quiénes están interesados en los requerimientos?
- Dentro de la Organización, ¿dónde es utilizado el sistema?
- ¿Quiénes se beneficiarán de la utilización del sistema?
- ¿Quiénes proporcionarán información al sistema y quiénes harán uso de ella?
- ¿Quién dará mantenimiento al sistema?
- ¿El sistema hará uso de algún recurso externo?
- ¿Existe algún usuario que realiza diferentes roles?

- ¿Existen varios usuarios que realizan el mismo rol?
- ¿El sistema interactúa con algún otro sistema dentro de la Organización?

Para el sistema en cuestión, las preguntas previas fueron respondidas de la siguiente manera:

- Los Usuarios Administrativos de cada Centro de Costos solicitan el pago de una orden a un proveedor de servicios o un anticipo de gastos para un empleado.

- Los Usuarios de Contabilidad requieren conocer todas las órdenes de pago pendientes para su autorización o rechazo.

- Los Usuarios del área de Finanzas y Tesorería requieren clasificar la información de los egresos a fin de darles seguimiento para su pago conforme a los recursos financieros disponibles de la Compañía.

- El Usuario Administrador mantiene la información de los catálogos del sistema actualizados.

- El Sistema Contable requiere recibir la información del nuevo sistema sobre los egresos realizados y autorizados por los usuarios.

- Los Sistemas de Información donde se realiza la operación de los Seguros de Daños, Vida y Gastos Médicos requieren enviar y recibir del nuevo sistema la información de pagos de siniestros y los estatus de autorización de pago, respectivamente.

- Los Sistemas Bancarios requieren recibir la información del nuevo sistema de todos aquellos pagos autorizados por medio de transferencia electrónica o bancaria.

Identificación de Casos de Uso.

Para la identificación de casos de uso dentro del caso de estudio en cuestión se partió de la revisión de los requisitos del sistema.

Los casos de uso, como ya mencionamos, modelan la interacción de los actores con el sistema y representan la funcionalidad provista por dicho sistema. Un caso de uso es una secuencia de transacciones realizadas por el sistema y que proporciona una serie de resultados o valores para un actor en particular.

Las siguientes preguntas pueden ser utilizadas para identificar los casos de uso del sistema.

- ¿Cuáles son las tareas de cada actor?
- ¿Cualquier actor podrá crear, almacenar, modificar, eliminar o consultar información del sistema?
- ¿Qué casos de uso crearán, almacenarán, modificarán, eliminarán o consultarán información del sistema?
- ¿Tendrá un actor la necesidad de informar al sistema acerca de cambios externos?
- ¿Tiene que ser un actor informado por cambios que ocurran en el sistema?
- ¿Qué casos de uso darán soporte y mantenimiento al sistema?
- ¿Pueden todos los requerimientos funcionales ser realizados por los casos de uso?

Basado en los requerimientos o alcances del sistema, los siguientes casos de uso fueron identificados. Asimismo, se enumeraron a fin de contar con una referencia de cada uno de ellos durante todo el ciclo de vida del proyecto.

1. **Funciones de Seguridad.**

Ref.	Función
R1.1	Validar clave de acceso y definir perfil de funcionalidad dentro del sistema.

R1.2	Registro de operaciones realizadas por el usuario dentro del sistema.

2. Funciones de Catálogos.

Ref.	Función
R2.1	Mantenimiento del Catálogo de Usuarios
R2.2	Mantenimiento de Catálogos de Operación del Sistema

3. Funciones de órdenes de pago y anticipos de gastos.

Ref.	Función
R3.1	Registro de órdenes de pago y anticipos de gastos dentro del Sistema
R3.2	Carga de información de los Sistemas de Información de Daños, Vida y Gastos Médicos.
R3.3	Envío de información contable al Sistema Central Contable.

4. Funciones para autorización de pagos.

Ref.	Función
R4.1	Autorización de Órdenes de Pago.
R4.2	Generación de archivo para liberación automática de cheques.
R4.3	Generación de archivo para carga de operaciones autorizadas de pago vía transferencia en los sistemas bancarios.

R4.4	Impresión de cheques para pago de operaciones autorizadas
R4.5	Conciliación bancaria por cuenta.

Descripción de Casos de Uso.

La importancia de describir brevemente un caso de uso, es mostrar el propósito de éste en unas cuantas oraciones, comunicando con esto el flujo y funcionalidad esperada que tendrá el sistema con un actor que utiliza o interviene en él.

Cabe señalar que UML no define un formato para describir un caso de uso, tan sólo define la manera de representar la relación entre actores y casos en un diagrama. Un formato que puede ser de utilidad para definir los casos de uso es el siguiente.

- **Caso de Uso:** Nombre del Caso de Uso.
- **Actores:** Lista de Actores.
- **Descripción:** Intención del Caso de Uso.
- **Precondiciones:** En términos de si existen reglas o condiciones que deban cumplirse antes de iniciar el flujo básico o principal del caso de uso.
- **Flujo básico de eventos:** Descripción de la interacción de los actores y el sistema mediante las acciones numeradas de cada uno. Describe la secuencia común de eventos, cuando todo va bien y el proceso se completa satisfactoriamente. En caso de haber alternativas se añaden secciones adicionales a la sección principal.
- **Flujo alternativo:** Si existen cursos alternos posibles por los que el flujo básico pueda desviarse antes de completarse.
- **Post-condiciones:** En términos de si existen reglas o condiciones que deban cumplirse antes de que el caso de uso finalice.
- **Puntos de extensión:** Representa aquellos puntos en el flujo del caso de uso que se pueden extender.

A continuación se muestra la descripción del caso de uso para la función de Orden de Pago y Anticipo de Gastos con referencia R3.1, la cual trata sobre el Registro de Órdenes de Pago.

Caso de Uso: R3.1 Registro de Órdenes de Pago.

Actores: Usuario Administrativo.

Descripción: Caso de Uso iniciado por el Usuario Administrativo. Éste proporciona la capacidad de registrar una orden de pago o anticipo de gastos.

Precondiciones: El usuario deberá estar validado por el sistema y que su perfil de autorización permita registrar una orden de pago.

Flujo Básico:

Acción del Actor	**Acción del Sistema**
1. Usuario ingresa a la pantalla de registro para órdenes de pago	2. Sistema presenta pantalla de registro de órdenes de pago. Se muestra la fecha actual del sistema. Se cargan de manera predeterminada los datos del centro del costo del usuario, con opción a modificar el centro si es que la orden va a ser registrada con aplicación a otro centro de costo. Los centros opcionales para el usuario están definidos con base en su rol.
3. Selecciona el centro de costos de la orden. 4. Selecciona el tipo de beneficiario.	5. Dependiendo del tipo de Beneficiario, el Sistema despliega lo beneficiarios registrados así como el banco y la cuenta de depósito en caso de contar con ella.
6. Selecciona Forma de Pago	7. Si el Beneficiario seleccionado contiene la información previamente capturada de la cuenta bancaria o CLABE. Flujo alternativo A1. Puede seleccionar la forma de pago cheque.
8. Seleccionará el concepto (plantilla), proporcionará el importe, dará clic en el botón de agregar.	9. El sistema calcula en base a las plantillas contables los rubros que se afectaran contablemente.
10. Podrá consultar los movimientos contables a afectar con dar clic en el botón de detalle contable.	
11. Para registrar la orden el usuario da	12. Si el tipo de pago seleccionado es por

clic en botón de aceptar.	transferencia bancaria, y el beneficiario no tiene dada de alta su cuenta bancaria, Flujo Alternativo A2. Asigna Número de Orden y registra.
13. Usuario Sale de la Pantalla y Termina Caso de Uso.	

Flujo Alternativo: Existen dos flujos alternativos, A1. El sistema sugerirá que la forma sea por transferencia bancaria. A2. El sistema enviará el mensaje de que no se puede registrar la orden si el tipo de pago es por transferencia y el beneficiario no cuenta con sus datos bancarios correspondientes. El sistema sugiere opción de pago con cheque o terminar el caso de uso.

Post-Condiciones: Orden registrada.

Puntos de Extensión: Ninguno.

Diagrama del Caso de Uso.

Como se comentó, el diagrama de Caso de Uso es una vista de uno o más actores y su interacción con el sistema. En el diagrama de la Figura 1, se muestra el Caso de Uso para el Registro de Órdenes de Pago (R3.1), mismo que fue descrito anteriormente y en el cual se observan las relaciones existentes entre los diferentes objetos que se identificaron del caso de uso. Hay que recordar que dichos objetos los componen lo Actores del Caso de uso, procesos o funcionalidades necesarias para realizar alguna tarea, así como objetos de datos que son parte del sistema.

Para el caso del Registro de Órdenes de Pago, el usuario administrador utiliza la funcionalidad de Seguridad, para validar que es un usuario autorizado, para acceder al registro de órdenes de pago, así como utilizar sólo aquella información relacionada con su centro de costos (Oficina+Departamento). Una vez que ingresa al registro de órdenes de pago, solamente puede generar y colocar órdenes para los tipos de gastos autorizados para su centro de costos y a favor de aquellas personas que están autorizadas por el área de finanzas y

contabilidad. Estas reglas y funcionalidades son mostradas y compartidas en la mayoría de los casos de uso, por lo que éstas están denotadas con <<*include*>>. Las funcionalidades que requieren de procesos o funciones adicionales para incrementar su funcionalidad son representadas por la notación <<*extend*>>.

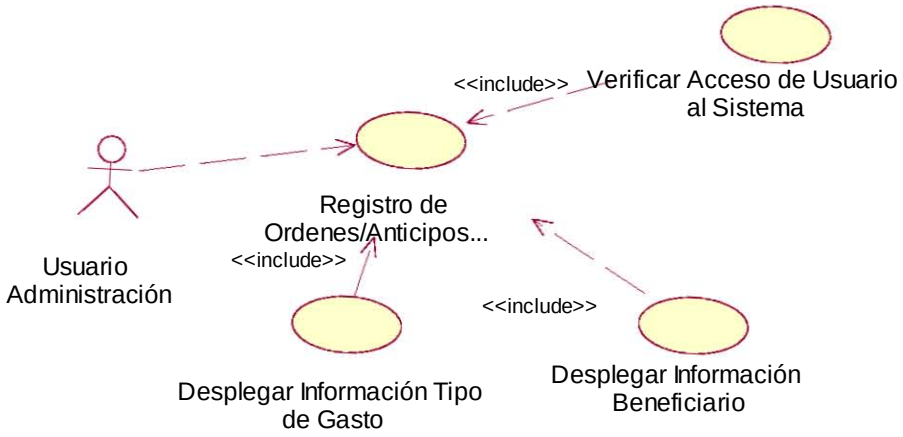


Figura 1. Diagrama de Caso de Uso

En el apéndice se hace referencia a toda la documentación y diagramas elaborados en el proyecto.

Creación de Diagramas de Actividades.

En esta etapa del proyecto pueden crearse los diagramas de actividades; ellos pueden proporcionar una mejor visión de las funcionalidades que se tienen definidas hasta este momento, ya sea para representar el flujo que existe entre los distintos casos de uso o para representar el flujo específico de un solo caso de uso.

La importancia que toma la creación de diagramas de actividades en esta etapa ayudará a la identificación de actividades iniciales y finales, puntos de decisión y puntos de transición que pudieran existir en el flujo de las actividades principales del sistema, por lo que posteriormente, conforme se vaya avanzando en el proyecto, veremos cómo los diagramas de actividades serán útiles para mostrar a detalle operaciones específicas dentro de una funcionalidad definida para el sistema.

El diagrama de actividades para el caso de Uso R3.1 que se refiere al proceso de Registro de Órdenes de Pago se muestra a continuación en la Figura 2, donde se identifica el flujo de las actividades durante el proceso.

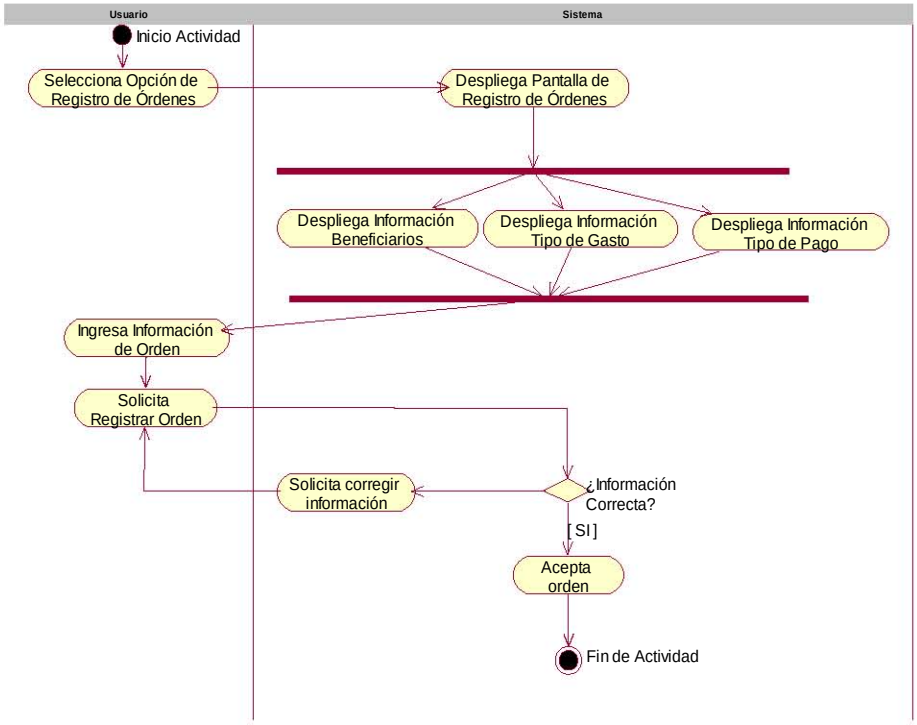


Figura 2. Diagrama de Actividades

En esta etapa del proyecto se ha visto el análisis y diseño del comportamiento del sistema, el cual se documenta en los casos de uso, la identificación de los actores y la funcionalidad que tendrán dentro del sistema. La importancia de estos, como se comentó, es la de comunicar la funcionalidad y el comportamiento que tendrá el sistema para el cliente o usuario final y se complementan con la elaboración de diagramas de actividades para aquellas funcionalidades específicas que requieran de una mayor comunicación con los usuarios finales.

Identificando Clases.

Una de las tareas más difíciles dentro de las fases de diseño de un proyecto es la identificación de clases.

Una clase es la descripción de un conjunto de objetos con propiedades, comportamientos, y relaciones comunes entre objetos, a fin de que funcione como una plantilla para la creación de dichos objetos.

Asimismo, el concepto de objeto se refiere a una representación de una clase o entidad, la cual puede representar algo concreto como por ejemplo una computadora, o puede representar algo conceptual como por ejemplo un proceso de transferencia bancaria. Un objeto tiene tres características principales, que son: estado, su comportamiento y su identidad.

El estado de un objeto trata sobre las posibles condiciones en la cual el objeto puede existir, su cambio con el tiempo, y está definido por un conjunto de propiedades usualmente llamadas atributos.

Por ejemplo, para el caso de estudio del Sistema de Información Financiera, después de registrar una orden de pago, ésta puede ser autorizada o cancelada conforme al flujo normal que debe seguir dentro de cada área responsable.

El comportamiento determina cómo un objeto responde a las peticiones hechas por otros objetos y es determinado por la definición de una serie de operaciones que debe realizar el objeto; como por ejemplo, al estar registrando la orden de pago, ésta presenta la información de beneficiarios dependiendo si es un pago a proveedores o el pago de un siniestro del ramo de automóviles.

Por último, la identidad significa que cada objeto es único, aún si su estado es idéntico al de otro objeto; por ejemplo, la orden de pago 5 del centro de costos 101 y la orden de pago 5 del centro de costos 102, son objetos distintos para el Sistema de Información Financiera.

Una buena forma para iniciar con la identificación de clases es partiendo de los Casos de Uso desarrollados en la fase de análisis y diseño, después listar nombres de actores o procesos que pudieran darnos una idea de cuáles son parte integral del sistema. Esta lista inicial puede ser denominada lista de candidatos a clases, la cual deberá pasar por un proceso de filtrado

para la identificación de aquellos nombres o procesos que indiquen duplicidad o redundancia, así como aquellos que están fuera de los límites y alcances del sistema.

Observando el escenario de Registro de Órdenes de Pago, se tiene como principal función la de proporcionar al Usuario Administrativo la capacidad de poder dar de alta o registrar una orden de pago, para los distintos tipos de conceptos u opciones de gastos que pueden realizarse para un centro de costo determinado y para un beneficiario autorizado.

Basado en los conceptos comentados, por ejemplo, para el escenario de Registro de Órdenes de Pago en particular (R3.1), se pueden identificar las siguientes clases:

- Empleado.
- Beneficiario.
- Concepto Tipo Gasto.
- Orden Pago.
- Registro Orden Pago.
- Oficina Centro Costos.
- Concepto Gasto Oficina.
- Administración Orden Pago.

Creación de Paquetes.

El siguiente paso después de tener identificadas todas las clases relacionadas con el sistema, es agruparlas en paquetes, con el fin de facilitar su utilización, poder darles mantenimiento y reutilizarlas. Además, la representación lógica que puede proporcionar el diagrama de paquetes es darnos una mejor visión de cómo estarán interactuando las diferentes clases dentro del sistema.

Para el escenario del Registro de Órdenes de Pago, en la cual se identificaron las clases: Empleado, Beneficiario, Concepto Tipo Gasto, Orden Pago, Registro Orden Pago, Oficina Centro Costos, Concepto Gasto Oficina y Administración Orden Pago; éstas se agruparon en

tres paquetes principales: los que son propios de la compañía aseguradora (Compañía), los que contiene información sobre gente o personas (Personas), y aquellas clases que funcionan como interfaces para los actores del sistema (Interfaces).

Las clases Concepto Tipo Gasto, Orden Pago, Oficina Centro Costos, Concepto Gasto Oficina y Administración Orden Pago, se integraron al paquete de cosas relacionadas con la Compañía. Las clases de Empleado y Beneficiario quedaron en el paquete de Personas, y por último, en el paquete de interfaces está Registro Orden Pago.

En la Figura 3 se presenta el diagrama de paquetes a nivel general.

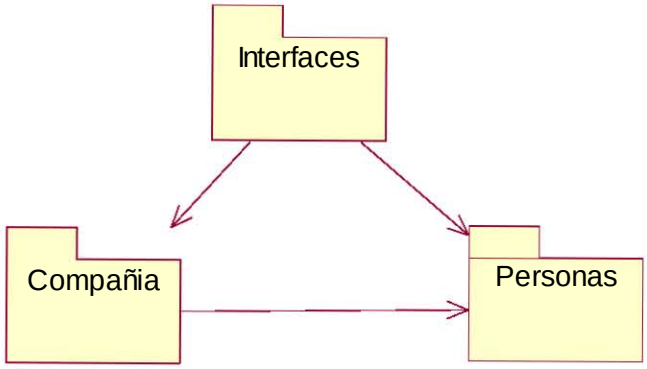


Figura 3. Diagrama de Paquetes

Diagrama de Clases.

Conforme se van agregando las clases al modelo y se van agrupando a su vez en paquetes, es necesario ir representándolas en un diagrama de clases.

Un diagrama de clases a nivel global o principal puede representar el nivel de todos los paquetes del sistema, y a su vez cada paquete tendrá su propio diagrama principal de clases en el cual se presentará una visión, también general, de cada una de las clases dentro de cada uno de los paquetes.

El diagrama de clases principal para el paquete que nombramos como las clases relacionadas con Compañía se presenta en la Figura 4. En él todavía no se muestran las relaciones entre cada clase ni los atributos ni las operaciones, ya que como se comentó, esta parte del modelado se refiere a la identificación y organización de las clases.

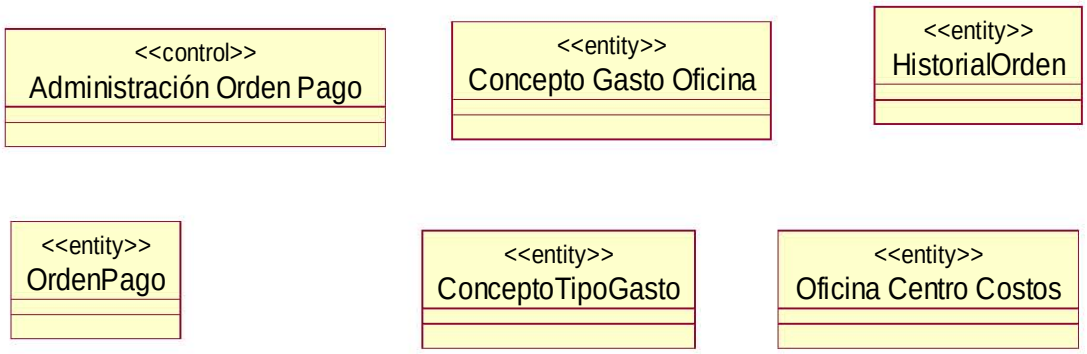


Figura 4. Diagrama de Clases

Modelando la interacción del sistema.

Hasta esta parte del modelado, se tienen creadas las clases y los objetos que se necesitarán en el sistema. En los diagramas se deberá mostrar la relación e interacción que tienen.

Los diagramas de Casos de Uso son utilizados nuevamente en esta etapa, debido a que representan los diversos escenarios que el sistema tendrá durante su funcionamiento. Para modelar la interacción del sistema y de cada uno de los objetos y clases que intervienen en estos escenarios se utilizan los diagramas de Secuencia y Colaboración.

Creando Diagramas de Secuencia.

El diagrama de secuencias nos mostrará la interacción de los objetos y clases que intervienen dentro de un escenario determinado del sistema y durante una secuencia de tiempo. Esta interacción se determina mediante mensajes o llamadas que deben existir entre los objetos, a fin de que la secuencia del proceso pueda realizarse de manera lógica.

Por ejemplo, el diagrama de secuencia de la Figura 5, muestra un escenario para el caso de uso del requerimiento R3.3, que se refiere a la funcionalidad de afectación contable a los sistemas centrales de la organización. Esta afectación se lleva a cabo una vez que las órdenes autorizadas por el área financiera son aplicadas por medio de pagos por transferencia bancaria o por medio de cheques. El proceso envía la información contable a los sistemas centrales de la compañía y a su vez da por terminada o aplicada esa orden.

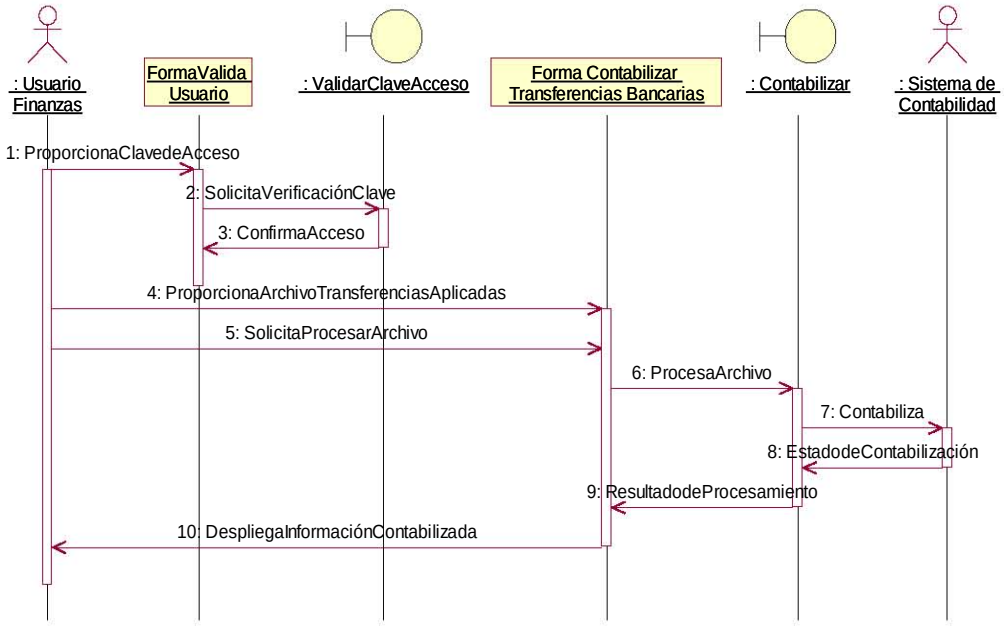


Figura 5. Diagrama de Secuencias

Creando Diagramas de Colaboración.

Al igual que los Diagramas de Secuencia, los Diagramas de Colaboración permiten ver la interacción de los objetos que intervienen dentro de un escenario determinado del sistema; sin embargo su representación es un poco más general y no se considera una línea de tiempo determinada.

El diagrama de colaboración para el mismo caso descrito del requerimiento R3.3 se muestra en la Figura 6.

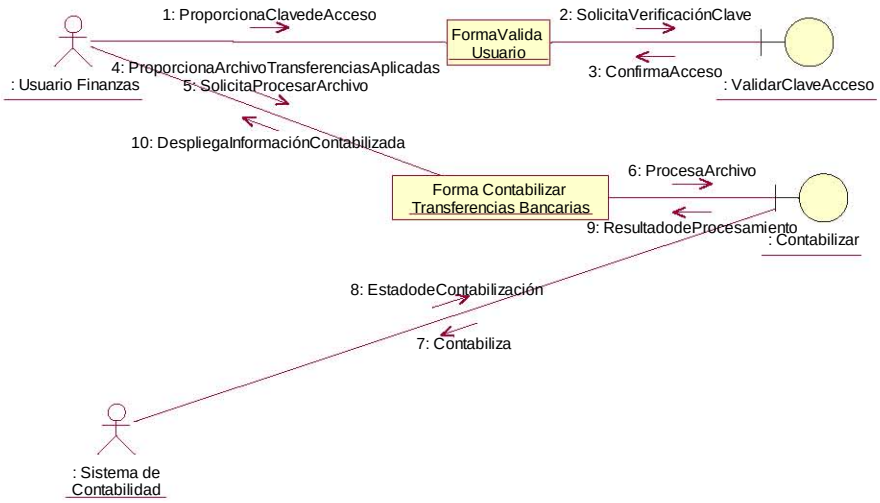


Figura 6. Diagrama de Colaboración

En lo que se refiere a la parte de la estructura de las clases, se agregaron los atributos y operaciones basándose ya sea en el planteamiento del problema, en los requerimientos del sistema o en el flujo de eventos descritos en los casos de uso.

Tomando como ejemplo uno de los requerimientos principales del sistema y que son mencionados al principio del capítulo, se tiene que las Órdenes de Pago se deben controlar por centro de costos, es decir, las órdenes se deben controlar y organizar dependiendo de la oficina y área a la que pertenece el empleado que está registrando la orden, por lo que tanto la clave de oficina como la del área deben formar parte de los atributos de la clase Orden de Pago.

Partiendo de una vista del Diagrama Principal de Clases, en la cual se habían agrupado aquellas clases dentro del paquete relacionado con conceptos propios de la organización o compañía, el diagrama de la Figura 7, muestra las clases a las que se han agregado sus atributos y operaciones.

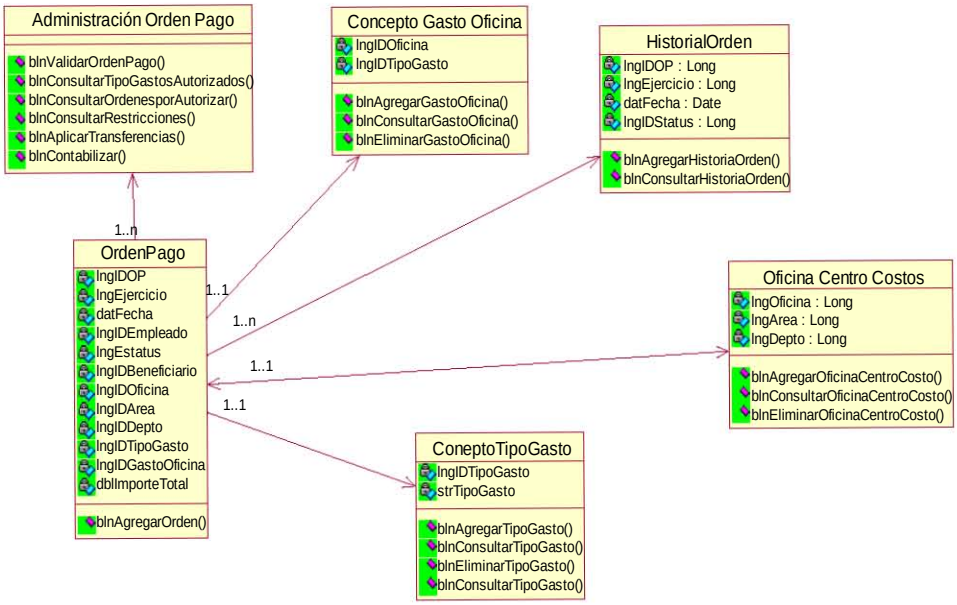


Figura 7. Diagrama de Clases / Atributos y Operaciones

Modelando la Arquitectura del Sistema.

Hasta este momento, se ha analizado y diseñado el comportamiento del sistema, se han creado y definido todos los objetos y clases que se requieren y la relación que existe entre cada uno de ellos y su respectivos atributos, sin embargo, no hemos modelado todavía la forma en que estos objetos y clases se construirán, qué tipo de tecnología o lenguaje se utilizará para su codificación y si podrán o no ser implementados físicamente a fin de cumplir con la funcionalidad esperada.

Dado lo anterior, es necesario contar con el modelado o el diseño de la Arquitectura. No es una tarea fácil, ya que para ello se deben analizar nuevamente los diagramas que modelan el comportamiento, construir prototipos para verificar que las decisiones tomadas en el diseño se cumplan, e inclusive dentro de las etapas mismas de desarrollo, corregir el diseño o crear nuevos objetos o clases para que las funcionalidades esperadas se satisfagan.

Para el modelado de la Arquitectura, se utilizó el proceso de vistas 4+1 sugerido por UML (Figura 8), en el cual la Arquitectura no se considera como un diagrama unidimensional, sino que es visto como la concurrencia de múltiples vistas.

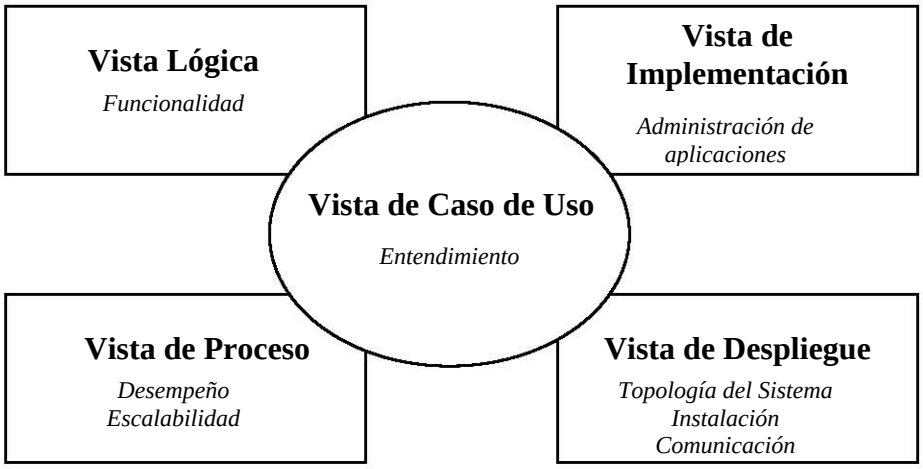


Figura 8. Vista de Procesos 4+1

La Vista Lógica será proporcionada por el Diagrama Principal de (Figura 9), el cual contiene las clases agrupadas y sus relaciones, y representa de manera abstracta al sistema para ser desarrollado.

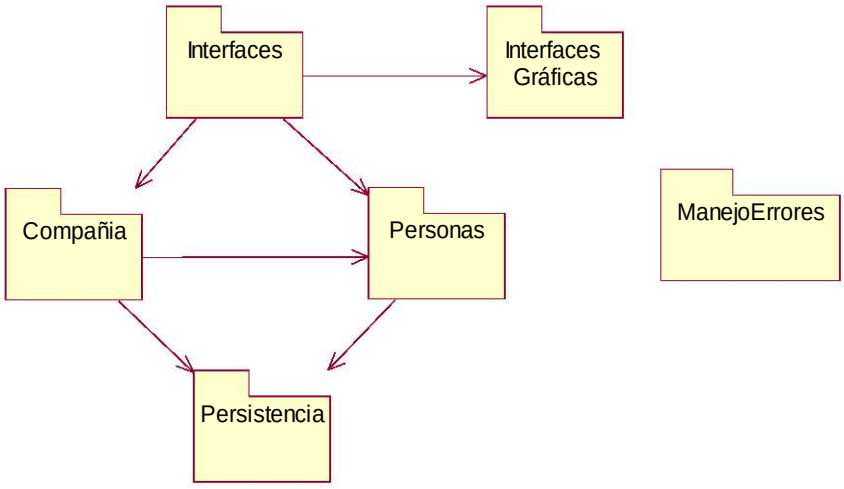


Figura 9. Diagrama de Vista Lógica

Para la Vista de Implementación se utilizó un Diagrama de Componentes a nivel general (Figura 10), en el cual se definieron paquetes para representar una parte física del sistema.

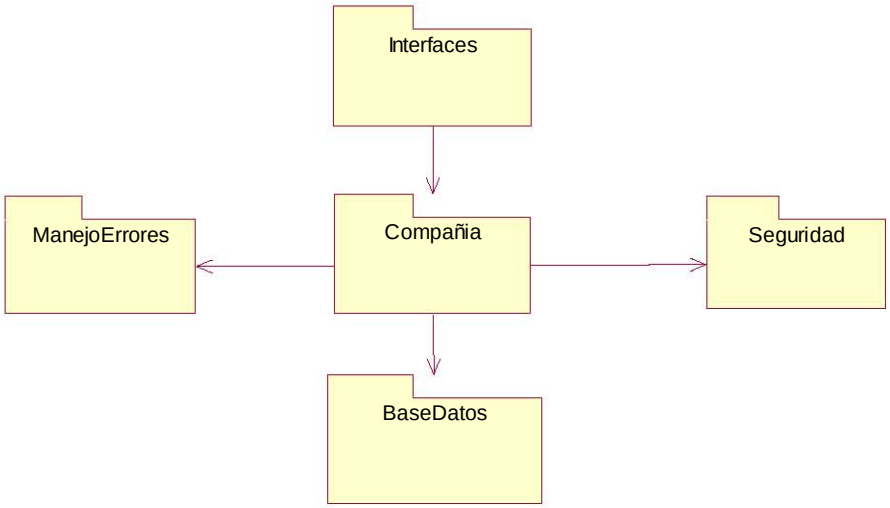


Figura 10. Diagrama de Vista de Implementación

Partiendo del diagrama anterior, se agregaron los componentes que estarán contenidos en cada uno de los paquetes y que representarán las piezas de software a desarrollar con el lenguaje definido para el proyecto; en este caso fue Visual Basic, pero que puede aplicar de manera similar para cualquier otro lenguaje como: C++, Java o C # (Figura 11).

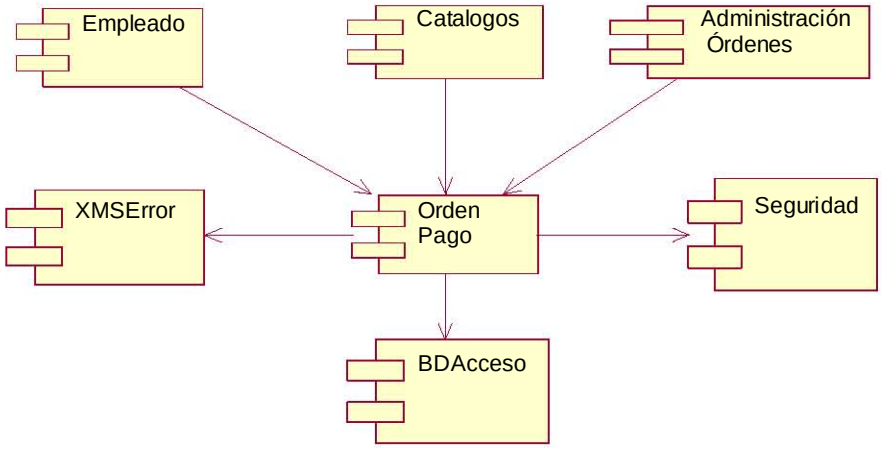


Figura 11. Diagrama de Vista de Implementación 2

La Vista de Proceso se enfoca en requerimientos tales como desempeño, confiabilidad, escalabilidad, integridad y administración del sistema. La información para esta vista es creada como un Diagrama de Componentes (Figura 12), el cual contendrá componentes que representen ejecutables o componentes en tiempo de ejecución.

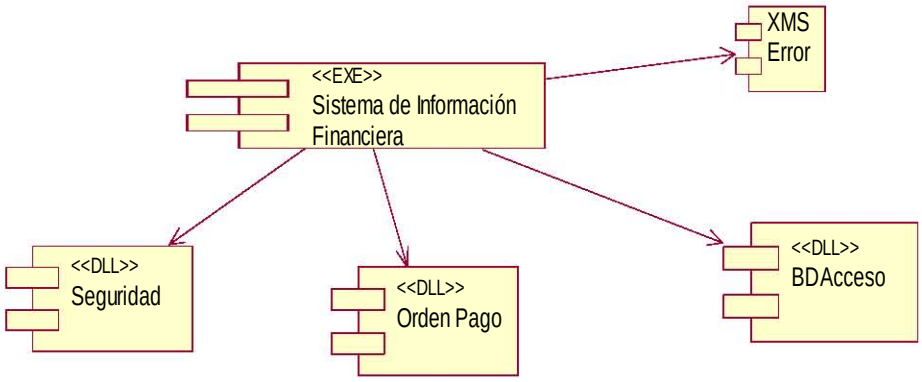


Figura 12. Diagrama de Vista de Proceso

La vista de Distribución se crea a partir de los Diagramas de Distribución (Figura 13), en la cual se visualiza la distribución de los componentes a través de la organización. Este diagrama permite entender la topología del sistema y es útil para visualizar los procesos ejecutables.

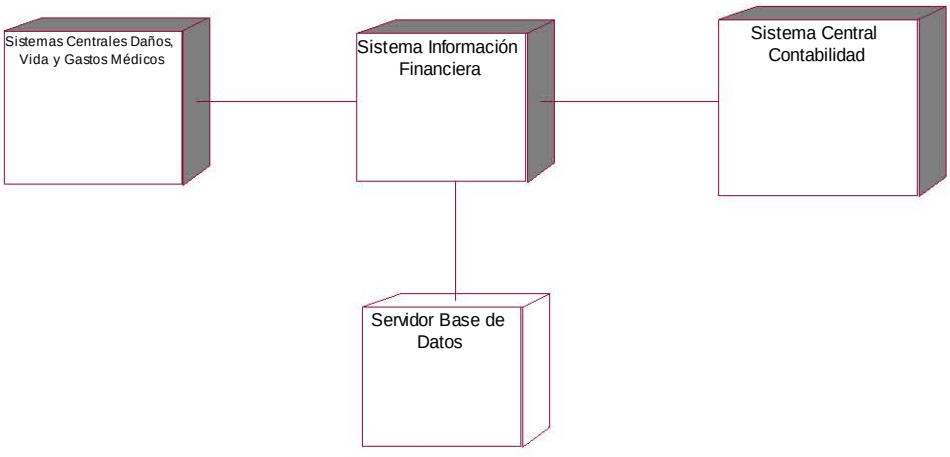


Figura 13. Diagrama de Vista de Distribución

Por último, la vista de Caso de Uso es simplemente crear sus instancias o escenarios a fin de demostrar y validar las vistas mencionadas en su conjunto. En esta última parte son creados los diagramas de secuencia para representar la relación objeto-escenario y los diagramas de colaboración para la relación objeto-interacción.

Partiendo de los modelos descritos en esta etapa, en el siguiente capítulo se describirán los aspectos más importantes de las fases de desarrollo o construcción del sistema, y finalmente su implementación.

CAPÍTULO III

DESARROLLO E IMPLEMENTACIÓN

En el capítulo anterior se describieron algunos de los procesos más importantes que fueron utilizados para el análisis y diseño del proyecto para el Sistema de Información Financiera (SIF) para Generali de México.

UML proporciona las herramientas necesarias para modelar el comportamiento, la estructura y definir los objetos y clases que son necesarios para que el sistema pueda cumplir con lo esperado; se pueden definir aspectos como las operaciones y las variables de entrada y de salida. Sin embargo, es durante la fase de desarrollo e implementación donde todos estos requerimientos tienen que ser bien entendidos y traducidos en código ejecutable y con buena calidad para que las funcionalidades esperadas puedan llevarse a cabo.

Para el proyecto del SIF, fue necesario tener en consideración al equipo de desarrollo durante ciertas etapas del Análisis y Diseño, a fin de ver la factibilidad y delimitar correctamente los alcances de algunos requerimientos, así como el diseño e implementación de la estructura de la base de datos. De cualquier manera, durante toda la vida del proyecto se corrigen muchas de las cosas de diseño que no pueden visualizarse en su oportunidad.

Cabe mencionar que UML no define un procedimiento para determinar qué objetos o cuales funcionalidades se deben ir construyendo, pero si sugiere que este proceso debe de realizarse de forma iterativa, es decir, la fase de construcción y desarrollo debe de dividirse en varias etapas o iteraciones en las cuales al final de cada etapa deben existir productos a entregar o mejor dicho, software que contendrá módulos o funcionalidades a probar ya sea por parte de una persona designada específicamente para ello o entregarlo a los usuarios para que sea probado.

Dado que el proyecto del SIF fue un sistema desarrollado con orientación a objetos, éste se diseñó en capas de tal manera que hubieran objetos o clases que son propias del negocio y que

guardan todas las políticas de las organización; objetos con funciones de seguridad y acceso al sistema, objetos de persistencia para las operaciones comunes como el mantenimiento a catálogos y objetos que establecen funcionalidades dentro del servidor de base de datos.

A continuación se describen algunas de las funcionalidades y los algoritmos desarrollados para resolver cada una de ellas.

Aspectos de Seguridad.

Para la parte de seguridad, por ejemplo, se implementaron dos procesos, uno para la validación del nombre del usuario y la clave de acceso; una vez validadas, se deberían de cargar todos los permisos que se establecieron para ese usuario y conservarlos durante toda su sesión. El segundo proceso se refiere al algoritmo utilizado para cifrar y descifrar la clave de acceso.

Los procesos se implementaron a nivel interfaz con el usuario, solamente eran permitidos letras y números al momento de crear la clave de acceso y no había restricción en el número de intentos si el usuario introducía equivocadamente su clave de acceso.

Proceso Valida Empleado.

1. Entrada: Nombre de Usuario, Clave de Acceso.
2. Salida: Carga el Perfil de Acceso del Usuario y Pantalla Principal.
3. Método:
    1. Se solicita el nombre del Usuario y clave de acceso
    2. Se llama a la clase de Seguridad para validar la existencia del Usuario. La clase de Seguridad regresa un valor de tipo long.
    3. Evaluar el dato de tipo Long.
        1. Mayor que 0, se utiliza la clase Empleado, para conseguir toda la información del perfil del usuario y niveles de acceso al sistema.
        2. En caso contrario, los posibles valores que puede regresar la clase de Seguridad son:
            1. -1 = No existe el empleado.
            2. -2 = Empleado Inactivo.

3. -3 = Clave de acceso incorrecta.
4. Se inicia la sesión si el dato que regresa la clase de Seguridad es mayor que 0.

Proceso de Cifrado (Por Sustitución).

1. Entrada: Clave de Acceso a cifrar de tipo String.
2. Salida: Clave de Acceso cifrada de tipo String.
3. Método:
    1. Se recibe la cadena de tipo string que contiene la clave de acceso a cifrar.
    2. Se lee la cadena carácter por carácter asociada a la siguiente función de transformación.
        1. Sea A= $\{a_0...a_{n-1}\}$ la clave a encriptar.
        2. Sea T(A) = $a_i \in A$, $T(a_i) = ASCII(a_i) * 57 + 121$.
        3. La nueva Clave cifrada será P= $\{T(a_0)\ldots T(a_{n-1})\}$.
4. Se regresa la cadena con la clave de acceso cifrada.

Cabe señalar que la seguridad del esquema de cifrado que se utilizó, a pesar de que funciona y cumple con los requerimientos del proyecto, tiene algunas debilidades, ya que se basa en uno de los primeros esquemas de seguridad utilizados y que se denominan de tipo monoalfabético. La forma general de este tipo de cifrado es:

$$c(x) = ax+b$$

y requiere que los parámetros de la transformación, que son 57 y 121, se mantengan en secreto, por lo que para romper la seguridad simplemente hay que despejar $ASCII(a_i)$ en la expresión del algoritmo.

Durante la construcción del sistema y antes de contar con el análisis descrito, no se tenía contemplado un esquema de seguridad mucho más sólido. La seguridad de un esquema de cifrado nunca debe basarse en mantener secretos los elementos del algoritmo o función de transformación si no en una clave.

Como mejoras al sistema, se recomienda utilizar otro esquema de seguridad basado en una función hash estándar, como el algoritmo MD5, el cual proporciona un nivel de cifrado a 128bits, que es el nivel de seguridad recomendable y sería prácticamente imposible romperlo.

El proceso sería similar y lo que se tendría que guardar como clave de acceso sería lo que se obtenga de aplicar la función hash estándar al dato que se quiere cifrar, es decir MD5(a) y para autentificar a un usuario sólo se tendría que comparar el resultado de aplicar MD5 a la cadena tecleada como clave de acceso y compararla con la almacenada. Si coinciden, se permite el acceso, en caso contrario no.

Aspectos de Acceso a Datos.

Durante el diseño de las funcionalidades que estaría realizando el SIF, existiría información generada por la operación de los usuarios así como la que se originaría por la interacción con otros sistemas centrales de la organización.

Para garantizar que toda esta información fuera registrada de manera consistente en la base de datos, se creo una Clase de Acceso de Datos, la cual se encargaría de contar con todos los procesos necesarios relacionados con la base de datos, como son: establecer la conexión con la base de datos; agregar, actualizar y eliminar registros, realizar consultas de información, entre otras.

Proceso de Conexión a la Base de Datos.

1. Entrada: Cadena de Conexión, con los datos del Servidor, Base de datos, Usuario y clave de acceso a la base de datos.
2. Salida: Si la conexión fue exitosa regresa un valor de verdadero, en caso contrario regresa un valor de Falso.
3. Método:
    1. Crear un Objeto de tipo Conexión.
    2. Pasar la cadena de conexión.
    3. Ejecutar el proceso de Abrir la Base de datos.
    4. Evaluar el estado de la operación abrir.

5. Si se logró la conexión con la base de datos, se regresará un valor de Verdadero.

4. En caso de Error con la conexión se regresará un valor de Falso.
5. Los posibles errores que pueden pasar con la conexión a la Base de datos son los siguientes:
    1. "Error en la cadena de conexión o faltan parámetros".
    2. "Los datos del Usuario o clave de acceso a la Base de Datos no son válidas".

Proceso de Afectación a la Base de Datos.

1. Entrada: Sentencia de SQL de tipo String.
2. Salida: Verdadero si la sentencia se ejecuto sin problemas y la Base de datos fue afectada. Falso en caso contrario.
3. Método:
    1. Crear la sentencia SQL de afectación a la base de datos. Este puede ser de inserción de datos, de actualización o de eliminación de registros.
    2. Crear una Conexión a la base de datos con el proceso de Conexión.
    3. Crear un objeto que utilice los métodos de ejecución de sentencias SQL.
    4. Ejecutar la sentencia SQL.
    5. Evaluar el resultado de la ejecución del paso anterior.
    6. Si la sentencia se ejecutó sin problemas, regresar un valor de Verdadero confirmando que la sentencia se ejecutó con éxito.
    7. En caso de Error:
        1. Error 1: de Sintaxis. La sentencia SQL no se reconoció como sentencia SQL y no se pudo ejecutar debidamente.
        2. Error 2: La sentencia SQL, no tiene errores de sintaxis, sin embargo no son correctos o no se encontraron en la Base de datos, el nombre de tablas o campos que especifica la sentencia SQL que se deberían de afectar.

3. Error 3: Se quiso agregar información duplicada o por la definición de índices en alguna tabla, el registro a insertar o actualizar crearía valores duplicados.
4. Error 4: No se puede eliminar el registro debido a que por integridad de la base de datos, existe información ligada a él.

Los Errores en el diseño de la base de datos generan las excepciones anteriores, las cuales pueden ser controladas ya sea por programación o corrigiendo el diseño de la base de datos. El uso de Bases de datos relacionales como Oracle o Microsoft SQL poseen la capacidad de definir índices, llaves primarias o secundarias y relacionarse entre sí a fin de que el motor de la base de datos sea quien controle la integridad de la información al momento de insertar un nuevo registro o borrar información.

Otro de los problemas a enfrentar al momento de implementar el manejo de información con la base de datos, es el de resolver los problemas de concurrencia. Por ejemplo para el caso particular del SIF, se implementó una solución en la parte de la capa de negocios para garantizar la integridad en el registro de órdenes de pago y que al momento de guardar la orden no se pudiera presentar el problema de que dos usuarios de manera simultánea pudieran registrar en la Base de datos su información con el mismo número de orden y que es comentado a continuación.

Aspectos de Negocio.

Para garantizar que no se pudieran registrar dos órdenes de pago con el mismo número, se utilizó el siguiente proceso, el cual creaba una transacción a la base de datos. Si existía algún error al momento de guardar la información, toda la transacción es rechazada y garantizaba que la información no quedara registrada de forma incompleta o inconsistente.

Al utilizar transacciones, el motor de la base de datos ejecuta las operaciones de la transacción del primer usuario que desea guardar su información, y si recibe otra transacción

en ese momento, esta última no es ejecutada hasta el momento de que la primera transacción termina. En ese momento la base de datos inicia con la siguiente transacción.

Proceso Guardado de Nuevas Órdenes de Pago.

1. Entrada: Información capturada por el Usuario (Centro de costos, Beneficiario, Conceptos o Detalle de la Orden, Tipo de Pago en Cheque o Transferencia, Tipo de Moneda Dólares o en Pesos, Fecha Solicitud).
2. Salida: Verdadero y Número de Orden si el registro se realizó, Falso si no se pudo registrar la orden.
3. Método:
    1. Iniciar Declaración de Transacción (BeginTrans).
    2. Obtener el último número de orden registrado en la base de datos.
    3. Crear sentencia SQL para los datos principales o de cabecera de la orden.
    4. Obtener los rubros contables por cada uno de los conceptos de la orden.
        1. Iniciar con el primer concepto de la Orden.
        2. Crear sentencia SQL para consultar los rubros contables del concepto.
        3. Llama Proceso de Acceso a Datos para ejecutar la sentencia SQL de consulta.
        4. Crear sentencia SQL para insertar los rubros contables del concepto.
    5. Llama Proceso de Acceso a Datos y ejecuta las sentencias SQL creadas.
    6. En caso de que todas las sentencias SQL se hayan realizado sin errores, realiza la finalización de la transacción (CommitTrans) y devuelve el valor de Verdadero.
    7. En caso de Error:

1. Error 1: Devolver el valor de Falso y declarar la transacción finalizada. Los datos que se hayan registrado hasta el momento de que se presentó la falla se descartan y la base de datos no se afecta (RollBack).
2. Error 2: Los posibles errores de acceso a la Base de datos son controlados por los respectivos procesos de la Clase Acceso de Datos y son regresados como mensaje de error.

Una de las reglas definidas para el proceso de autorización de órdenes, es que deberían contar con un historial de autorización primeramente por parte del área contable y posteriormente por parte del área de finanzas. Esta definición se controló por medio de una tabla en la Base de datos que guardaba todo el historial de las órdenes, inclusive de aquellas que ya habían sido contabilizadas después de haber generado su respectivo pago, lo cual permitió generar información de auditoria para aquellas órdenes que se pagaban con cheque y que por algún problema en el momento de la impresión del documento, se requería nuevamente reimprimirlo.

El proceso de autorización que se presenta se realiza al momento de que la persona responsable de finanzas ha seleccionado en la pantalla todas las órdenes que desea autorizar y para las cuales ha seleccionado la cuenta bancaria de la empresa con la cual desea que se paguen.

Proceso de Autorización de Órdenes (Finanzas).

1. Entrada: Órdenes de Pago registradas en el SIF y autorizadas por Contabilidad, Órdenes de Pago de Siniestros Daños y por concepto de Gastos Médicos y Vida importadas de los Sistemas Centrales.
2. Salida: Estatus de Autorización, asignación de Número de Cheque y generación de información para pago por transferencia bancaria.
3. Método:

1. Iniciar con cada una de las órdenes seleccionadas por el usuario para su autorización.
2. Validar el tipo de Pago Cheque o Transferencia.
    1. Si es Cheque validar la Cuenta Bancaria y asigna consecutivo de cheque.
    2. Si es transferencia, asignar información de la cuenta bancaria.
3. Crear sentencia SQL para actualización de información en la base de datos y guardar en historial.
4. Llamar proceso de Acceso a Datos y ejecutar sentencia SQL
5. Continuar con la siguiente orden.
6. Crear sentencia SQL de todas las órdenes autorizadas para ser pagadas por transacción bancaria agrupadas por banco.
7. Llamar proceso de Acceso a Datos y ejecutar sentencia SQL.
8. Llamar proceso de generación de archivos de transferencia.
9. En caso de Error:
    1. Error 1: No pudo ser autorizada una orden seleccionada.
        1. Crear Sentencia SQL para registrar el error y la orden afectada.
        2. Llamar al proceso de Acceso a Datos para ejecutar la sentencia.
    2. Error 2: No contar con una cuenta bancaria habilitada o dada de baja para la autorización de pagos, o no contar con un suficiente número de cheques conforme al número de órdenes seleccionadas.
    3. Error 3: Los errores de acceso a la Base de datos son controlados por la Clase de Acceso de Datos.

La codificación de los procesos brevemente descritos se muestran en el Apéndice a fin de ver el detalle de cómo se implementaron.

Como se señaló al principio de este capítulo, la parte de implementación depende mucho de las bases sólidas de programación con que cuente la gente de desarrollo y el buen entendimiento que se tenga sobre cada una de las funcionalidades que se elaboraron en la fase de análisis y diseño, por lo que al llegar a esta etapa es nuevamente se requiere de realizar procesos de análisis y corregir el diseño al momento de la implementación.

Con lo anterior, no se pretende decir que las fases de análisis y diseño no cumplen con el propósito deseado. El proceso de desarrollo de software es un proceso que debe ser constantemente revisado en cada una de sus etapas durante todo el ciclo de vida del proyecto.

Uno de los aspectos que se deben cuidar para la parte de implementación, es que realmente el código generado pueda tener las características de poder ser reutilizado y que se le pueda dar mantenimiento, que son algunas de las consideraciones de la programación orientada a objetos.

Para el proyecto del SIF, se fueron liberando los módulos conforme a lo acordado con los usuarios, es decir, la aplicación no se probó hasta que todo estuviera concluido, sino que fue probándose módulo por módulo en el siguiente orden.

1. Seguridad y Mantenimiento a Catálogos.
2. Registro de Órdenes de Pago/Anticipo de Gastos.
3. Autorización de Órdenes.
4. Generación de Archivos de Transferencias y Aplicación de Pagos por Sistemas Bancarios.
5. Afectación contable después de aplicar pagos.
6. Importación de Información de Pago de Seguros de Daños y Vida de los Sistemas Centrales.

Para cada una de las pruebas se realizaron matrices correspondientes basados en los Casos de Uso, ya que son en estos documentos donde se describe la funcionalidad esperada por los usuarios.

Una vez concluido el proceso de Desarrollo de todos los módulos del proyecto del SIF, se elaboró un plan con la respectiva área de Liberación, para realizar implantación física de la aplicación, es decir, su distribución hacia los usuarios con la respectiva documentación y la creación de la base de datos en los servidores de producción.

La distribución de la aplicación fue realizada por el área de Soporte Técnico de la compañía a nivel nacional, y la puesta en marcha de los servidores productivos estuvo a cargo del área de Redes de Computo.

CONCLUSIONES Y RECOMENDACIONES

La experiencia que se obtuvo durante el proyecto presentado, así como los conocimientos aportados para su realización, no se hubieran logrado de no contar con las bases que me proporcionó la carrera, como pueden ser el uso de la lógica matemática para resolver muchos de los problemas planteados y los conocimientos en programación que se aprendieron en materias como Organización y Programación Administrativa e Instrumentos y Programas de Cálculo.

En relación al trabajo presentado y al uso del Lenguaje Unificado de Modelado UML, se puede concluir lo siguiente:

UML se ha venido estableciendo como una notación estándar para el desarrollo de software, sin embargo no hay que pensarlo como una metodología ya que no dicta los pasos o etapas a realizar en la elaboración de cualquier proyecto de desarrollo de software ya sea creado desde la nada o continuar con el mantenimiento de algún sistema existente.

Muchos de los artefactos de UML son útiles para ayudar en el proceso de pruebas de los sistemas. Hoy en día, las exigencias en este sentido son mayores debido a que los procesos operativos de las empresas están cada vez dependientes de procesos automatizados.

El desarrollo orientado a objetos se ha establecido como una estrategia en la creación de sistemas, ya que las compañías requieren que sus sistemas puedan ser mantenidos o reutilizados para futuros cambios.

La importancia de elaborar las especificaciones con UML es contar con una documentación que ayuda a detectar detalles que se pasaron por alto o inconsistencias que se escapan a la vista y que pueden ser revisadas constantemente durante todas las etapas del proyecto para su corrección. UML ayuda en este sentido, sin embargo no se debe tomar como la única herramienta para iniciar un proyecto de desarrollo de sistemas.

UML, combinado con un buen proceso de administración de proyectos puede reducir en mucho el costo que puede ocasionar el desarrollo de sistemas para las empresas, ya que no solamente basta con contar con los requisitos iniciales para comenzar un sistema. Es bien sabido que durante todo el ciclo de vida de un proyecto de sistemas, los usuarios constantemente están cambiando los requisitos, por lo que la capacidad para mitigar ese tipo de riesgos y lo que conlleva implementarlos en código fuente depende mucho de la capacidad y experiencia de la gente responsable de los desarrollos.

APÉNDICE

El CD contiene para su consulta la siguiente información en formato PDF

- Documentos y diagramas elaborados durante la fase de análisis y diseño
- Documento para ver la implementación de las principales clases del proyecto.
- Modelo de Entidad Relación de la Base de Datos en formato PDF.
- Código original del proyecto que contiene además de las clases comentadas, la implementación de las pantallas e interfaces del Usuario para poder verse en Microsoft Visual Studio 6.0

Casos de Uso

Control de Accesos a Usuarios

Versión: 1.2
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 14 de Abril de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
	1.1	Actualización de Versión y Fechas de modificación	AMP	CGC
	1.2	Revisión narrativa.		MSE

Casos de Uso

Proyecto

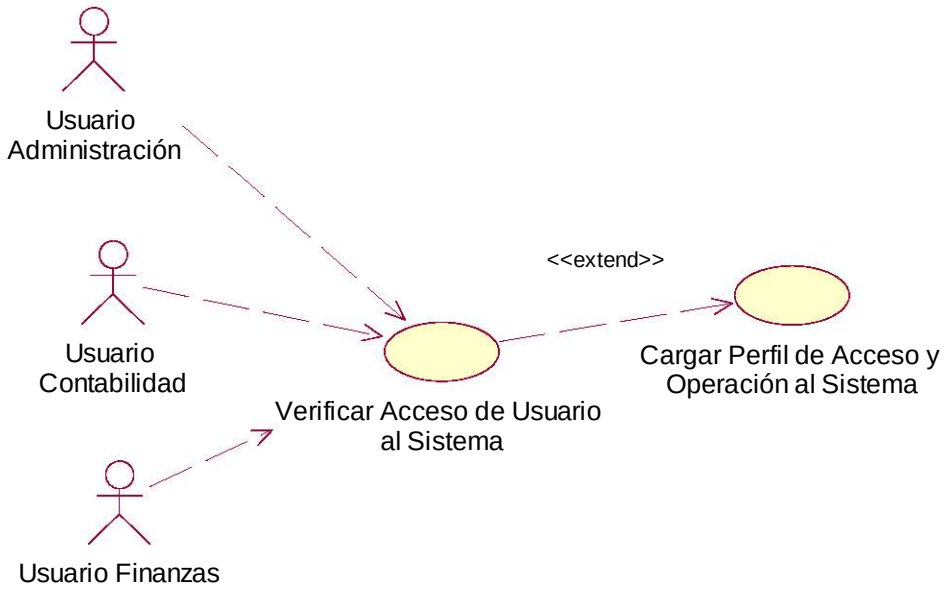
Contenido

Casos de Uso

Proyecto

R1.1. Control de Accesos a Usuarios.

Diagrama



Descripción

El caso de uso inicia con cualquiera de los usuarios administrativo, de contabilidad o finanzas que desean ingresar al sistema.

Precondiciones

Usuario debe de contar con cuenta y clave de acceso proporcionada por el Administrador del Sistema.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Inicia el sistema	2. Presenta Pantalla para ingresar la cuenta del usuario y clave de

	acceso.
3. Proporciona cuenta de usuario y clave de acceso	4. Valida que la cuenta de usuario, en caso contrario Flujo Alternativo A1. Valida clave de acceso. En caso contrario flujo alternativo A2. Realiza carga del perfil del usuario para mostrar los módulos autorizados del sistema a operar. Inicia Pantalla principal de operación del sistema.
	5. Fin del Caso de Uso

Flujo Alternativo

A1. Cuenta de Usuario Inválida. Solicitar ingresarla nuevamente o Terminar Caso de Uso.

A2. Clave de Acceso inválida. Solicitar ingresarla nuevamente o Terminar Caso de Uso

Requerimientos Especiales

Ninguno

Post-Condiciones

6. Acceder al sistema

Puntos de Extensión

Casos de Uso

Registro de Operaciones Realizadas del Usuario

Versión: 1.4
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Abril de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
17/04/2004	1.1	Modificaciones Narrativa	JE	CGC

Casos de Uso

Proyecto

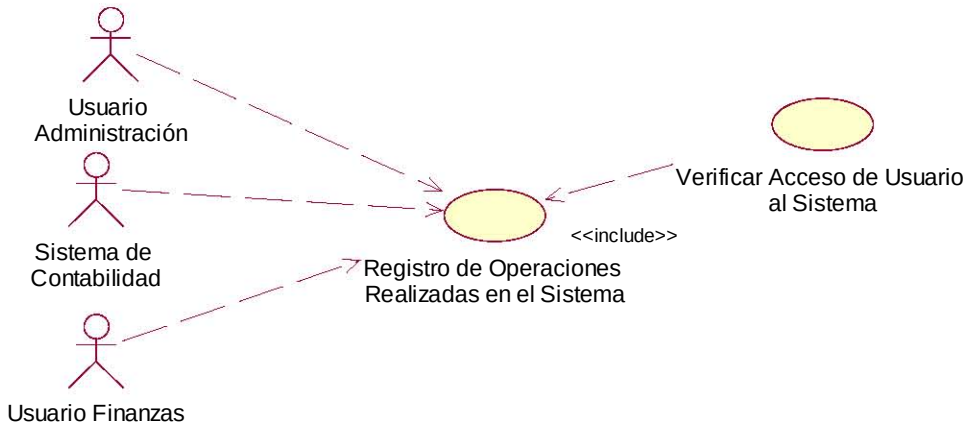
Contenido

Casos de Uso

Proyecto

R1.2. Registrar operaciones realizadas por el usuario

Diagrama



Descripción

El Caso de Uso es iniciado por cualquiera de los Usuarios durante toda su operación con el sistema.

Precondiciones

El usuario debe haber sido validado antes de iniciar la operación con el sistema

FLUJO BASICO

Acción del Actor	Acción del Sistema
Inicia con la selección de alguno de los módulos a los que tiene acceso el usuario.	1. Sistema registra en una bitácora o log de operaciones cada una de las actividades del usuario dentro del sistema, registrándose Fecha, Hora, Usuario y Operación realizada.
2. El usuario puede dar por terminado en cualquier momento la operación con el sistema	Fin del Caso de Uso

FLUJO ALTERNO

Ninguno

Requerimientos Especiales

Ninguno

Post-Condiciones

Operación registrada.

Casos de Uso

Administración Catálogo de Usuarios

Versión: 1.2
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 14 de Abril de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
	1.1	Actualización de Versión y Fechas de modificación	AMP	CGC
	1.2	Revisión narrativa.		MSE

Casos de Uso

Proyecto

Contenido

Casos de Uso

Proyecto

R2.1. Administración Catálogo de Usuarios.

Diagrama

Descripción

El caso de uso inicia con el Usuario de Sistemas. Proporciona la capacidad de dar de alta, modificar o inhabilitar cuentas de usuario en el sistema.

Precondiciones

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Clic en el botón “Usuario Nuevo”	2. Se despliegan los campos para registro de un nuevo usuario del sistema.
3. El administrador deberá especificar el nombre, apellido paterno y apellido materno del usuario que será agregado al catálogo, también deberá indicar si estará activo habilitando la opción “Activo”. 4. El administrador deberá asignar una clave – no mayor a 13 caracteres- y contraseña –no mayor a 8 caracteres-.	

5. El administrador deberá indicar el centro de costo al cual pertenecerá el usuario.	
6. Una vez terminada la captura de información con clic en el botón de aceptar, el administrador deberá guardar el registro para que sea dado de alta dentro del catálogo de usuarios.	7. Valida El nombre, apellido paterno y materno no existan dentro del catálogo. La clave y contraseña asignadas no existan dentro del catálogo. En caso de existir alguna información previamente registrada Flujo Alterno 1
9. Termina Caso de Uso.	

Con formato: Numeración y viñetas

Con formato: Numeración y viñetas

Con formato: Numeración y viñetas

Flujo Alternativo

A1. Obtener el campo con la información inválida y solicitar su corrección al usuario.

Requerimientos Especiales

Ninguno

Post-Condiciones

Haber agregado, modificado o inhabilitado un determinado Usuario.

Puntos de Extensión

Casos de Uso

Mantenimiento Catálogos del Sistema.

Versión: 1.1
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 12 de Abril del 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
12/04/2004	1.1	Se corrige información del beneficiario.	HM	CGC

Casos de Uso

Proyecto

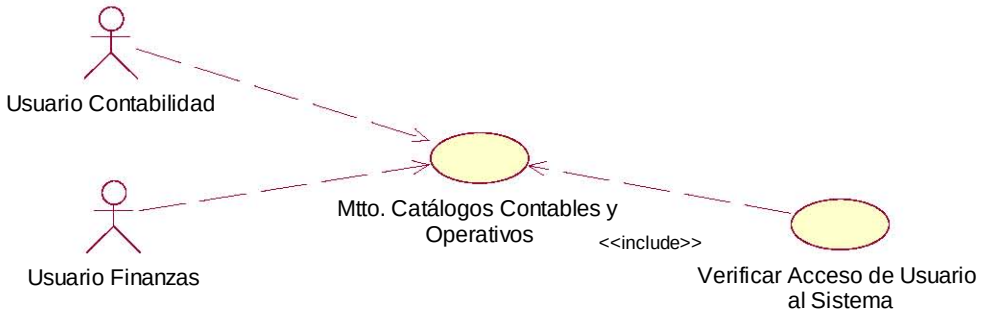
Contenido

Casos de Uso

Proyecto

R2.2.Mantenimiento Catálogos del Sistema.

Diagrama



Descripción

Caso de Uso iniciado tanto por el Usuario de Contabilidad como el Usuario de Fianazas. Permitirá Modificar tanto los catálogos contables, de información de beneficiarios, centros de costos e información de bancos con que opera la organización.

Precondiciones

Los usuarios deben contar con cuenta y clave de acceso válidas así como perfil para modificar la información de los catálogos de operación.

FLUJO BÁSICO

Acción del Actor	Respuesta del Sistema
1. El usuario solicita el Catálogo a administrar.	Sistema muestra la pantalla con la información del catálogo solicitado por el usuario.
1. El usuario tendrá para cada catálogo las opciones de Alta,	SI el usuario solicita la opción de Alta. Flujo alternativo A1. Si el

Eliminar y Modificación de registros	usuario solicita la opción de Modificación, Flujo alternativo A2. Si solicita la opción de Eliminar. Flujo alternativo A3.
2. El usuario sale de la opción de Catálogos.	Fin del Caso de Uso.

FLUJO ALTERNATIVO

A1. El sistema valida que la información no se encuentre repetida y confirma que el registro fue dado de alta exitosamente.

A2. Confirma la modificación del registro.

A3. Confirma que la información del registro no este ligado a alguna operación del sistema que dependa de este registro. Si existe alguna operación dependiente de este registro. Flujo alternativo A4. Confirma la eliminación del registro

A4. Envía mensaje diciendo que no se puede realizar la operación solicitada por el usuario.

Requerimientos Especiales

Ninguno.

Post-Condiciones

Registro dado de alta

Registro Modificado

Registro Eliminado

Casos de Uso

Registro de Órdenes de Pago

Versión: 1.3
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 13 de Abril de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
13/04/2004	1.1	Corrección Narrativa		CGC

Casos de Uso

Proyecto

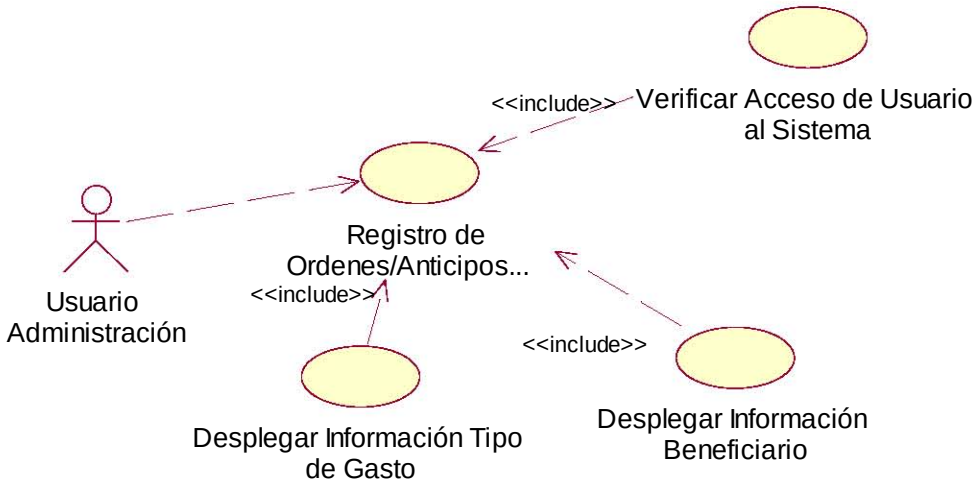
Contenido

Casos de Uso

Proyecto

R3.1.Registro de Órdenes de Pago.

Diagrama



Descripción

Permitirá le registro de órdenes de pago dentro del sistema a fin de darle trámite al pago de un servició u operación administrativa relacionada con las actividades de la compañía.

Pre-Condiciones

El Usuario deberá estar validado por el sistema y que su perfil de autorización permita registrar una orden de pago.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Usuario accesa a la pantalla de registro para órdenes de pago	2. Sistema presenta pantalla de registro de órdenes de pago. Se muestra la fecha actual del sistema. Se cargan de manera predeterminada los datos del centro del costo del usuario, con

	opción a modificar el centro si es que la orden va a ser registrada con aplicación a otro centro de costo. Los centros opcionales para el usuario están definidos en base a su rol.
3. Selecciona el centro de costos de la orden. 4. Selecciona el tipo de beneficiario.	5. Dependiendo del tipo de Beneficiario, Sistema despliega lo beneficiarios registrados así como el banco y la cuenta de depósito, en caso de contar con ella.
6. Selecciona Forma de Pago	7. Si el Beneficiario seleccionado contiene la información previamente capturada de la cuenta bancaria o CLABE. Flujo alternativo A1.. Puede seleccionar la forma de pago cheque.
8. Seleccionará el concepto (plantilla), proporcionará el importe dará clic en el botón de agregar.	9. El sistema calcula en base a las plantillas contables los rubros que se afectaran contablemente.
10. Podrá consultar los movimientos contables a afectar con clic en botón de detalle contable.	
11. Para registrar la orden el usuario da clic en botón de aceptar.	Si el tipo de pago seleccionado es por transferencia bancaria, y el beneficiario no tiene dada de alta su cuenta bancaria, Flujo Alternativo A2. Asigna No. de Orden y registra.
12. Usuario Sale de la Pantalla y Termina Caso de Uso.	

Flujo Alternativo

A1. El sistema sugerirá que la forma sea por transferencia bancaria.

A2. El sistema enviará el mensaje de que no se puede registrar la orden si el tipo de pago es por transferencia y el beneficiario no cuenta con sus datos bancarios respectivos para poder realizarse. El sistema sugiere opción de pago con cheque o terminar el caso de uso.

Post-Condiciones

Orden registrada

Puntos de Extension

Ninguno.

Casos de Uso

Extraer Información SISE

Versión: 1.2
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 19 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
09/05/2004	1.2	Modificación de Formato y Narrativa		CGC

Casos de Uso

Proyecto

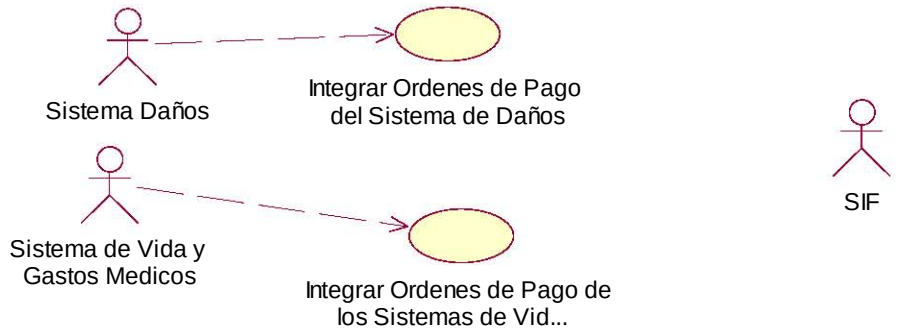
Contenido

Casos de Uso

Proyecto

R3.2.Integrar información de los Sistemas de Vida, Gastos Médicos y Daños.

Diagrama



Descripción

Caso de Uso iniciado por cualquiera de los sistemas centrales de Vida y Gastos Médicos, y Daños. Tiene la capacidad de importar la información de estos sistemas al Sistema de Información Financiera para que se le pueda dar trámite de pago.

Precondiciones

Que existan nuevas órdenes por concepto de pago de seguros de los sistemas centrales.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Sistemas Centrales(SISE) genera archivo con la información de la orden de pago y la envía a Servidor de SIF.	2. Proceso de carga, en el servidor de SIF, el cual esta monitoreando nuevos archivos de ordenes, detecta archivo nuevo.
	3. Proceso de carga asigna centro de costos en base al usuario de SISE, no. de orden de SIF, fecha, hora y registra en el sistema como orden de pago autorizada pendiente de seguimiento por parte de Tesorería.

	4. En el caso de que existan registros de ordenes de SISE que no se puedan procesar o registrar en el sistema por alguna causa, se continua con el flujo alternativo A1.
5. Termina caso de uso.	

FLUJO ALTERNATIVO

A1- Registros no Procesados	
Acción del Actor	**Acción del Sistema**
	1. Sistema registrará de manera independiente las órdenes de pago que presentaron problemas durante el proceso de importación.
2. Usuario podrá acceder dentro de la pantalla de listado de seguimiento de ordenes, a una sección de rechazos donde se mostrarán todas las solicitudes por no. de orden y causa por la cual no pudo ser registrada o importada dentro del sistema.	3. Presenta información de órdenes rechazadas en una nueva pestaña o sección dentro del listado para consulta general de órdenes. La información que se presenta será Fecha, No. de Orden, Beneficiario y Causa del rechazo, ordenada por Fecha con opción a reordenarla con clic del ratón por cada columna.
4. Revisa Información.	

Requerimientos Especiales

Ninguno.

Post-Condiciones

Orden de Pago SISE registrada en el sistema.

Casos de Uso

Proyecto

Orden de Pago SISE rechazada

Usuario dará seguimiento a la orden dependiendo del motivo del rechazo.

Casos de Uso

Exportación de Información Contable

Versión: 1.4
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 19 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
09/May/2004	1.1	Modificación Formato y Narrativa		CGC
15/May/2004	1.2	Revisión narrativa		MSE
19/05/2004	1.3	Modificar Flujo Básico		CGC

Casos de Uso

Proyecto

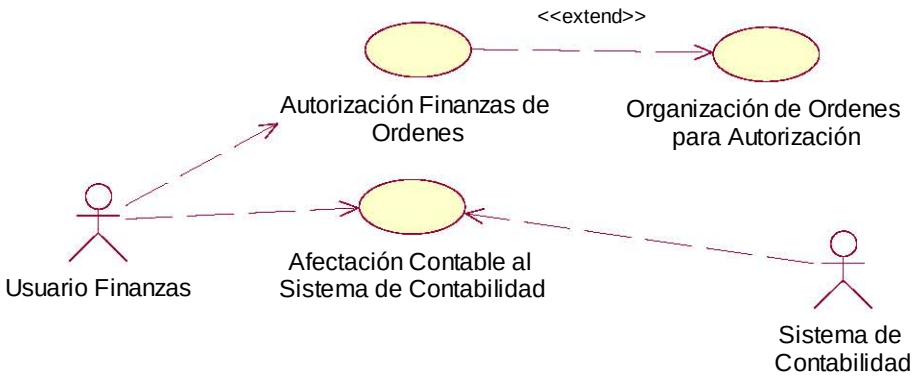
Contenido

Casos de Uso

Proyecto

R3.3.Exportar información Contable.

Diagrama



Descripción

Caso de Uso iniciado por el Usuario de Finanzas al momento de aplicar los movimientos de pagos por transferencia bancaria y por cheque. Permitirá el envío de información contable a SISE para aplicar dentro del módulo de contabilidad todos lo movimientos de las ordenes de pago.

Precondiciones

Las órdenes de pago deberán haber estado autorizadas por finanzas para que el movimiento en caso de que el tipo de pago sea vía cheque pueda hacerse la impresión del mismo y para la transferencia, se permita la generación del archivo a los sistemas bancarios en donde posteriormente se importarán las transferencias aplicadas.

FLUJO BÁSICO

Acción Actor	Acción Sistema
1. El usuario de finanzas realiza alguna de las siguientes acciones: a. Importa archivo de operaciones de transferencias aplicadas.	2. El sistema tomará la información de los movimientos contables a aplicar con la fecha en la que se esta generando y envía la información a SISE por medio de la interfase. Por cada movimiento contable generado, el sistema la

b. El usuario de caja imprime cheques dentro de su oficina. c. El usuario de caja realiza una reimpresión de cheque para una orden de pago. d. Los Usuarios de Finanzas o Contabilidad cancela órdenes de pago ya aplicadas o contabilizadas.	formatea y la envía a SISE.
	3. El SISE asigna un asiento y carátula que valida que el movimiento ha sido registrado correctamente.
	4. El sistema recibe la información de Asiento y No. de Carátula de SISE. Si el sistema no recibe la información de Asiento y Carátula se continúa con el flujo alterno A1.
	5. Actualiza información de la orden de pago con el Asiento y No. de Carátula.
	6. Sistema registra estatus de Contabilizada
7. Termina Caso de Uso.	

FLUJO ALTERNATIVO

A1-No se Recibe Asiento y Carátula	
	1. Si el sistema no recibe la información de Asiento y Carátula, en el caso de la solicitud de impresión o reimpresión de cheque, se enviará un mensaje al usuario y se cancelara el proceso de

	impresión para esa orden, solicitando al usuario que reintente nuevamente.
	2. Si el sistema no recibe la información de Asiento y Carátula al procesar el archivo de operaciones bancarias autorizadas, dicho proceso no continuará y se enviará mensaje de error, notificando al usuario que deberá reiniciar o cancelar el proceso.
	3. Si el sistema no recibe la información de Asiento y Carátula al cancelar una orden ya contabilizada, dicho proceso no continuará y se enviará mensaje de error, notificando al usuario que deberá cancelar o reintentar el proceso.

Requerimientos Especiales

Ninguno.

Post-Condiciones

Orden de Pago con estatus de Contabilizada

Orden de Pago actualizada con información de Asiento y Carátula

Casos de Uso

Autorización de Órdenes de Pago.

Versión: 1.4
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
02/May/2004	1.1	Revisión narrativa.		MSE
07/May/2004	1.2	Modificación Narrativa		CGC
09/May/2004	1.3	Modificación Especificación		CGC
27/05/2004	1.4	Modificación Especificación. Estatus Pendiente de Liberación	LG	CGC

Casos de Uso

Proyecto

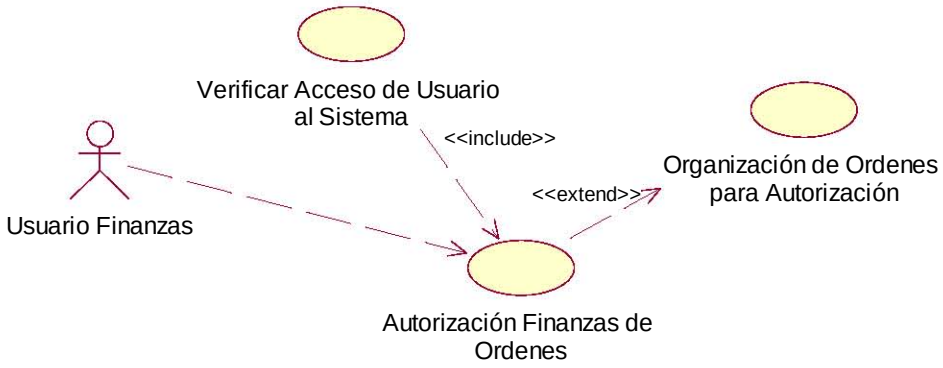
Contenido

Casos de Uso

Proyecto

R4.1.Autorización de Ordenes de Pago.

Diagrama



Descripción

Caso de Uso iniciado por el Usuario de Finanzas. Proporciona la capacidad de consultar, organizar y autorizar de manera centralizada todas las ordenes de pago registradas en el sistema SIF.

Precondiciones

Cuenta de Usuario válida y con perfiles de acceso al módulo de autorización de órdenes.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. El usuario de Finanzas ingresa a la Pantalla de Gestión Centralizada de órdenes de pago.	2. Carga criterios para búsqueda y clasificación de órdenes para su autorización.
3. El usuario de finanzas deberá especificar el tipo de documento que mostrará la consulta, estos pueden ser orden de pago o anticipo de gastos.	

4. Dentro de la sección "Datos Generales" el usuario de finanzas podrá indicar los criterios por los cuales desee filtrar la consulta de información. Para esto deberá indicar dentro de la columna "Selección", el filtro de cada concepto podrá seleccionarlo dentro de la columna "Valor". En caso de no especificar el filtro el sistema tomará todos los valores del criterio seleccionado.	
	5. Muestra también el campo de Estatus de Ordenes, el cual tendrá como valor predeterminado "Pendientes de Autorización para Pago". Las otras opciones son "Autorizadas" y "Canceladas".
6. Una vez seleccionados los criterios de consulta el usuario de finanzas deberá presionar el botón "Buscar".	7. El sistema mostrará el resultado de la consulta dentro de la sección "Autorizadas por Contabilidad".
	8. Sistema también presentará del resultado de la consulta las órdenes de pago que pueden ser consolidadas por Beneficiario y Forma de Pago en la sección "Consolidación".
9. Para autorizar órdenes el usuario de finanzas puede seleccionar de la sección "Autorizadas por Contabilidad", la(s) orden(es) y con clic derecho del Mouse se despliega un menú emergente donde puede	10. Las órdenes autorizadas se borraran de la pantalla principal y se acumularán en la sección "Autorizadas para Pago"y se borraran respectivamente de la sección de "Consolidación".

escoger la opción de autorizar.	
	11. Para aquellas órdenes donde la forma de pago es por cheque, valida la existencia de números de cheque y asigna el número de cheque.
12. El Usuario también puede optar por autorizar ordenes de la sección "Consolidación", en la que podrá seleccionar por beneficiario o forma de pago la autorización de todas sus ordenes y confirmar con clic en el botón "Consolidar"	13. Las órdenes autorizadas se borraran de la sección de Consolidación y se acumularán en la sección "Autorizadas para Pago"y se borraran respectivamente de la sección de "Autorizadas por Contabilidad".
	14. Consolidará aquellas órdenes de pago que se tengan que hacer a un mismo beneficiario con la asignación de un solo número de cheque por centro de costo y cuenta bancaria correspondiente.
	15. Si el sistema no pudiera asignar un número de cheque, porque el rango esta agotado, se continuará con el caso excepcional (E1).
	16. Actualiza estatus de órdenes confirmadas para pago.
17. Para rechazar órdenes el usuario de finanzas puede seleccionar la(s) orden(es) de Cualquiera de las dos secciones donde se presenta la lista de ordenes por autorizar: "Autorizadas por Contabilidad" y "Consolidación", y	18. Las órdenes rechazadas se borraran de la sección principal y se acumularán en la sección de Ordenes Canceladas.

con clic derecho del Mouse se despliega un menú emergente donde puede escoger la opción de rechazar.	
	19. Actualiza estatus de ordenes de pago canceladas.
20. Usuario sale de la pantalla.	
21. Para utilizar el módulo como consulta de las operaciones autorizadas, el usuario continua con flujo alterno A1.	

FLUJO ALTERNATIVO

A1- Consulta de Información	
Acción del Sistema	**Acción del Sistema**
1. En las opciones de Filtros de Búsqueda, el usuario selecciona una opción de consulta de estatus de ordenes distinta de “Pendientes de Autorización para Pago” y un periodo de fechas determinado” y con clic en el botón “Búsqueda” solicita realizar el proceso..	2. Sistema carga información en la pestaña o sección correspondiente para su consulta, conforme al estatus solicitado de búsqueda y periodo de fecha.
3. Uusario Termina caso	

FLUJO EXCEPCIONAL

E1. El sistema avisará al usuario de finanzas que no pudo realizar la asignación de cheques, y las órdenes pendientes conservarán su estatus de pendientes de confirmación para pago por el área de Finanzas.

Requerimientos Especiales

Ninguno.

Post-Condiciones

Ordenes autorizadas para su aplicación.

Casos de Uso

Generación de Archivos para liberación de cheques

Versión: 1.3
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
9/Mayo/2004	1.1	Modificación de Narrativa y Formato		CGC
20/05/2004	1.2	Modificación Flujo Básico.		CGC
27/05/2004	1.3	Modificación Especificación	JE	CGC

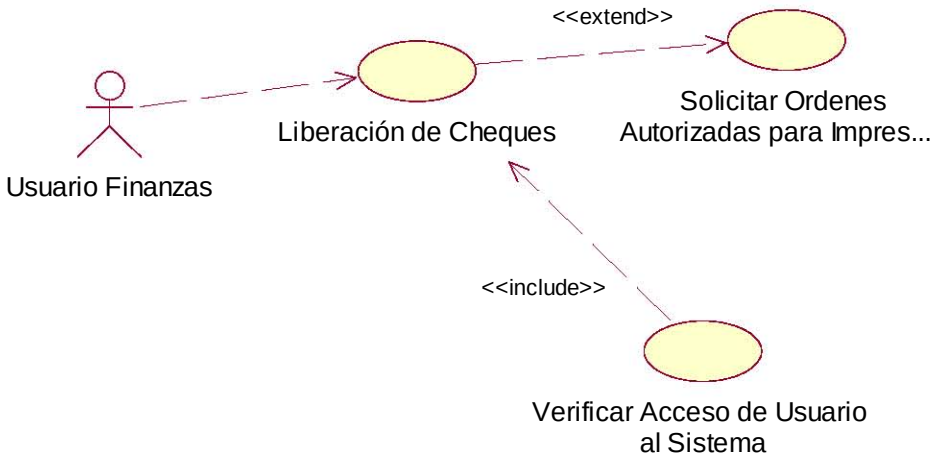
Casos de Uso

Proyecto

Contenido

R4.2.Generación de Archivos para liberación de cheques.

Diagrama



Descripción

Caso de Uso iniciado por el Usuario de Finanzas. Permitirá la generación de información para la liberación de cheques para órdenes de pago autorizadas para esta forma de pago.

Precondiciones

Cuenta de usuario valida y perfiles de acceso a la funcionalidad.

Que existan órdenes de pago autorizadas para impresión de cheque.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Usuario accesa al menú de generación de archivos para Bancos.	2. Presenta pantalla con opciones para generación de archivos bancarios.
3. Selecciona la opción de generar liberación de cheques y clic en el botón “Continuar”.	4. Sistema presenta la información de órdenes de cheques autorizados para su liberación y de Ordenes

	Reimpresas Pendientes de Liberación.
5. Usuario da clic en el botón de "Generar Archivo".	
	6. Sistema validará que exista la ruta en el servidor del SIF para guardar el archivo de liberación
	7. El sistema creará el archivo para liberación de cheques y asignará a las ordenes de pago de ese archivo un numero de control para asociar e identificar que el archivo se proceso. Al finalizar la operación mostrará al usuario la ruta, el nombre del archivo y el número de registros que fueron generados con éxito. Sistema asigna nombre de archivo en base a la siguiente nomenclatura: C – Archivo de Transferencia. DD – Dia de creación del archivo. MM - Mes de creación del archivo. YY - Año de creación del archivo. A-Z – Una letra como consecutivo en caso de que se generen varios archivos el mismo día. Sistema genera archivo y guarda en la ruta fija del servidor dentro del fólder correspondiente. X:\Nombre del Banco\Operaciones\Salidas.
8. Usuario accesa a su sistema de banca electrónica y con el uso del archivo generado procederá a realizar su operación de liberación de	

cheques.	
	9. En caso de que el archivo no se generé correctamente o se requiera volver a crearlo el usuario continua con el flujo alterno A1.
10. Usuario sale de la pantalla y termina caso de uso.	

FLUJO ALTERNATIVO

A1- Regeneración de Archivo	
Acción del Actor	**Acción del Sistema**
1. Usuario escoge opción de regeneración de archivos para Bancos.	2. Solicita segundo clave de acceso de validación.
3. Proporciona clave de acceso especial.	4. Válida clave de acceso y en caso de ser correcto, presenta listado con el número y nombre de archivos procesados hasta ese momento.
5. Usuario escoge el número de archivo a reprocesar y con clic en el botón de continuar solicita la regeneración del archivo.	6. Sistema regenerará archivo con el nombre correspondiente y en la ruta asignada.
	7. Sistema registra histórico de la regeneración, con los datos de la fecha, usuario, archivo y hora.

Requerimientos Especiales

Ninguno.

Post-Condiciones

Archivo de liberación generado.

Puntos de Extension

Sistema de Banca Electrónica.

Casos de Uso

Generación de Archivos para Transferencias Bancarias

Versión: 1.2
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
9/Mayo/2004	1.1	Modificación Formato y Narrativa		CGC
27/05/2004	1.2	Modificación Especificación	JE	CGC

Casos de Uso

Proyecto

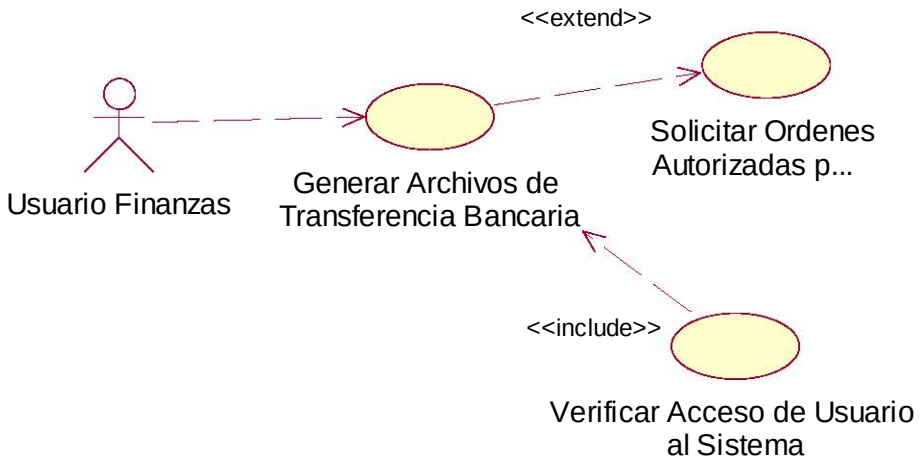
Contenido

Casos de Uso

Proyecto

R4.3.Generación de Archivos para Transferencias Bancarias.

Diagrama



Descripción

Permitirá la generación a petición del usuario de Finanzas del archivo para carga de órdenes de pago autorizadas vía transferencia en los sistemas bancarios.

Precondiciones

Cuanta de Usuario Validada por el sistema. Que el usuario de Finanzas tenga el perfil para acceder a la funcionalidad.

Que existan órdenes autorizadas para pago por transferencia bancaria.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. El usuario de finanzas ingresa al menú de Generación de Archivos para Bancos.	2. Presenta pantalla de opciones par generación de archivos bancarios.
3. Selecciona la opción de generación de archivos por transferencia	4. Sistema presenta las órdenes autorizadas para transferencias.

bancaria y clic en el botón de "Continuar".	
5. Usuario Selecciona las ordenes de pago, el tipo de operación, el Banco y cuenta pagadora.	
6. Clic en el botón de "Generar Archivo".	
	Sistema validará que exista la ruta en el servidor del SIF para guardar el archivo de liberación
	Sistema asigna nombre de archivo en base a la siguiente nomenclatura: T – Archivo de Transferencia. DD – Dia de creación del archivo. MM - Mes de creación del archivo. YY - Año de creación del archivo. A-Z – Una letra como consecutivo en caso de que se generen varios archivos el mismo día. Sistema genera archivo y guarda en la ruta fija del servidor dentro del fólder correspondiente. X:\Banamex\Operaciones\Salidas. Una vez terminada la operación el sistema mostrará el nombre y ruta asignada para guardar el archivo y el número de registros que fueron generados dentro del archivo.
Usuario deberá acceder a su sistema de banca electrónica y con el uso del archivo generado procederá a realizar su operación.	
	7. En caso de que el archivo no se generé correctamente o se requiera

	volver a crearlo el usuario continúa con el flujo alterno A1.
Usuario sale de la pantalla.	

FLUJO ALTERNATIVO

A1- Regeneración de Archivo	
Acción del Actor	**Acción del Sistema**
1. Usuario escoge opción de regeneración de archivos para Bancos.	2. Solicita segunda clave de acceso.
3. Proporciona clave de acceso especial	4. Valida clave de acceso y en caso de ser correcto, presenta listado con el número y nombre de archivos procesados hasta ese momento.
5. Usuario escoge el número de archivo a reprocesar y con clic en el botón de continuar solicita la regeneración del archivo.	6. Sistema regenerará archivo con el nombre correspondiente y en la ruta asignada.

Requerimientos Especiales

Ninguno.

Post-Condiciones

Archivos generados.

Casos de Uso

Impresión de Cheques

Versión: 1.2
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
12/Mayo/2004	1.1.	Modificación Formato y Narrativa		CGC
27/May/2004	1.2	Modificación Narrativa.	LGL	CGC

Casos de Uso

Proyecto

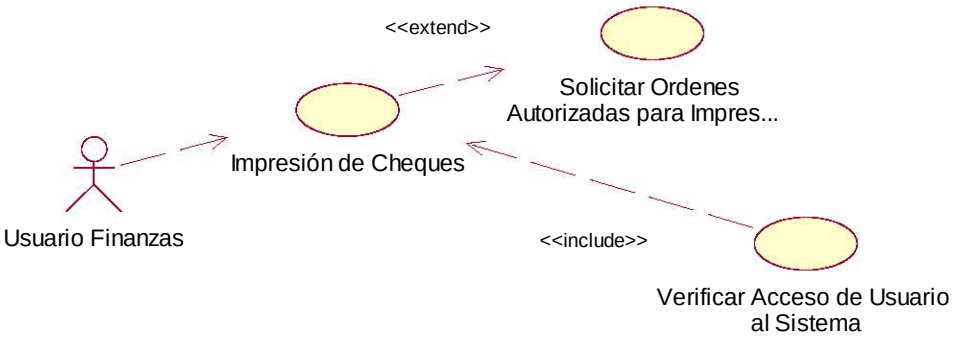
Contenido

Casos de Uso

Proyecto

R4.4.Impresión de Cheques.

Diagrama



Descripción

Caso de Uso inicia por el usuario de finanzas. Permitirá la impresión de cheques autorizados para este tipo de pago.

Precondiciones

Cuenta de Usuario validada y con perfiles de acceso a esta funcionalidad.

Que existan órdenes autorizadas para pago por cheque.

FLUJO BASICO

Acción del Actor	Acción del Sistema.
1. El usuario de impresión accesa a la pantalla de Impresión de Cheques.	Consulta cheques por imprimir y los muestra al usuario. Las órdenes a imprimir serán aquellas que se encuentran ya con el estatus de Liberadas por Finanzas.
2. Proporciona número de orden especifica y da clic en el botón de "Buscar".	3. Sistema presenta la información solicitada. Si el usuario no especifica el número de orden, el Sistema traerá todos los cheques pendientes por imprimir.

4. El usuario selecciona el Cheque a imprimir con clic del Mouse dentro de la sección “Cheques por imprimir” o selecciona opción de impresión de todos los cheques de manera continua o dependiendo del usuario solo se podrá imprimir los cheques uno a uno.	Sistema elimina cheques de la pantalla. Si la opción de impresión múltiple esta activa, solicita clave de autorización especial para proceder de manera continua o no.
5. Da clic en el botón de “Imprimir” para confirmar.	
	6. El sistema solicita al usuario que coloque la forma con el número del cheque correspondiente.
	7. El Sistema genera la información contable a transferir a SISE, en el momento que se reciba el Asiento y carátula, se procederá a la impresión del cheque. Si no se recibe la información de Asiento y Carátula, el sistema notificará al usuario que no es posible realizar la impresión de cheque y que si desea reintentar el proceso de impresión o volver a solicitarlo más tarde.
8. Si procede a la impresión y el usuario detecta que la impresión no fue satisfactoria procederá con el caso alterno A1. Reimpresión de Cheque o solicitar la cancelación de la orden a	

Finanzas.	
	9. Imprime Cheque. Si no se recibe ningún problema con la impresión, se actualiza Estatus de Impreso en la Orden
	10. Fin del Caso de Uso

FLUJO ALTERNATIVO

A1 – Reimpresión de Cheques	
Acción del Actor	**Acción del Sistema**
1. El usuario selecciona la reimpresión de cheques.	2. Sistema solicita clave de autorización para ingresar al proceso de reimpresión.
	3. Muestra una pantalla limpia con los grids de resultado de consulta y de confirmación.
4. Usuario especifica un número de orden ya impresa o si lo omite, el sistema asumirá que la consulta debe regresar todas las órdenes impresas por el usuario. Da clic en el botón de “Buscar”.	5. Muestra en el primer grid el resultado de la consulta.
6. Selecciona la(s) órdenes a reimprimir.	
7. Da clic en el botón de “Imprimir” para confirmar.	8. Sistema asigna nuevo número de cheque conforme al rango de cheques asignado para la oficina. 9. Registra histórico de reimpresión de cheque. Se generará información contable

	para recibir el no. de Asiento y Carátula de SISE. Si recibe información de SISE procede a la reimpresión. Solicitará al usuario la colocación de la forma con el no. de cheque reasignado para la reimpresión y solicita confirmación.
	10. Reimprime Cheque. Si no se recibe ningún problema con la impresión, se actualiza Estatus de Reimpreso Pendiente de Liberación en la Orden.
	11. La reimpresión de la orden de pago implicará generar el registro dentro del archivo para cancelar la numeración de cheques y el nuevo cheque se incluirá dentro de la información del archivo para liberación de cheques.

Requerimientos Especiales

Ninguna.

Post-Condiciones

Archivo de impresión enviado a impresora.

Cheque impreso

Puntos de Extension

Ninguno

Casos de Uso

Conciliación Bancaria.

Versión: 1.3
Revisión 0
Elaboró: Carlos A. Gutiérrez
Autorizó:
Validó:
Fecha: 27 de Mayo de 2004.

Casos de Uso

Proyecto

Control de Versiones

Fecha	Versión	Descripción de Cambios	Solicitud de Cambio	Autor
	1.0	Creación del Documento		
13/05/2004	1.1.	Modificación Formato y Narrativa	LGL	CGC
21/05/2004	1.2	Modificación Post Condición	AMP	CGC
27/05/2004	1.3	Modificación Narrativa	JE	CGC

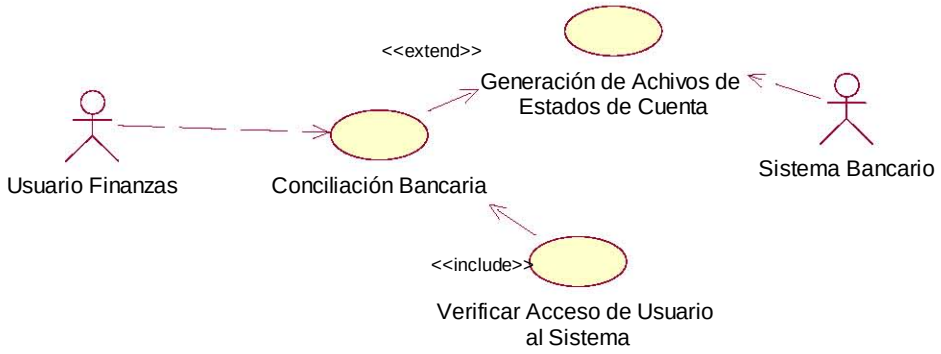
Casos de Uso

Proyecto

Contenido

R4.5. Importación de Información de Estados de Cuenta para Conciliación Bancaria.

Diagrama



Descripción

Usuario Finanzas inicia caso de uso. Proporciona la capacidad de importar el archivo de estados de cuenta de los sistemas bancarios para realizar la conciliación bancaria.

Precondiciones

Cuenta de usuario validada con perfiles de acceso a la funcionalidad de Conciliación Bancaria.

Que existan archivos de estados de cuentas bancarias.

FLUJO BÁSICO

Acción del Actor	Acción del Sistema
1. Ingresa a la pantalla para cargar de Conciliación Bancaria.	2. Presenta pantalla con las opciones de especificar nombre del banco.
	3. Despliega todos los archivos de conciliación pendientes de procesar que se encuentran en el

	fólder con el nombre del banco.
4. Usuario selecciona de la lista el o los archivos a procesar.	
5. Con clic en el botón de “Cargar Archivo” solicita el procesamiento del archivo seleccionado.	6. Procesa archivo presentando avance de procesamiento y al finalizar presenta en una ventana resultado de nombre de archivo, periodo y no. de registros procesados.
7. Usuario da clic en el botón de “Realizar Conciliación” para solicitar la información de conciliación.	8. Presenta resultado del proceso. En una sección se muestra el resultado de la conciliación sobre ordenes mostrando en color Azul los registros conciliados y en Rojo los registros no conciliados.
	9. Si durante el proceso del archivo de Conciliación Bancaria, existen movimientos bancarios que no corresponden a las operaciones propias de órdenes de pago se mostraran en otra sección para su consulta.
10. Fin de Caso de Uso	

FLUJO ALTERNATIVO

Ninguno.

Requerimientos Especiales

Ninguno.

Post-Condiciones

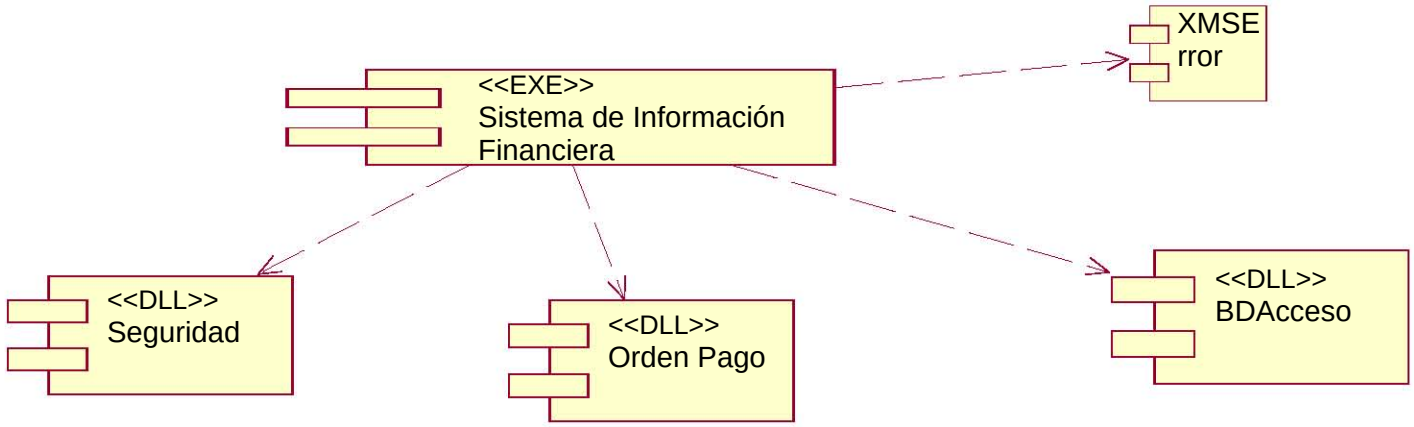
Resultado de procesamiento de Archivo

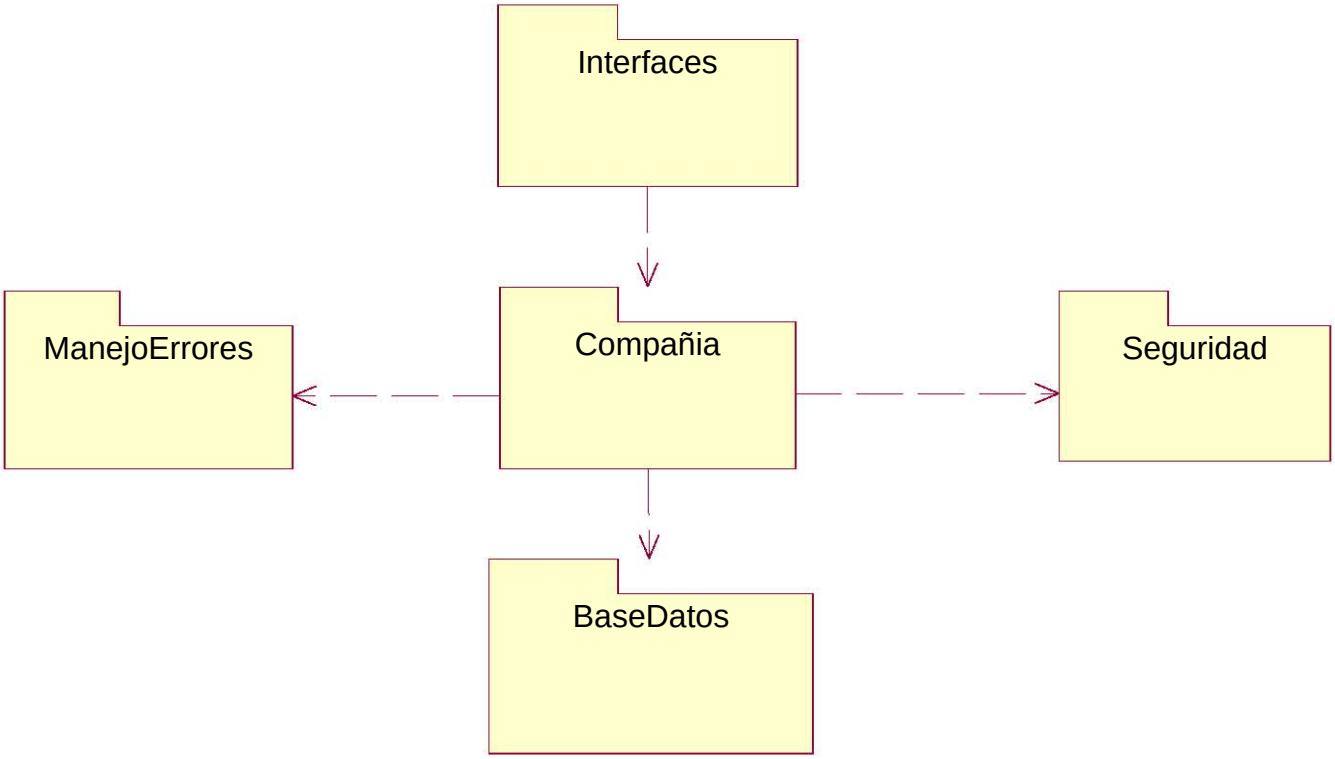
Casos de Uso

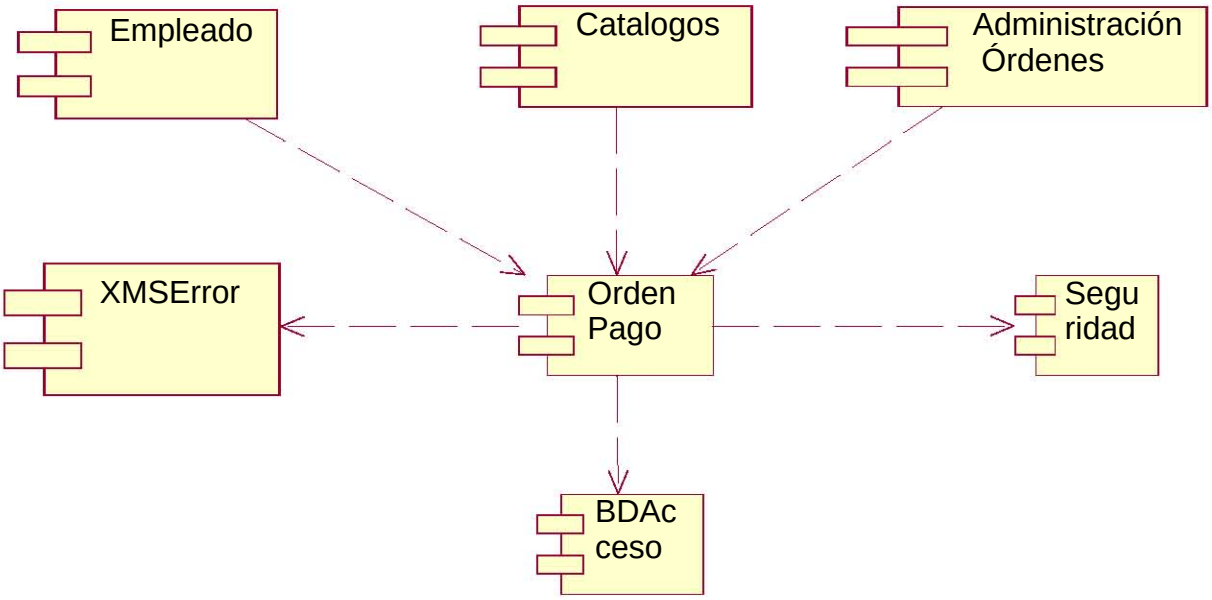
Proyecto

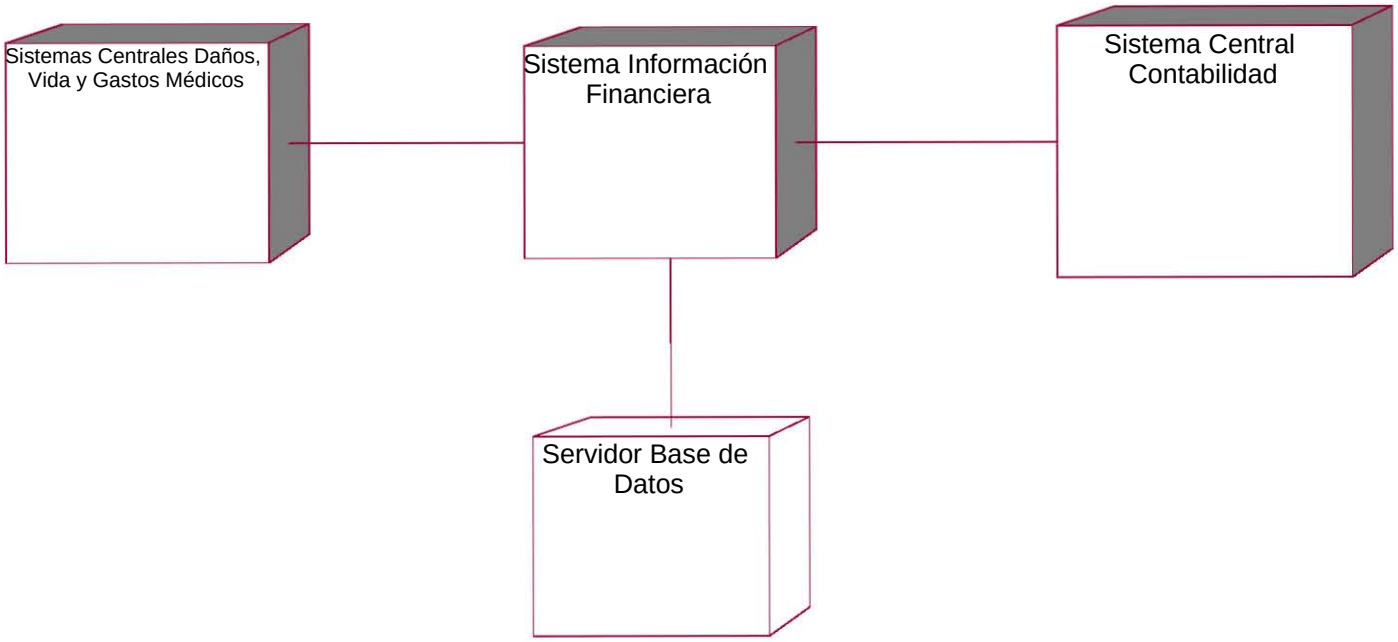
Puntos de Extension

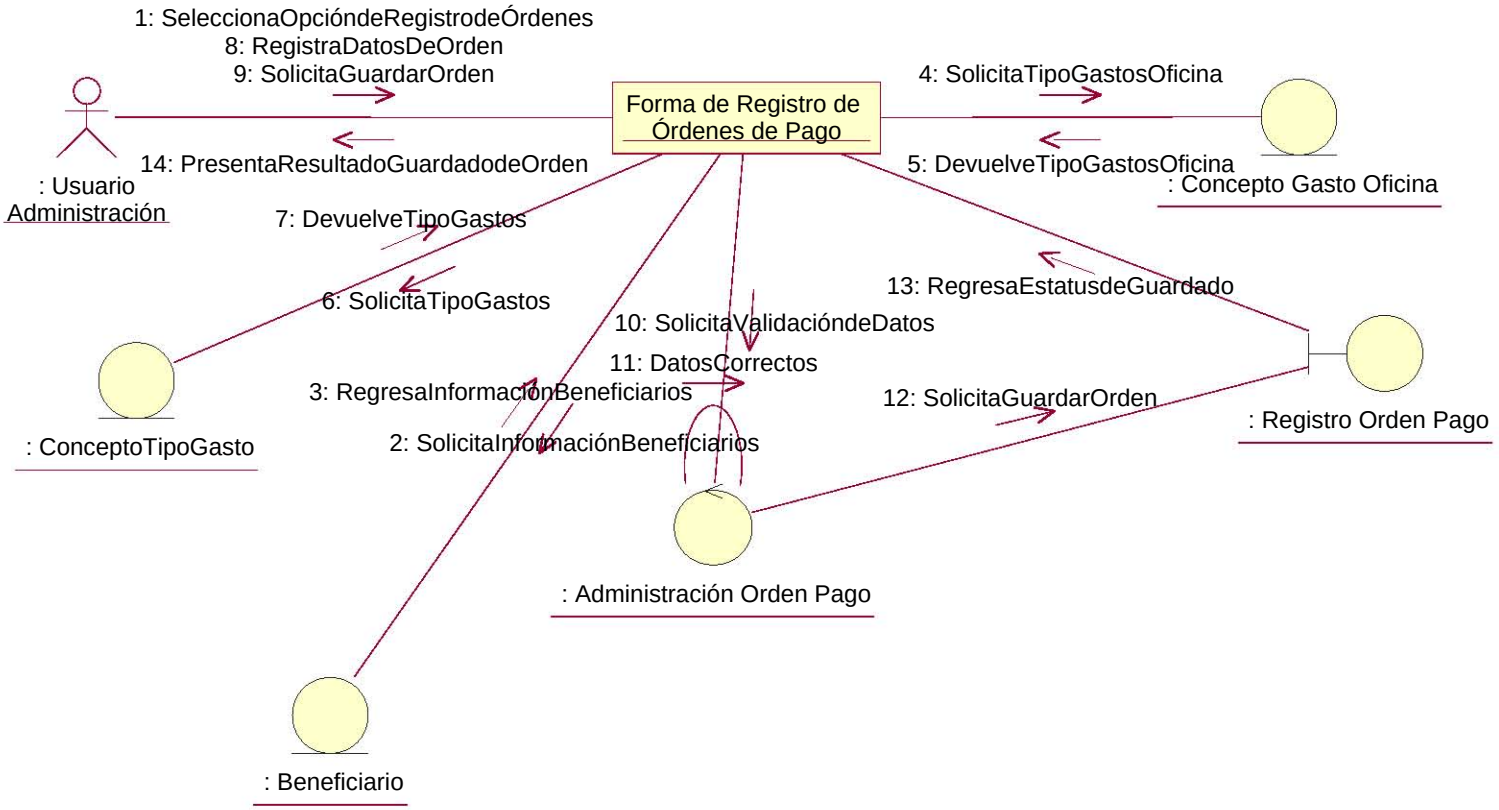
Ninguno

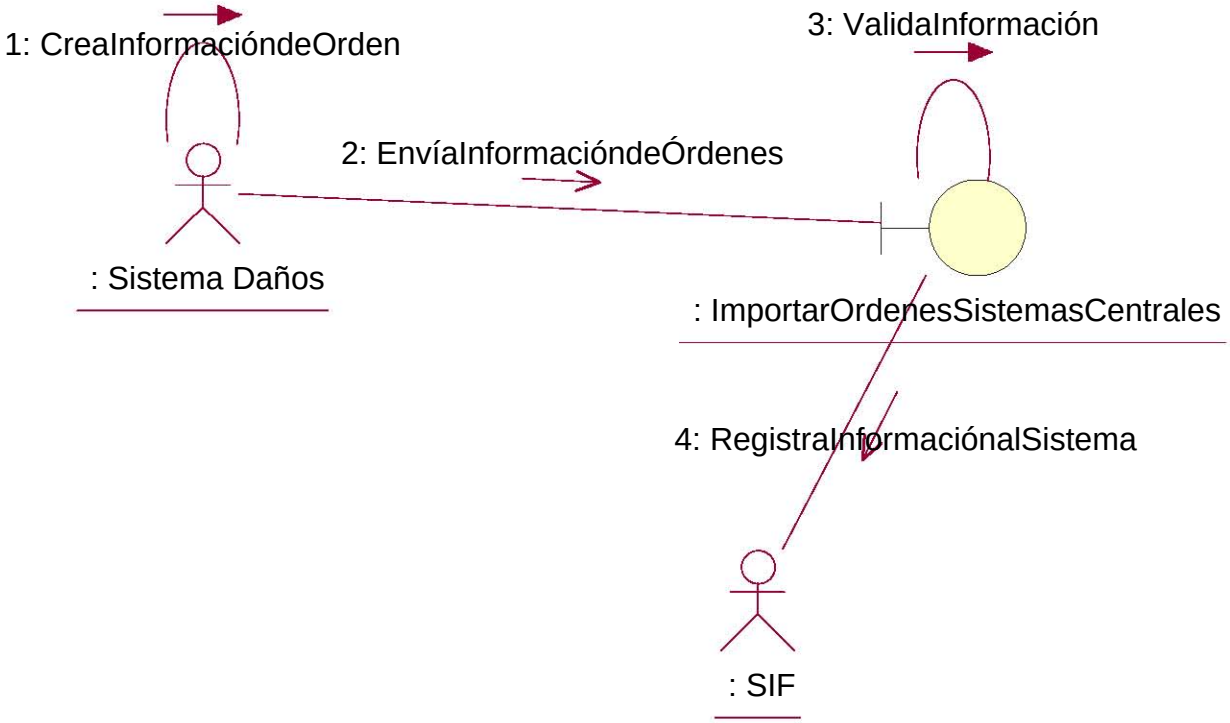

XMSE
rror
<<EXE>>
Sistema de Información
Financiera
<<DLL>>
Seguridad
<<DLL>>
Orden Pago
<<DLL>>
BDAcceso

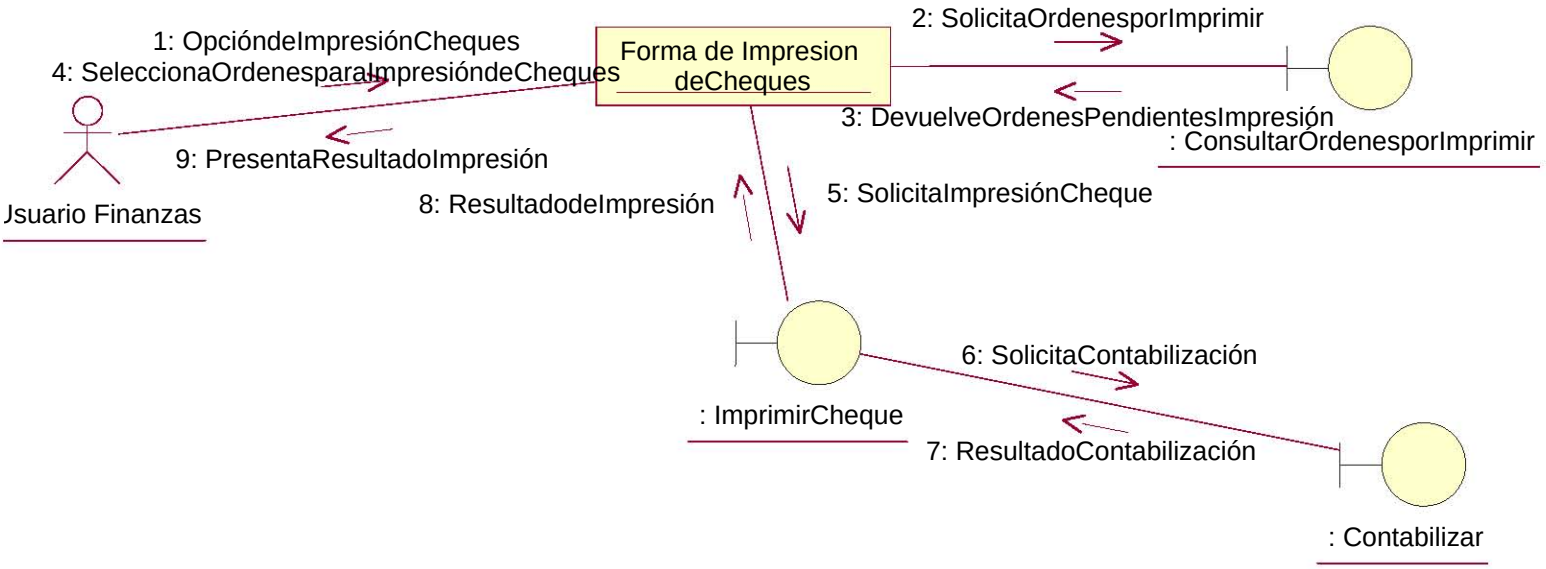

Interfaces
ManejoErrores
Compañia
Seguridad
BaseDatos

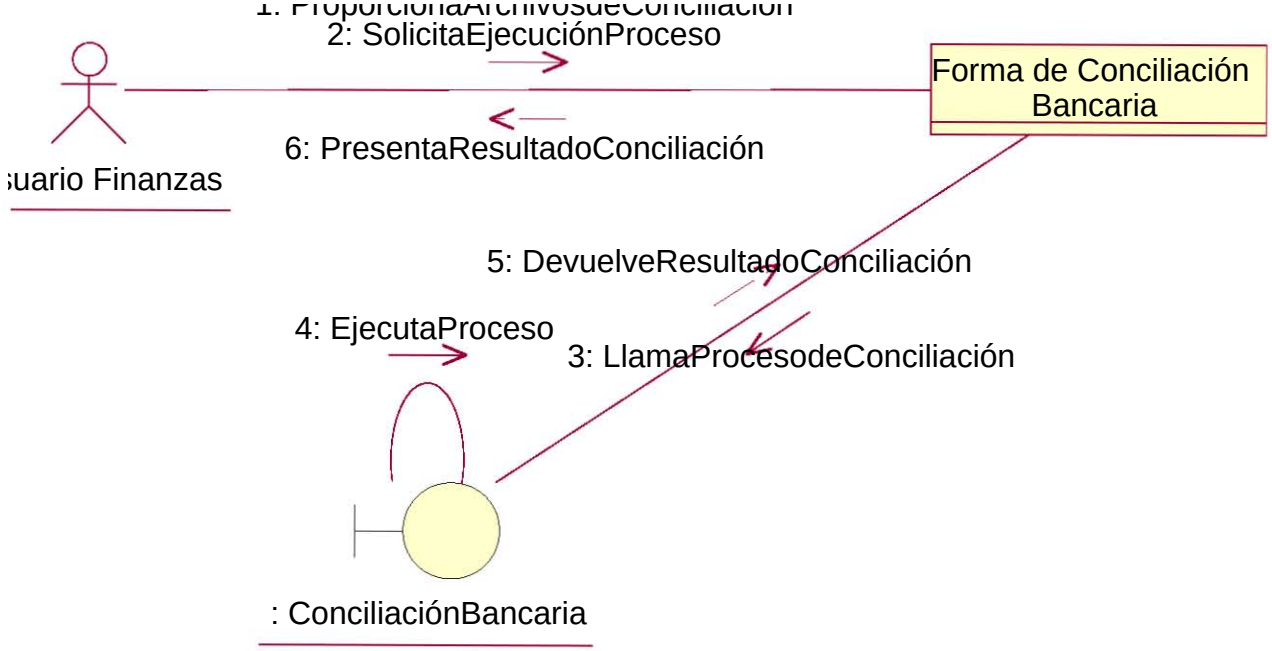

Empleado
Catalogos
Administración
Órdenes
XMSError
Orden
Pago
Segu
ridad
BDAc
ceso

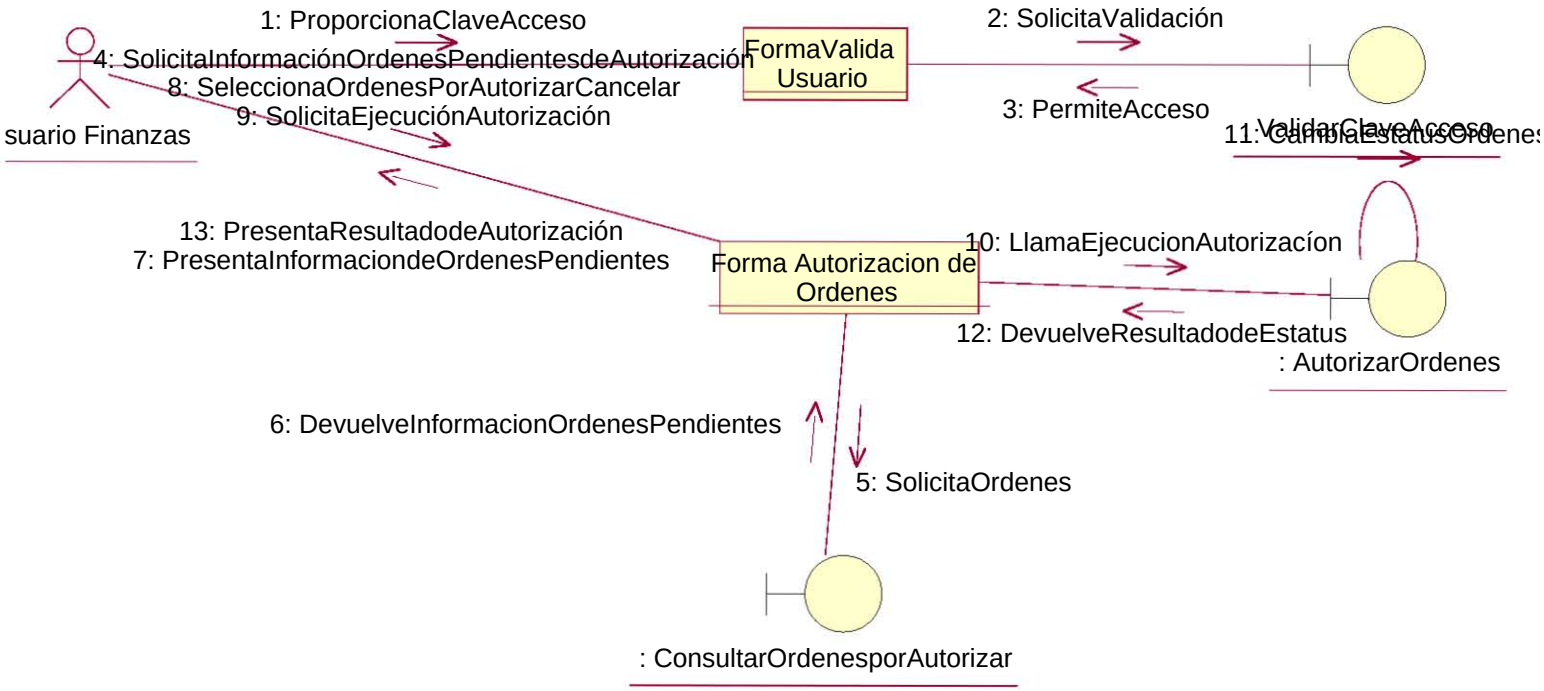

Sistemas Centrales Daños, Vida y Gastos Médicos
Sistema Información Financiera
Sistema Central Contabilidad
Servidor Base de Datos

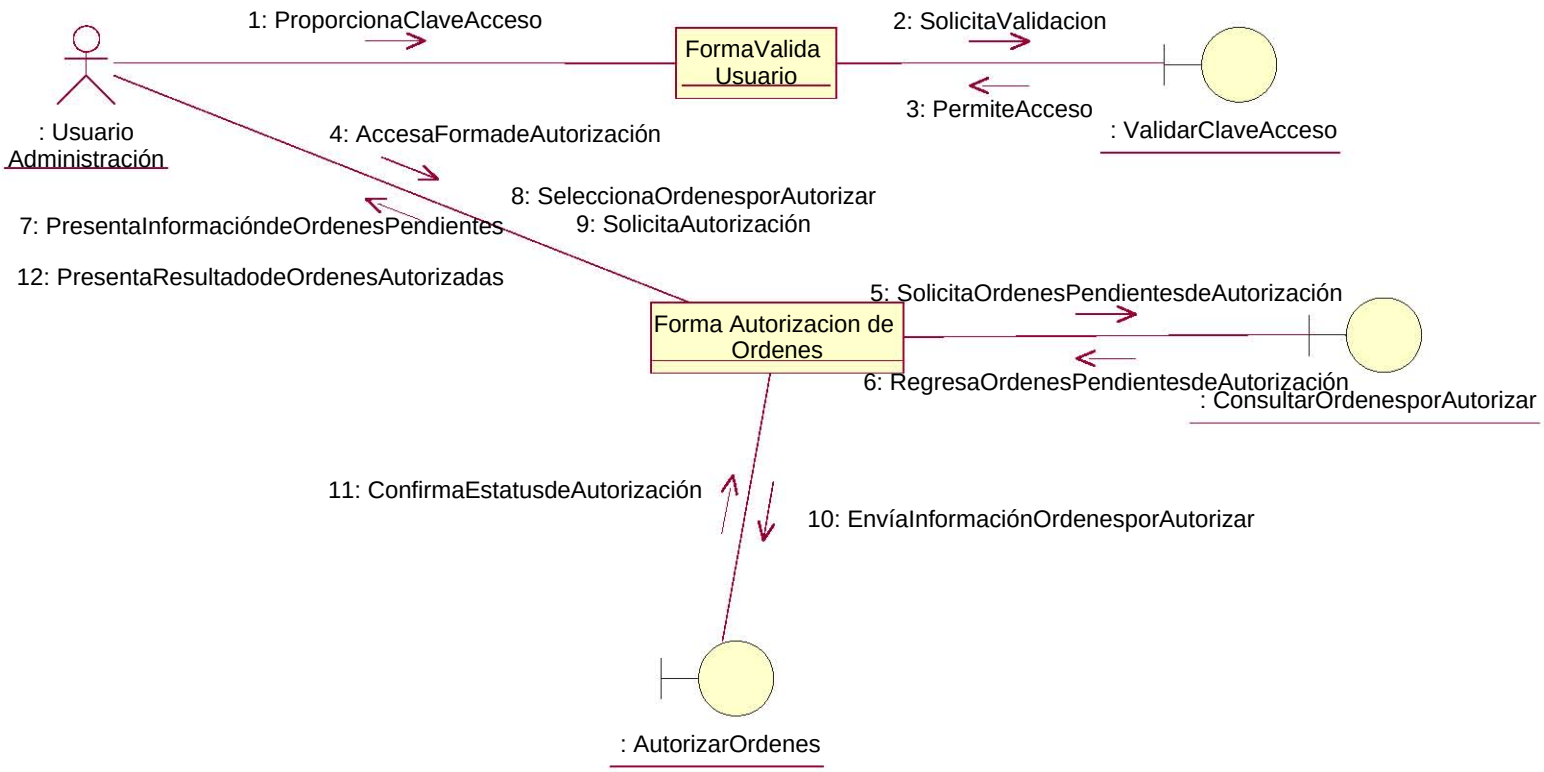

1: SeleccionaOpciónde RegistrodeÓrdenes
8: RegistraDatosDeOrden
9: SolicitaGuardarOrden
4: SolicitaTipoGastosOficina
Forma de Registro de Órdenes de Pago
14: PresentaResultadoGuardadodeOrden
5: DevuelveTipoGastosOficina
: Usuario Administración
: Concepto Gasto Oficina
7: DevuelveTipoGastos
6: SolicitaTipoGastos
13: RegresaEstatusdeGuardado
10: SolicitaValidaciónde Datos
11: DatosCorrectos
3: RegresaInformaciónBeneficiarios
12: SolicitaGuardarOrden
: Registro Orden Pago
: ConceptoTipoGasto
2: SolicitaInformaciónBeneficiarios
: Administración Orden Pago
: Beneficiario

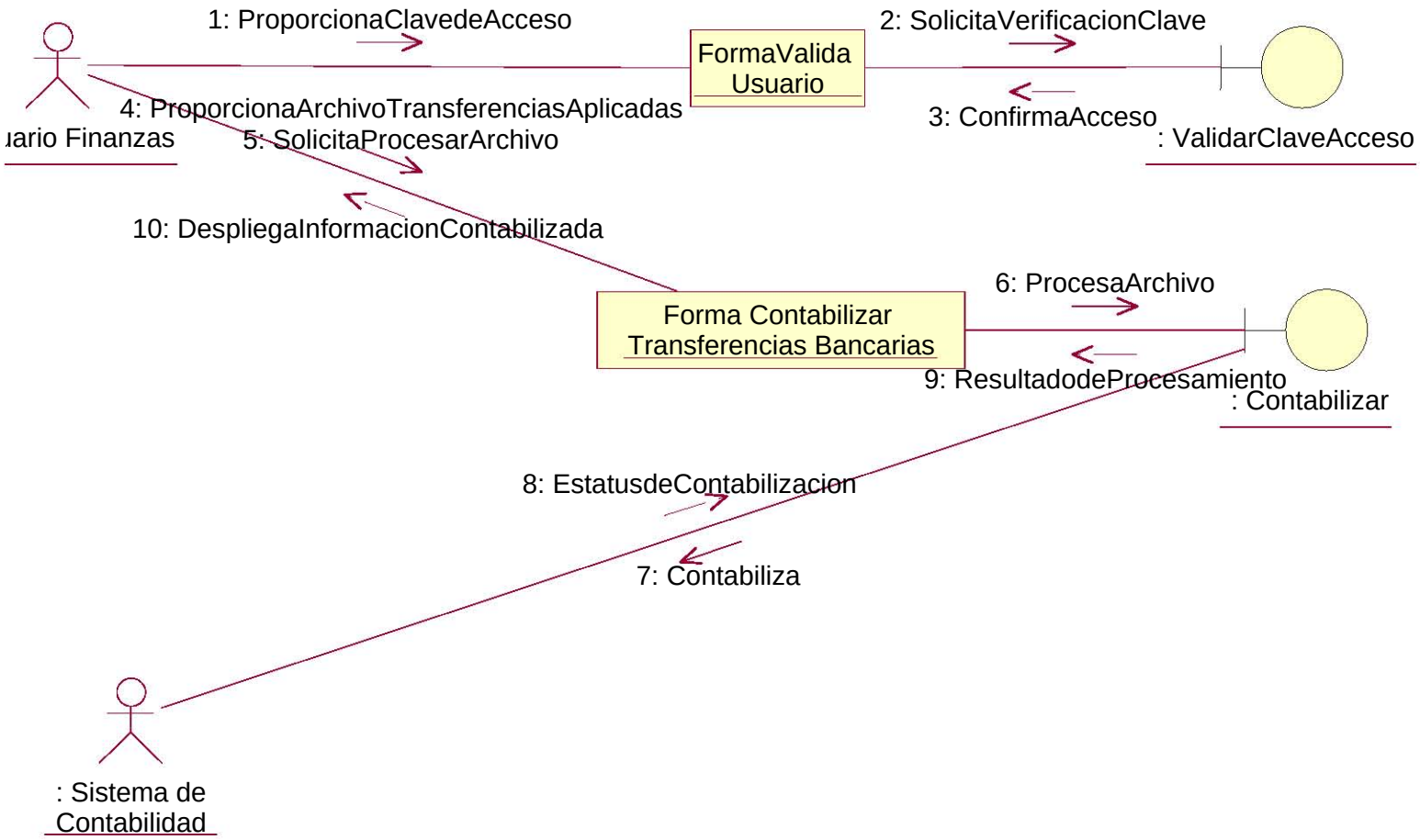

1: CreaInformacióndeOrden
3: ValidaInformación
2: EnvíaInformacióndeÓrdenes
: Sistema Daños
: ImportarOrdenesSistemasCentrales
4: RegistraInformaciónalSistema
: SIF

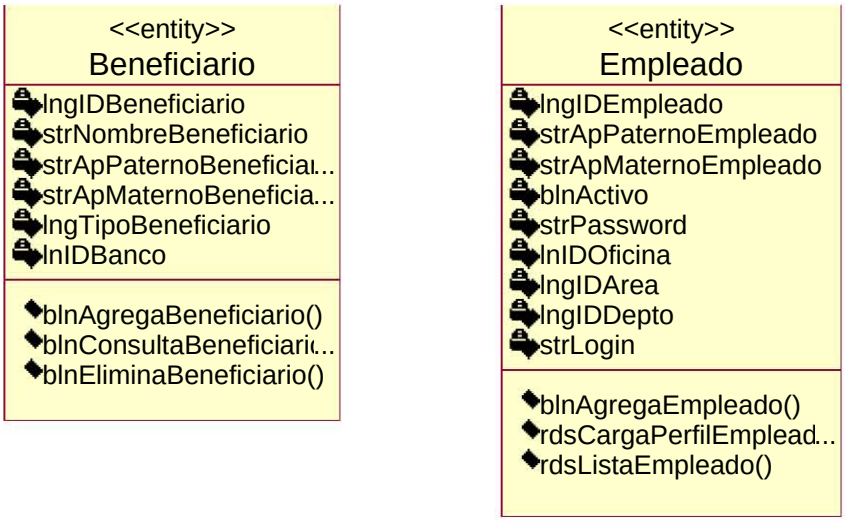

1: OpciónIdeImpresiónCheques
4: SeleccionaOrdenesparaImpresióndeCheques
Forma de Impresion deCheques
2: SolicitaOrdenesporImprimir
3: DevuelveOrdenesPendientesImpresión
: ConsultarÓrdenesporImprimir
9: PresentaResultadoImpresión
Jsuario Finanzas
8: ResultadodeImpresión
5: SolicitaImpresiónCheque
6: SolicitaContabilización
: ImprimirCheque
7: ResultadoContabilización
: Contabilizar

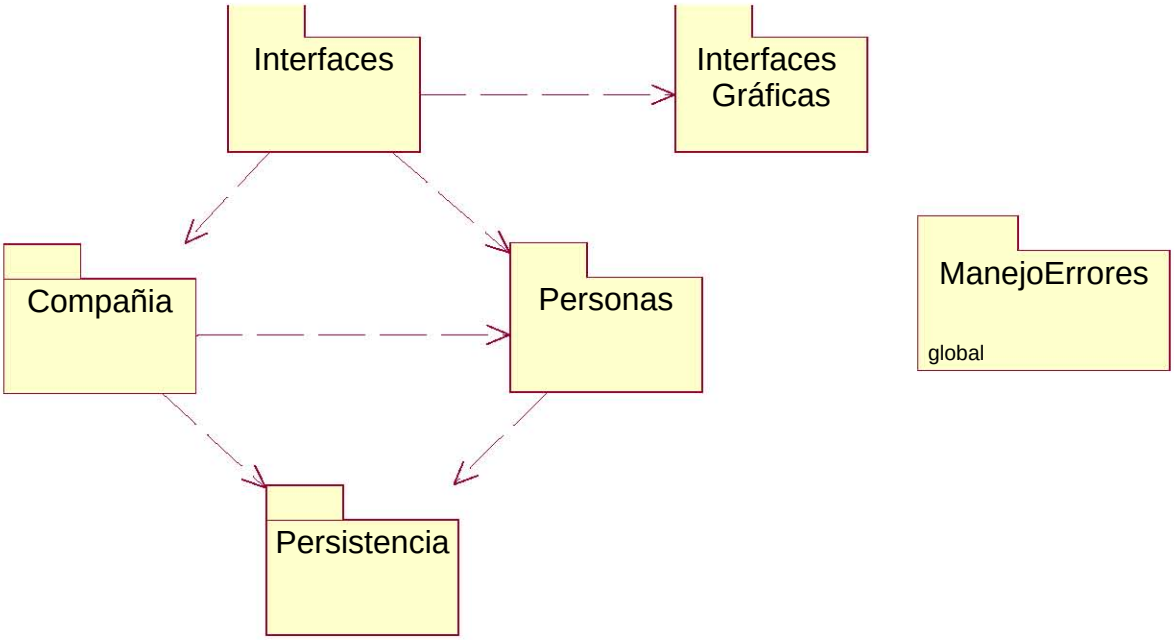

1: ProporcionaArchivosdeConciliación
2: SolicitaEjecuciónProceso
Forma de Conciliación Bancaria
6: PresentaResultadoConciliación
suario Finanzas
5: DevuelveResultadoConciliación
4: EjecutaProceso
3: LlamaProcesodeConciliación
: ConciliaciónBancaria

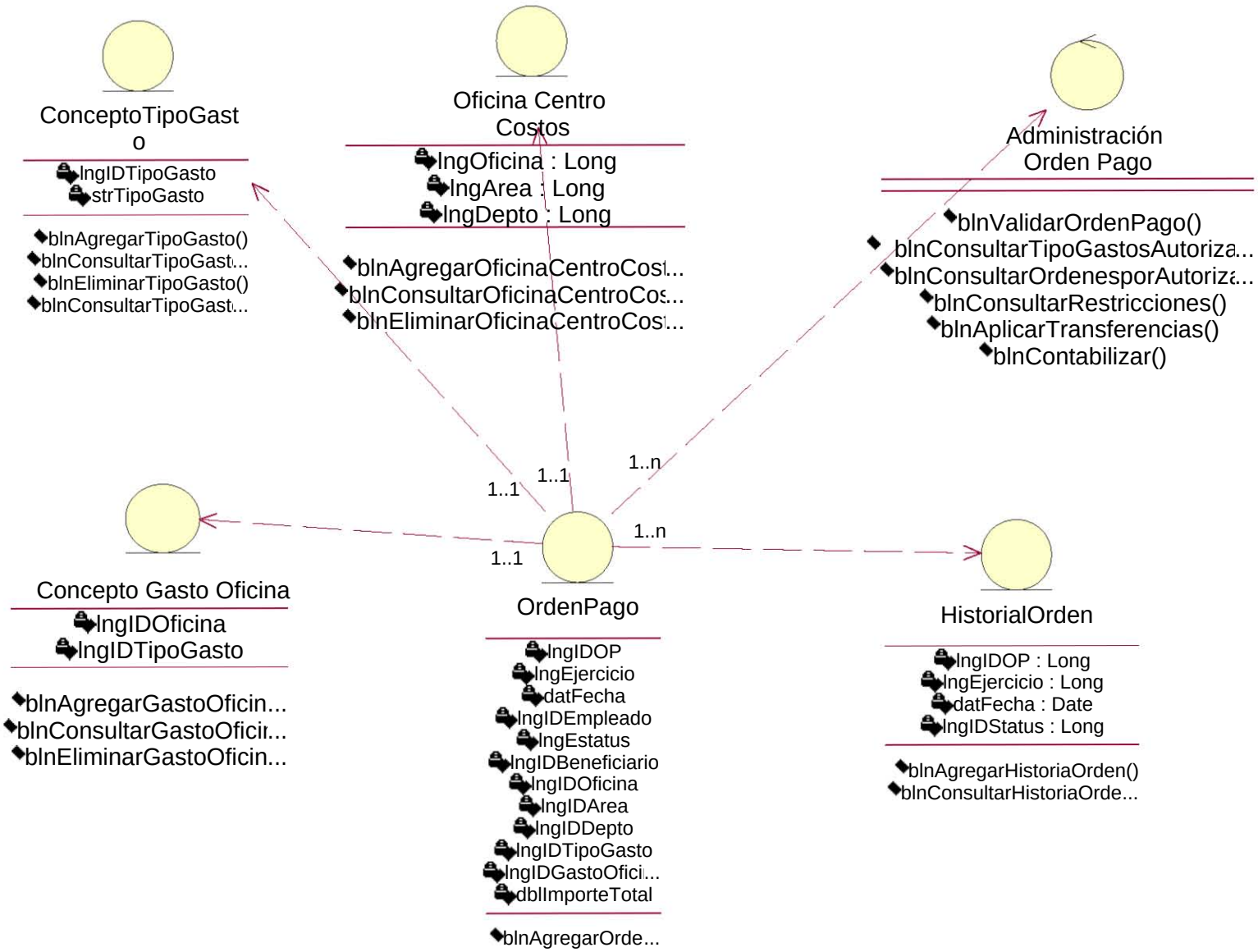

1: ProporcionaClaveAcceso
2: SolicitaValidación
FormaValida Usuario
4: SolicitaInformaciónOrdenesPendientesdeAutorización
8: SeleccionaOrdenesPorAutorizarCancelar
9: SolicitaEjecuciónAutorización
3: PermiteAcceso
ValidarClaveAcceso
suario Finanzas
11: CambiaEstatusOrdenes
13: PresentaResultadodeAutorización
7: PresentaInformaciondeOrdenesPendientes
Forma Autorizacion de Ordenes
10: LlamaEjecucionAutorizacíon
12: DevuelveResultadodeEstatus
: AutorizarOrdenes
6: DevuelveInformacionOrdenesPendientes
5: SolicitaOrdenes
: ConsultarOrdenesporAutorizar

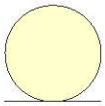

1: ProporcionaClaveAcceso
FormaValida Usuario
2: SolicitaValidacion
3: PermiteAcceso
: ValidarClaveAcceso
: Usuario Administración
4: AccesaFormadeAutorización
8: SeleccionaOrdenesporAutorizar
9: SolicitaAutorización
7: PresentaInformaciónde OrdenesPendientes
12: PresentaResultadodeOrdenesAutorizadas
5: SolicitaOrdenesPendientesdeAutorización
Forma Autorizacion de Ordenes
6: RegresaOrdenesPendientesdeAutorización
: ConsultarOrdenesporAutorizar
11: ConfirmaEstatusdeAutorización
10: EnvíaInformaciónOrdenesporAutorizar
: AutorizarOrdenes


1: ProporcionaClavedeAcceso
2: SolicitaVerificacionClave
FormaValida Usuario
3: ConfirmaAcceso
: ValidarClaveAcceso
uario Finanzas
4: ProporcionaArchivoTransferenciasAplicadas
5: SolicitaProcesarArchivo
10: DespliegaInformacionContabilizada
6: ProcesaArchivo
Forma Contabilizar Transferencias Bancarias
9: ResultadodeProcesamiento
: Contabilizar
8: EstatusdeContabilizacion
7: Contabiliza
: Sistema de Contabilidad


<<entity>>
Beneficiario
lngIDBeneficiario
strNombreBeneficiario
strApPaternoBeneficia...
strApMaternoBeneficia...
lngTipoBeneficiario
lnIDBanco
blnAgregaBeneficiario()
blnConsultaBeneficiari...
blnEliminaBeneficiario()
<<entity>>
Empleado
lngIDEmpleado
strApPaternoEmpleado
strApMaternoEmpleado
blnActivo
strPassword
lnIDOficina
lngIDArea
lngIDDepto
strLogin
blnAgregaEmpleado()
rdsCargaPerfilEmplead...
rdsListaEmpleado()


Interfaces
Interfaces Gráficas
Compañia
Personas
ManejoErrores
global
Persistencia

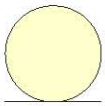

ConceptoTipoGasto
lngIDTipoGasto
strTipoGasto
blnAgregarTipoGasto()
blnConsultarTipoGast...
blnEliminarTipoGasto()
blnConsultarTipoGast...
Oficina Centro Costos
lngOficina : Long
lngArea : Long
lngDepto : Long
blnAgregarOficinaCentroCost...
blnConsultarOficinaCentroCos...
blnEliminarOficinaCentroCost...
Administración Orden Pago
blnValidarOrdenPago()
blnConsultarTipoGastosAutoriza...
blnConsultarOrdenesporAutoriza...
blnConsultarRestricciones()
blnAplicarTransferencias()
blnContabilizar()
1..1
1..1
1..n
1..n
1..1
Concepto Gasto Oficina
lngIDOficina
lngIDTipoGasto
blnAgregarGastoOficin...
blnConsultarGastoOficin...
blnEliminarGastoOficin...
OrdenPago
lngIDOP
lngEjercicio
datFecha
lngIDEmpleado
lngEstatus
lngIDBeneficiario
lngIDOficina
lngIDArea
lngIDDepto
lngIDTipoGasto
lngIDGastoOfici...
dblImporteTotal
blnAgregarOrde...
HistorialOrden
lngIDOP : Long
lngEjercicio : Long
datFecha : Date
lngIDStatus : Long
blnAgregarHistoriaOrden()
blnConsultarHistoriaOrde...



Beneficiario

lngIDBeneficiario
strNombreBeneficiario
strApPaternoBeneficiario
strApMaternoBeneficiario
lngTipoBeneficiario
lnIDBanco

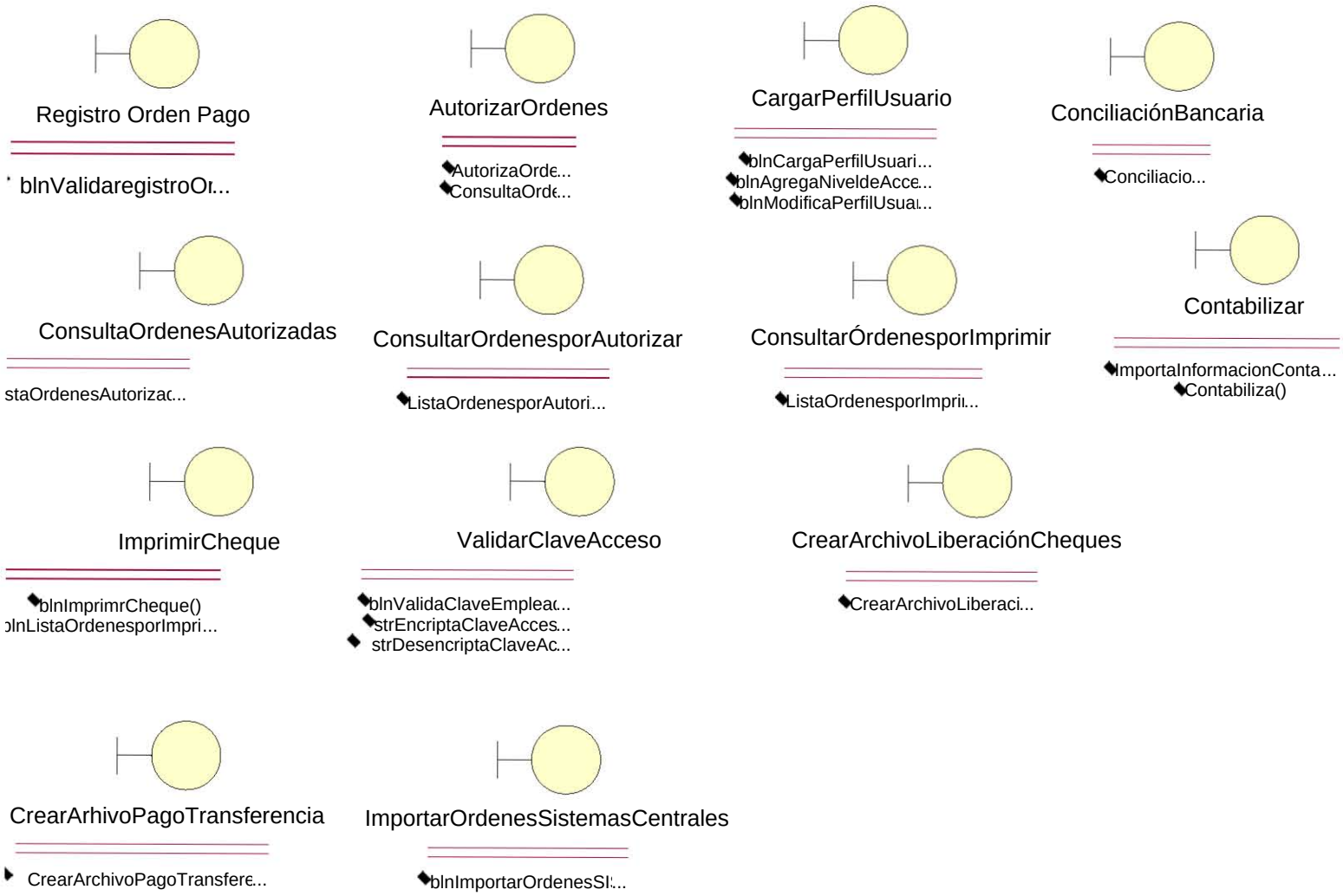
blnAgregaBeneficiario()
blnConsultaBeneficiario()
blnEliminaBeneficiario()

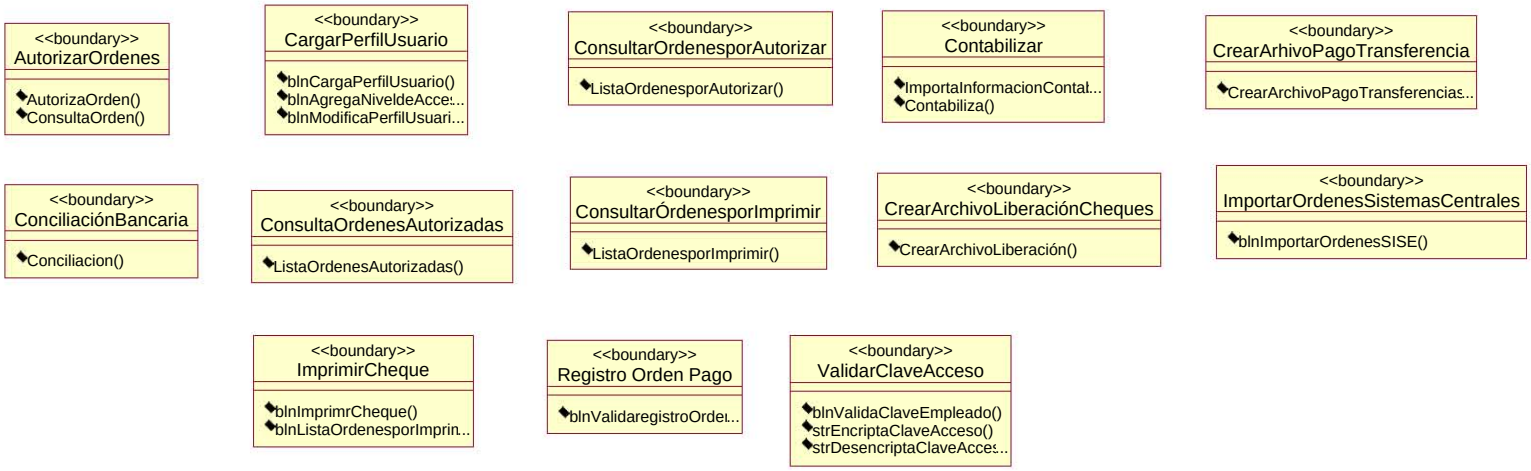


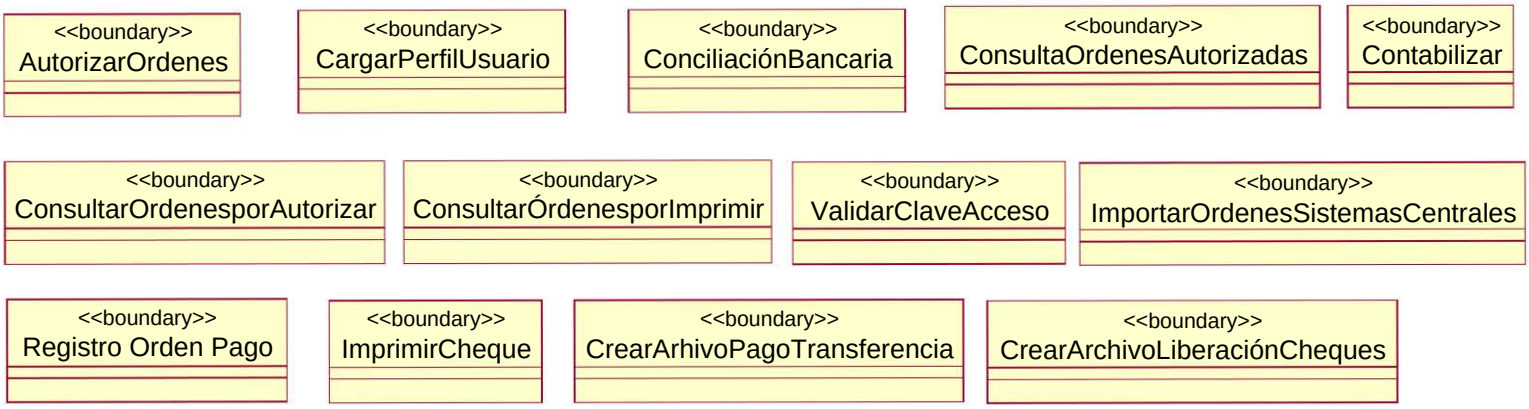
Empleado

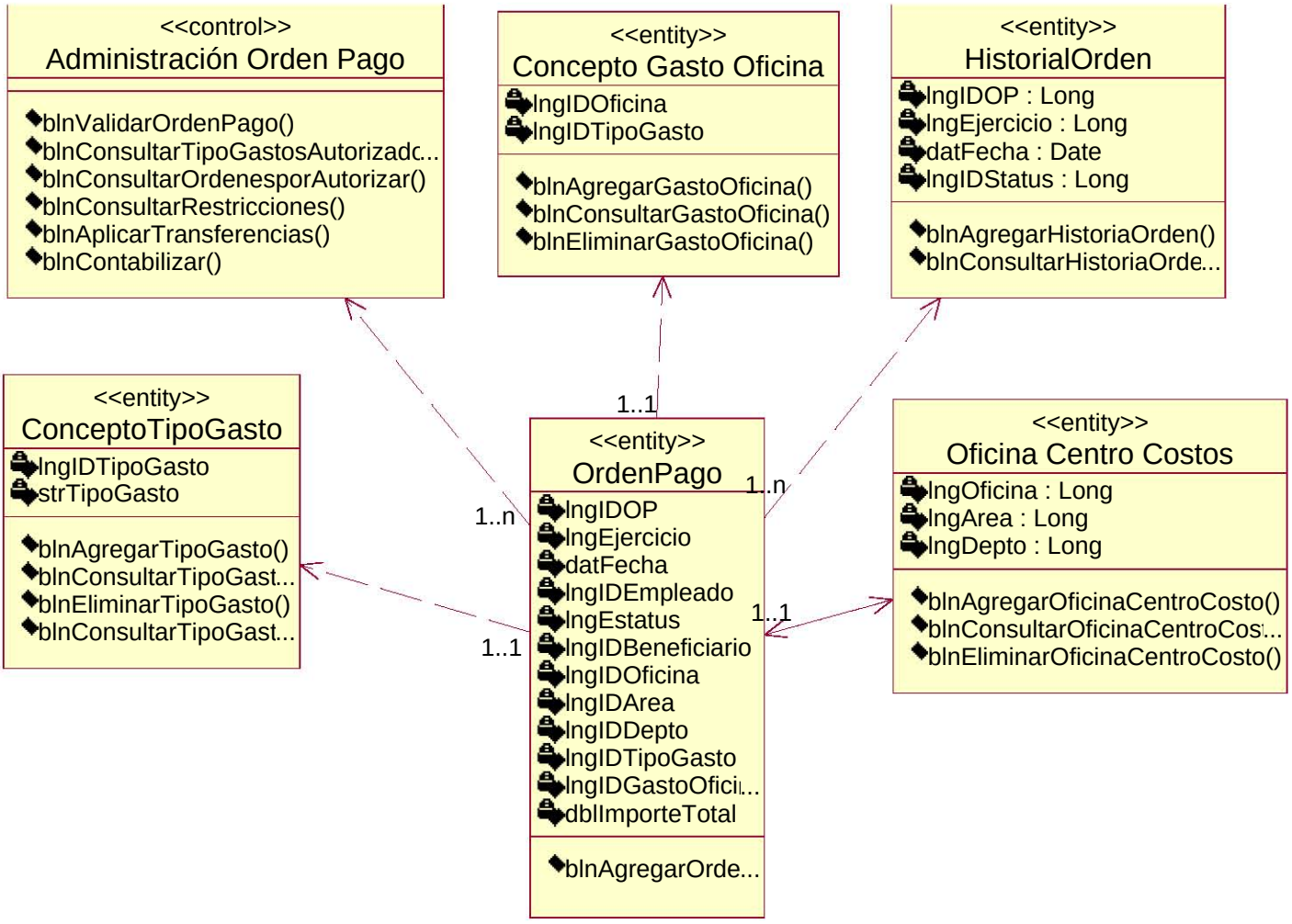
lngIDEmpleado
strApPaternoEmpleado
strApMaternoEmpleado
blnActivo
strPassword
lnIDOficina
lngIDArea
lngIDDepto
strLogin

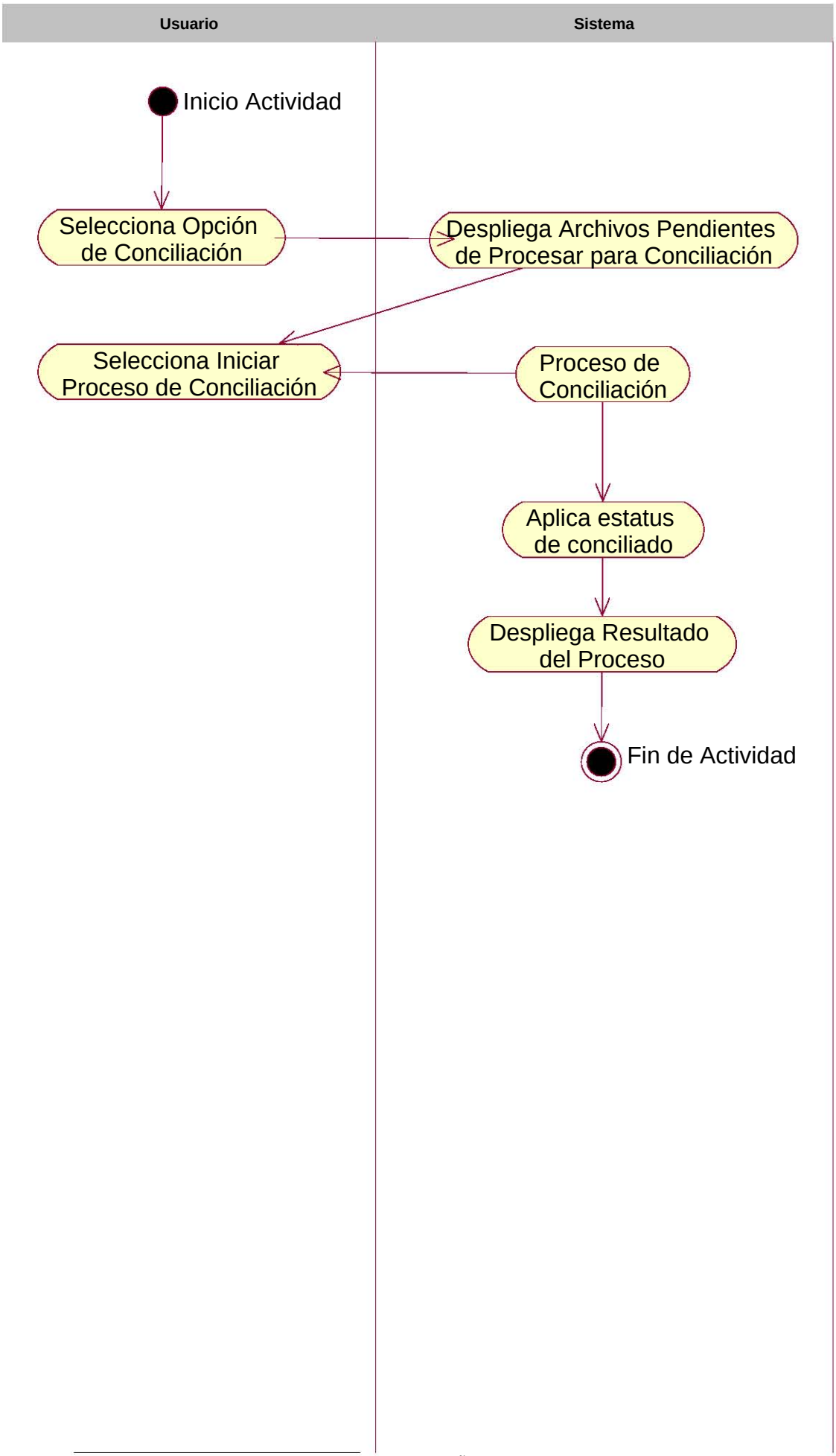
blnAgregaEmpleado()
rdsCargaPerfilEmpleado()
rdsListaEmpleado()

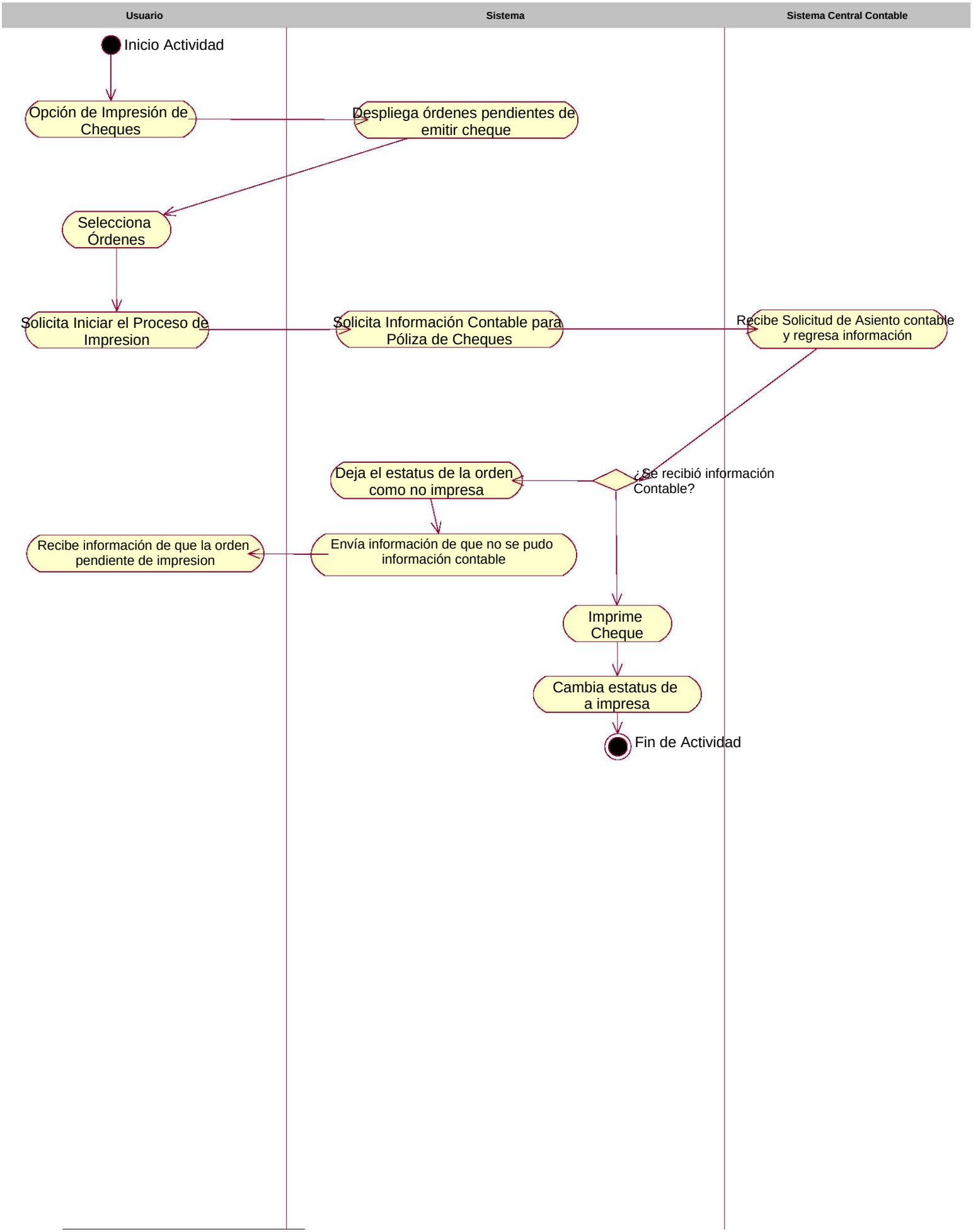

Registro Orden Pago
blnValidaregistroOr...
AutorizarOrdenes
AutorizaOrde...
ConsultaOrde...
CargarPerfilUsuario
blnCargaPerfilUsuari...
blnAgregaNiveldeAcce...
blnModificaPerfilUsua...
ConciliaciónBancaria
Conciliacio...
ConsultaOrdenesAutorizadas
staOrdenesAutorizac...
ConsultarOrdenesporAutorizar
ListaOrdenesporAutori...
ConsultarÓrdenesporImprimir
ListaOrdenesporImpri...
Contabilizar
ImportaInformacionConta...
Contabiliza()
ImprimirCheque
blnImprimrCheque()
blnListaOrdenesporImpri...
ValidarClaveAcceso
blnValidaClaveEmplea...
strEncriptaClaveAcces...
strDesencriptaClaveAc...
CrearArchivoLiberaciónCheques
CrearArchivoLiberaci...
CrearArhivoPagoTransferencia
CrearArchivoPagoTransfere...
ImportarOrdenesSistemasCentrales
blnImportarOrdenesSI...

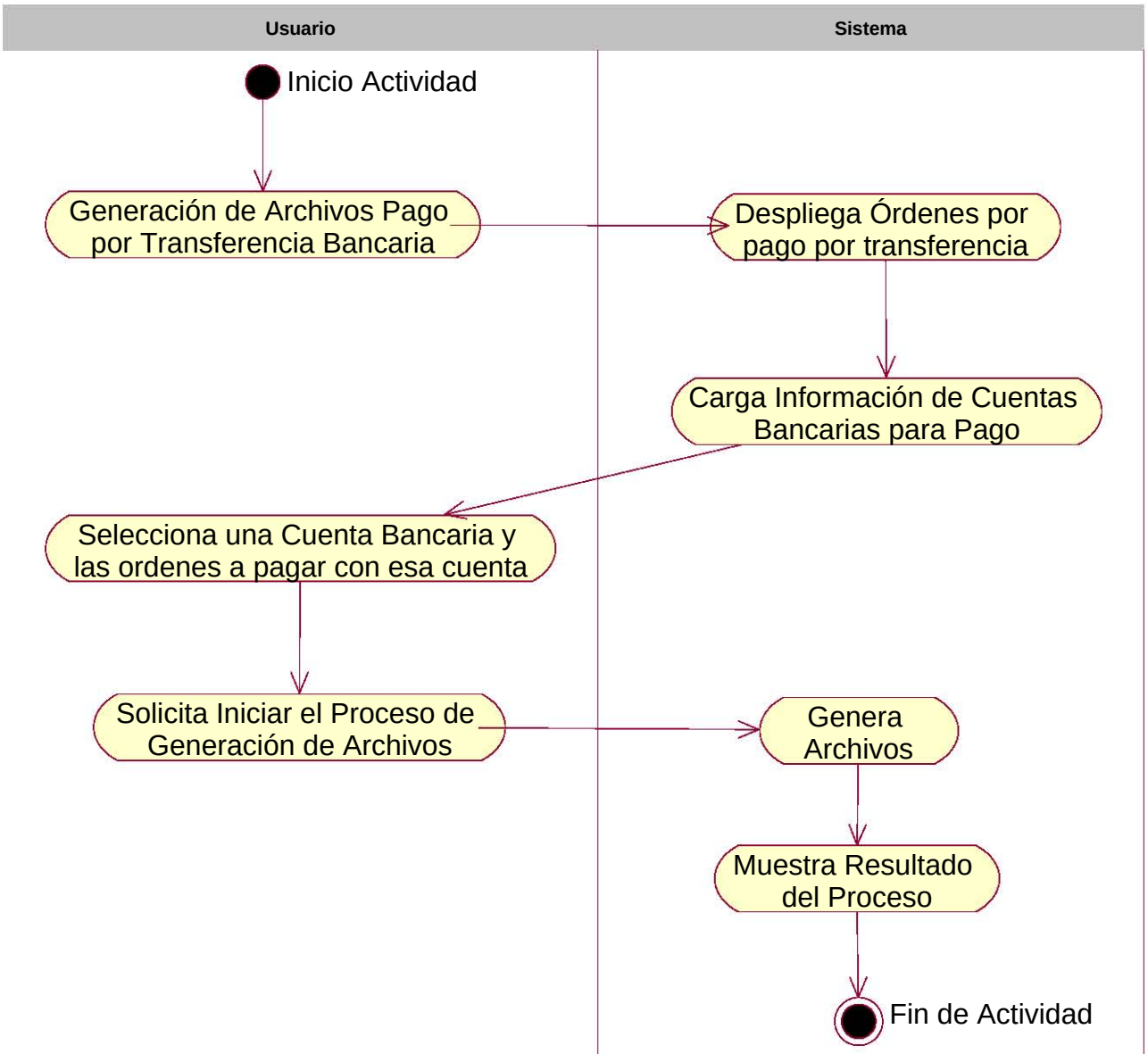

<<boundary>>
AutorizarOrdenes
AutorizaOrden()
ConsultaOrden()
<<boundary>>
CargarPerfilUsuario
blnCargaPerfilUsuario()
blnAgregaNiveldeAcce...
blnModificaPerfilUsuari...
<<boundary>>
ConsultarOrdenesporAutorizar
ListaOrdenesporAutorizar()
<<boundary>>
Contabilizar
ImportaInformacionContal...
Contabiliza()
<<boundary>>
CrearArhivoPagoTransferencia
CrearArchivoPagoTransferencias...
<<boundary>>
ConciliaciónBancaria
Conciliacion()
<<boundary>>
ConsultaOrdenesAutorizadas
ListaOrdenesAutorizadas()
<<boundary>>
ConsultarÓrdenesporImprimir
ListaOrdenesporImprimir()
<<boundary>>
CrearArchivoLiberaciónCheques
CrearArchivoLiberación()
<<boundary>>
ImportarOrdenesSistemasCentrales
blnImportarOrdenesSISE()
<<boundary>>
ImprimirCheque
blnImprimrCheque()
blnListaOrdenesporImprin...
<<boundary>>
Registro Orden Pago
blnValidaregistroOrde...
<<boundary>>
ValidarClaveAcceso
blnValidaClaveEmpleado()
strEncriptaClaveAcceso()
strDesencriptaClaveAcce...

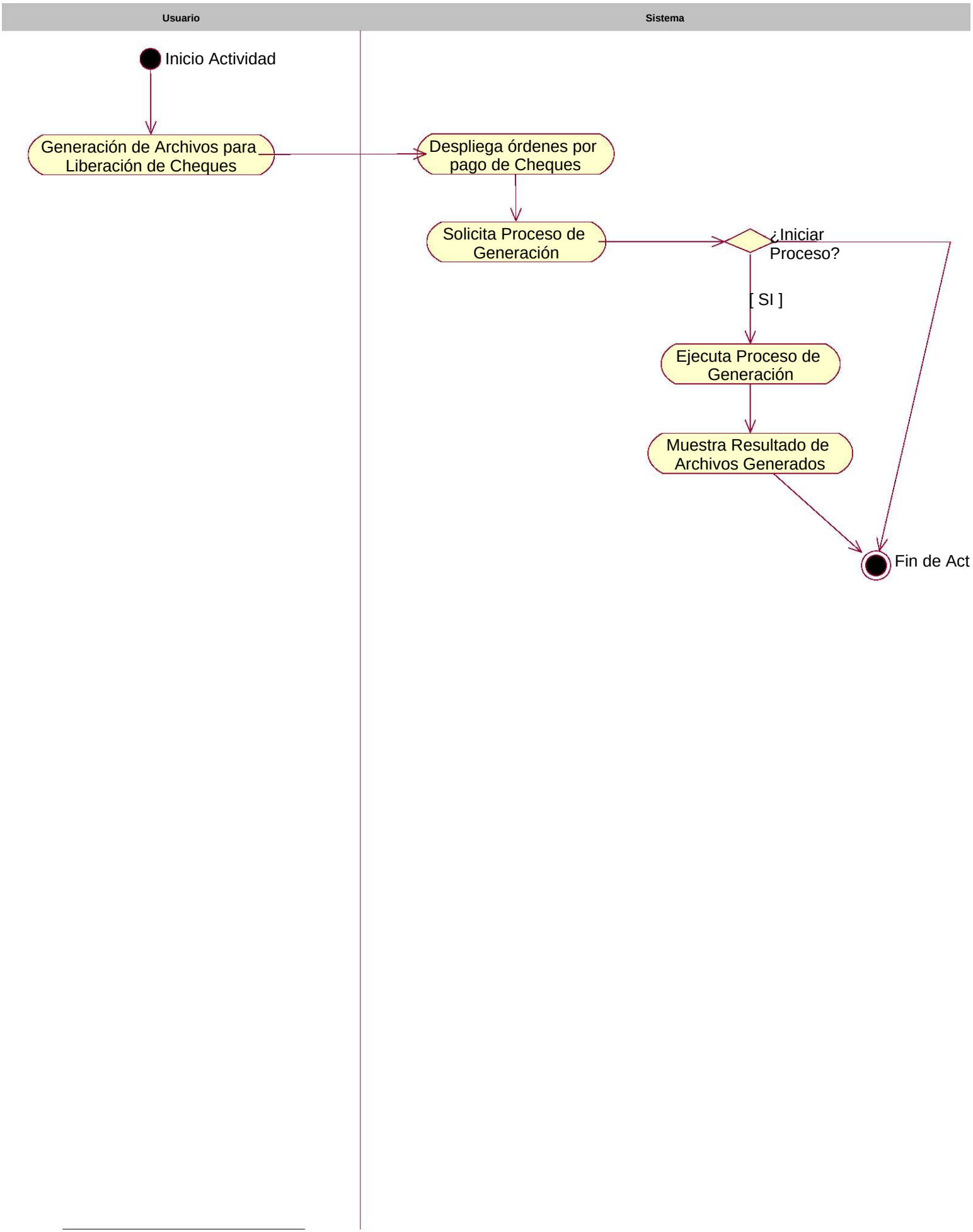

<<boundary>>
AutorizarOrdenes
<<boundary>>
CargarPerfilUsuario
<<boundary>>
ConciliaciónBancaria
<<boundary>>
ConsultaOrdenesAutorizadas
<<boundary>>
Contabilizar
<<boundary>>
ConsultarOrdenesporAutorizar
<<boundary>>
ConsultarÓrdenesporImprimir
<<boundary>>
ValidarClaveAcceso
<<boundary>>
ImportarOrdenesSistemasCentrales
<<boundary>>
Registro Orden Pago
<<boundary>>
ImprimirCheque
<<boundary>>
CrearArhivoPagoTransferencia
<<boundary>>
CrearArchivoLiberaciónCheques

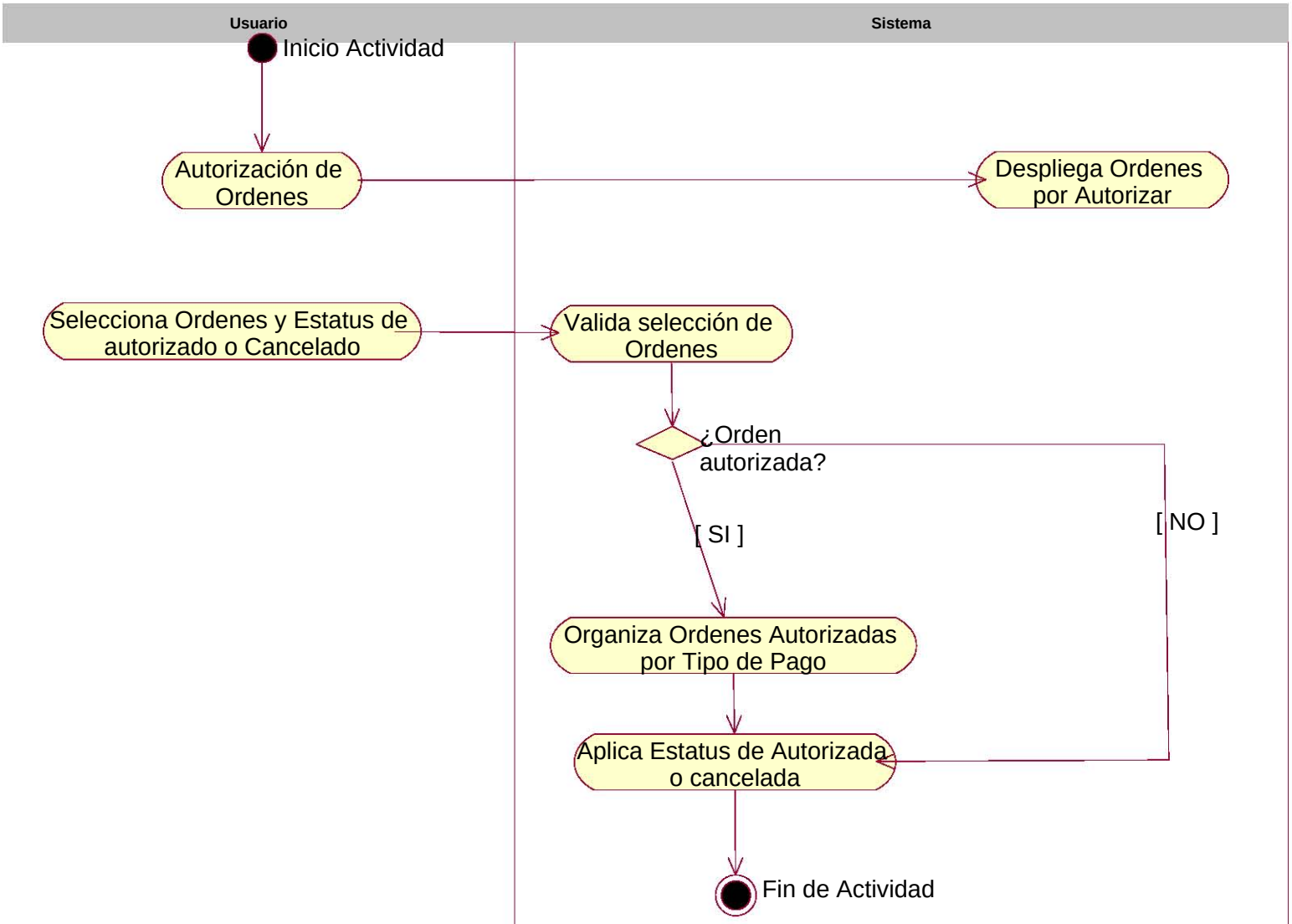

<<control>>
Administración Orden Pago
blnValidarOrdenPago()
blnConsultarTipoGastosAutorizadc...
blnConsultarOrdenesporAutorizar()
blnConsultarRestricciones()
blnAplicarTransferencias()
blnContabilizar()
<<entity>>
Concepto Gasto Oficina
lngIDOficina
lngIDTipoGasto
blnAgregarGastoOficina()
blnConsultarGastoOficina()
blnEliminarGastoOficina()
<<entity>>
HistorialOrden
lngIDOP : Long
lngEjercicio : Long
datFecha : Date
lngIDStatus : Long
blnAgregarHistoriaOrden()
blnConsultarHistoriaOrde...
<<entity>>
ConceptoTipoGasto
lngIDTipoGasto
strTipoGasto
blnAgregarTipoGasto()
blnConsultarTipoGast...
blnEliminarTipoGasto()
blnConsultarTipoGast...
1..1
1..n
1..n
1..1
1..1
<<entity>>
OrdenPago
lngIDOP
lngEjercicio
datFecha
lngIDEmpleado
lngEstatus
lngIDBeneficiario
lngIDOficina
lngIDArea
lngIDDepto
lngIDTipoGasto
lngIDGastoOfici...
dblImporteTotal
blnAgregarOrde...
<<entity>>
Oficina Centro Costos
lngOficina : Long
lngArea : Long
lngDepto : Long
blnAgregarOficinaCentroCosto()
blnConsultarOficinaCentroCos...
blnEliminarOficinaCentroCosto()

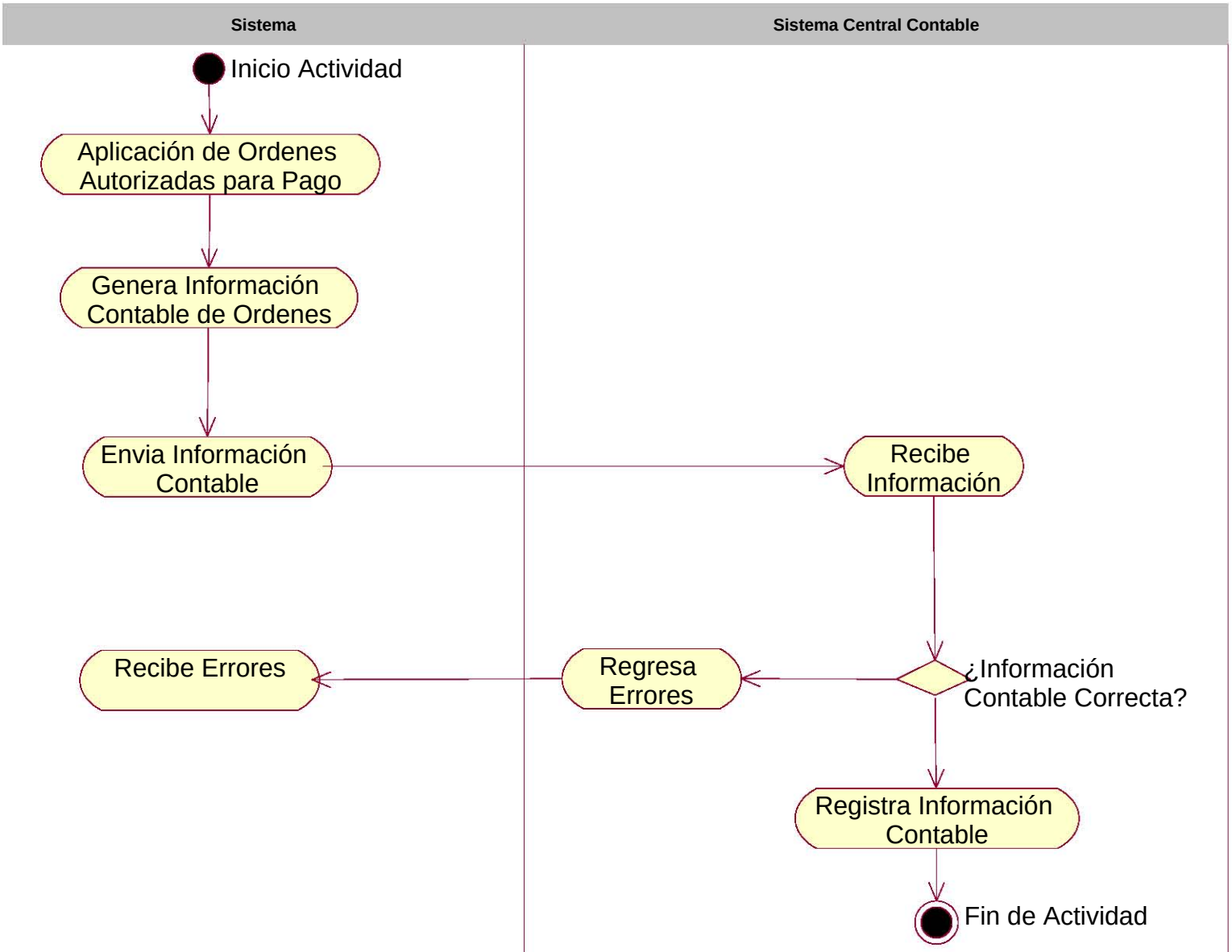

Usuario
Sistema
Inicio Actividad
Selecciona Opción de Conciliación
Despliega Archivos Pendientes de Procesar para Conciliación
Selecciona Iniciar Proceso de Conciliación
Proceso de Conciliación
Aplica estatus de conciliado
Despliega Resultado del Proceso
Fin de Actividad

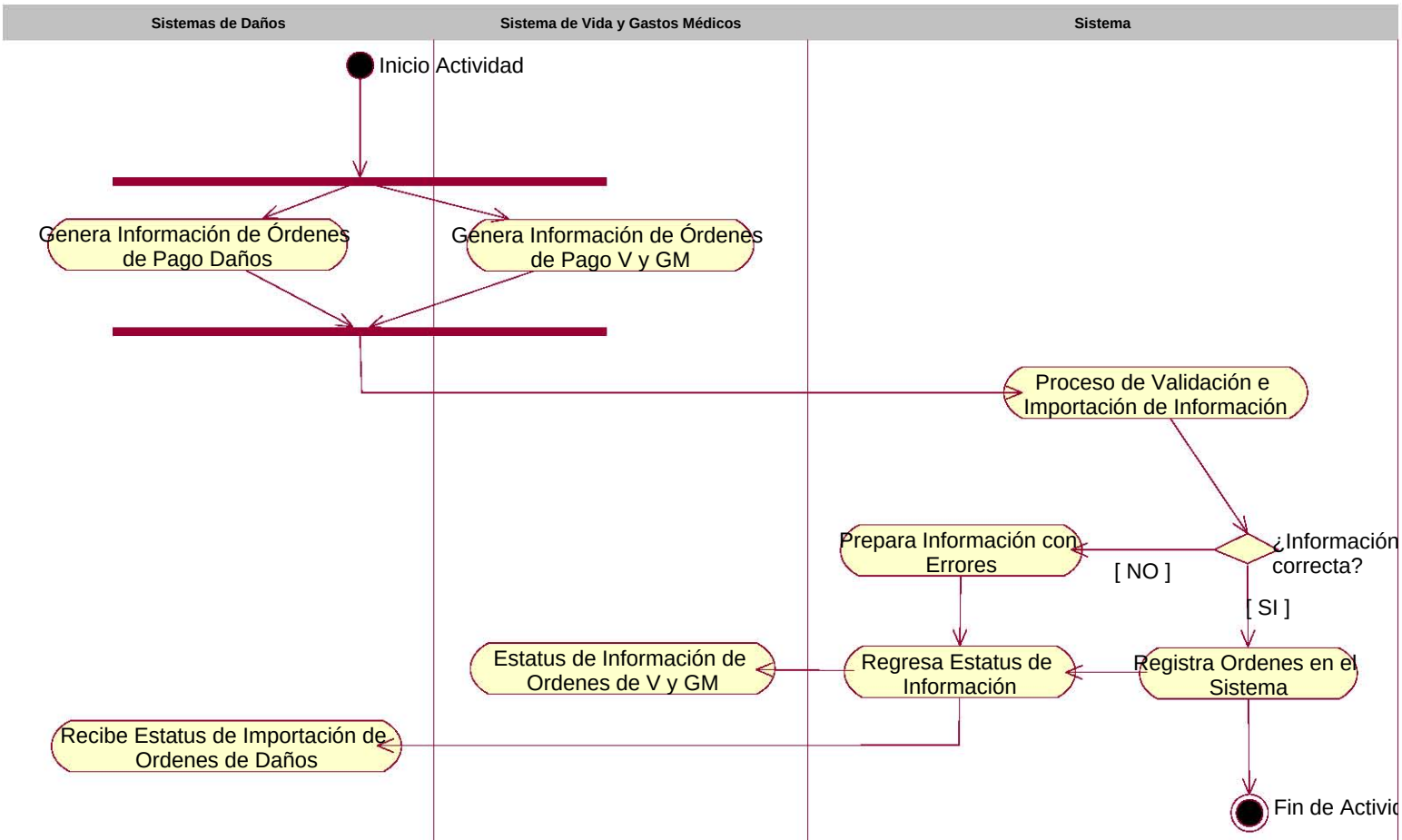

Usuario
Sistema
Sistema Central Contable
Inicio Actividad
Opción de Impresión de Cheques
Despliega órdenes pendientes de emitir cheque
Selecciona Órdenes
Solicita Iniciar el Proceso de Impresion
Solicita Información Contable para Póliza de Cheques
Recibe Solicitud de Asiento contable y regresa información
¿Se recibió información Contable?
Deja el estatus de la orden como no impresa
Envía información de que no se pudo información contable
Recibe información de que la orden pendiente de impresion
Imprime Cheque
Cambia estatus de a impresa
Fin de Actividad

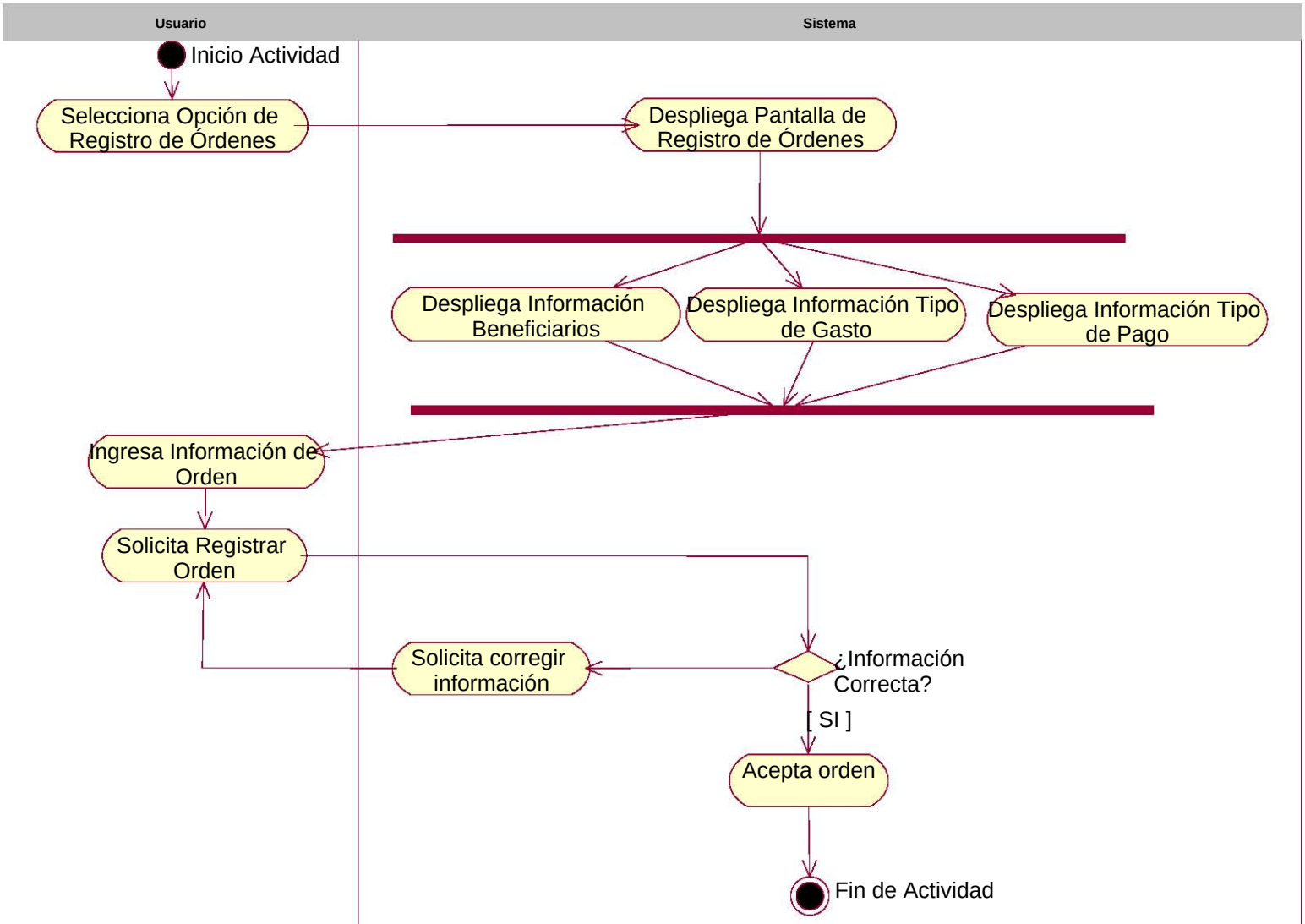

Usuario
Sistema
Inicio Actividad
Generación de Archivos Pago por Transferencia Bancaria
Despliega Órdenes por pago por transferencia
Carga Información de Cuentas Bancarias para Pago
Selecciona una Cuenta Bancaria y las ordenes a pagar con esa cuenta
Solicita Iniciar el Proceso de Generación de Archivos
Genera Archivos
Muestra Resultado del Proceso
Fin de Actividad

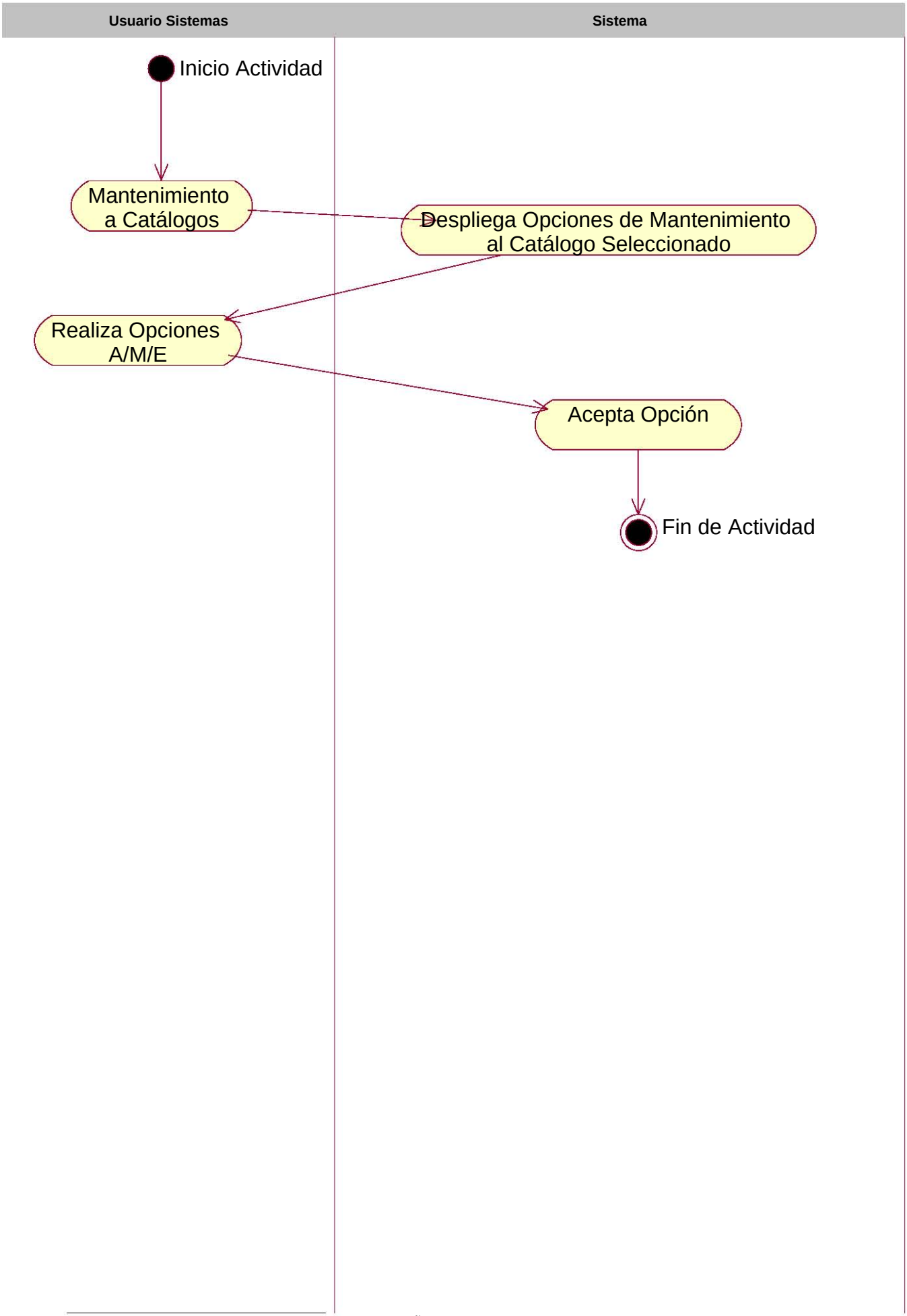

Usuario
Sistema
Inicio Actividad
Generación de Archivos para Liberación de Cheques
Despliega órdenes por pago de Cheques
Solicita Proceso de Generación
¿Iniciar Proceso?
[SI]
Ejecuta Proceso de Generación
Muestra Resultado de Archivos Generados
Fin de Act

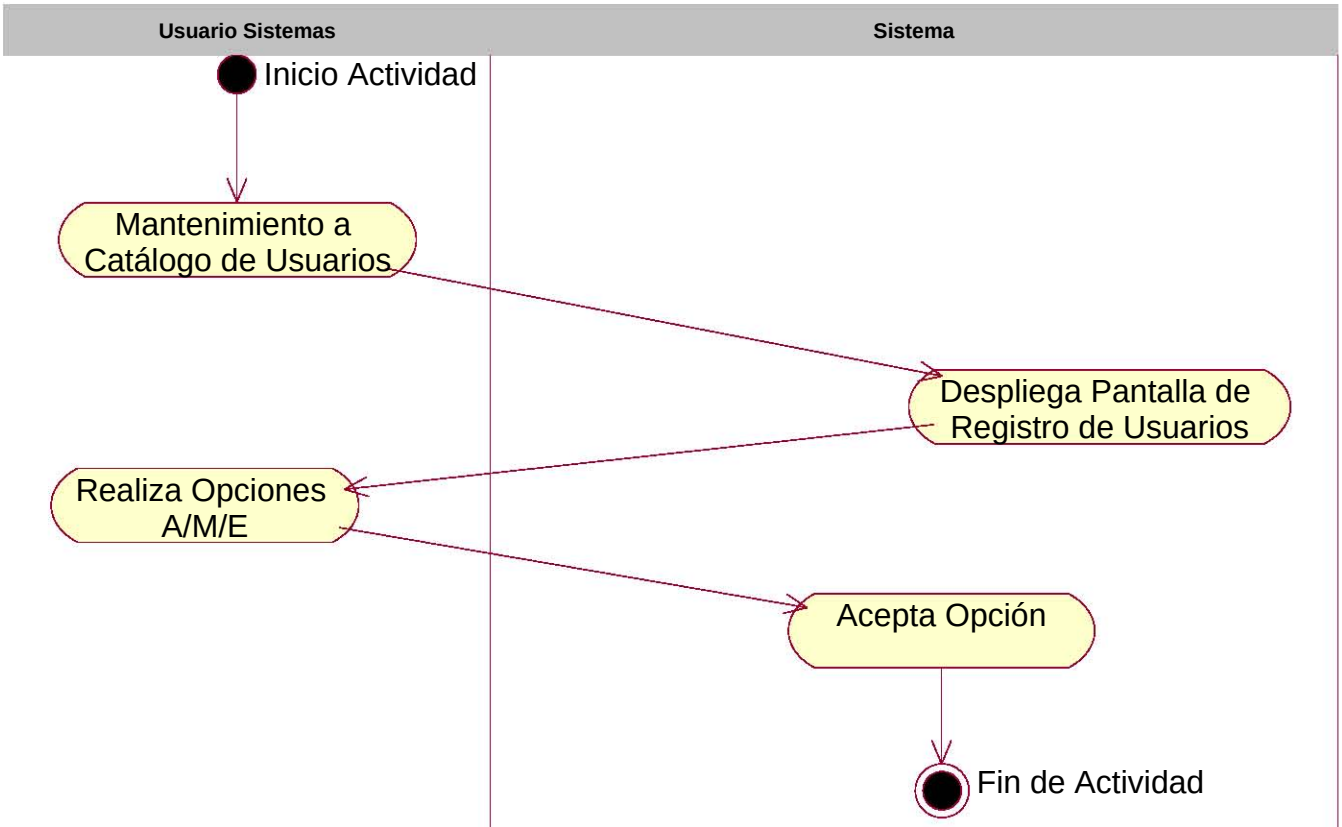

Usuario
Sistema
Inicio Actividad
Autorización de Ordenes
Despliega Ordenes por Autorizar
Selecciona Ordenes y Estatus de autorizado o Cancelado
Valida selección de Ordenes
¿Orden autorizada?
[SI]
[NO]
Organiza Ordenes Autorizadas por Tipo de Pago
Aplica Estatus de Autorizada o cancelada
Fin de Actividad

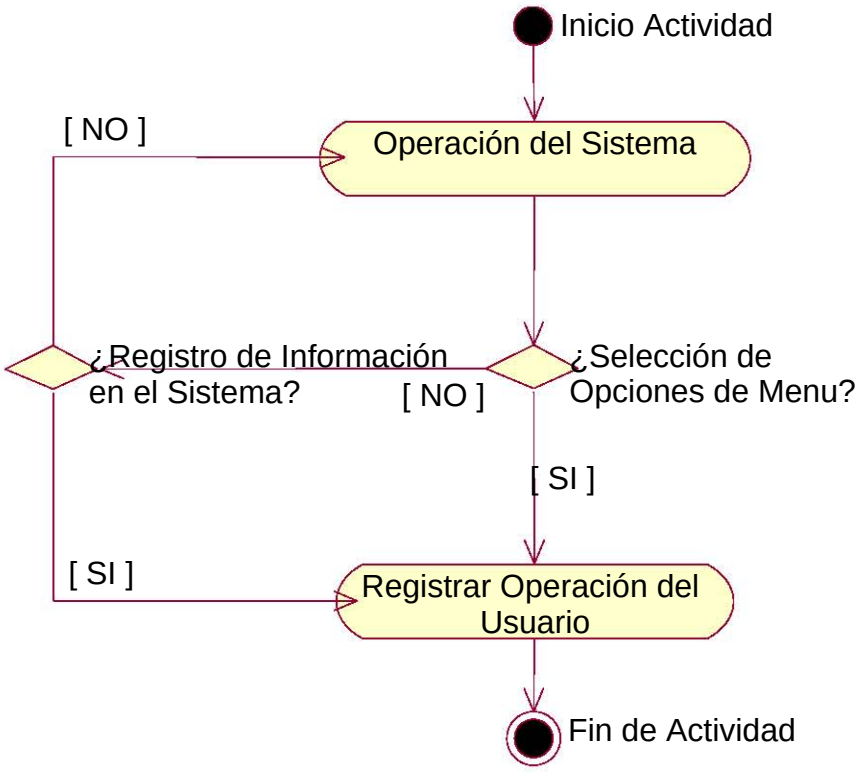

Sistema
Sistema Central Contable
Inicio Actividad
Aplicación de Ordenes Autorizadas para Pago
Genera Información Contable de Ordenes
Envia Información Contable
Recibe Información
¿Información Contable Correcta?
Regresa Errores
Recibe Errores
Registra Información Contable
Fin de Actividad

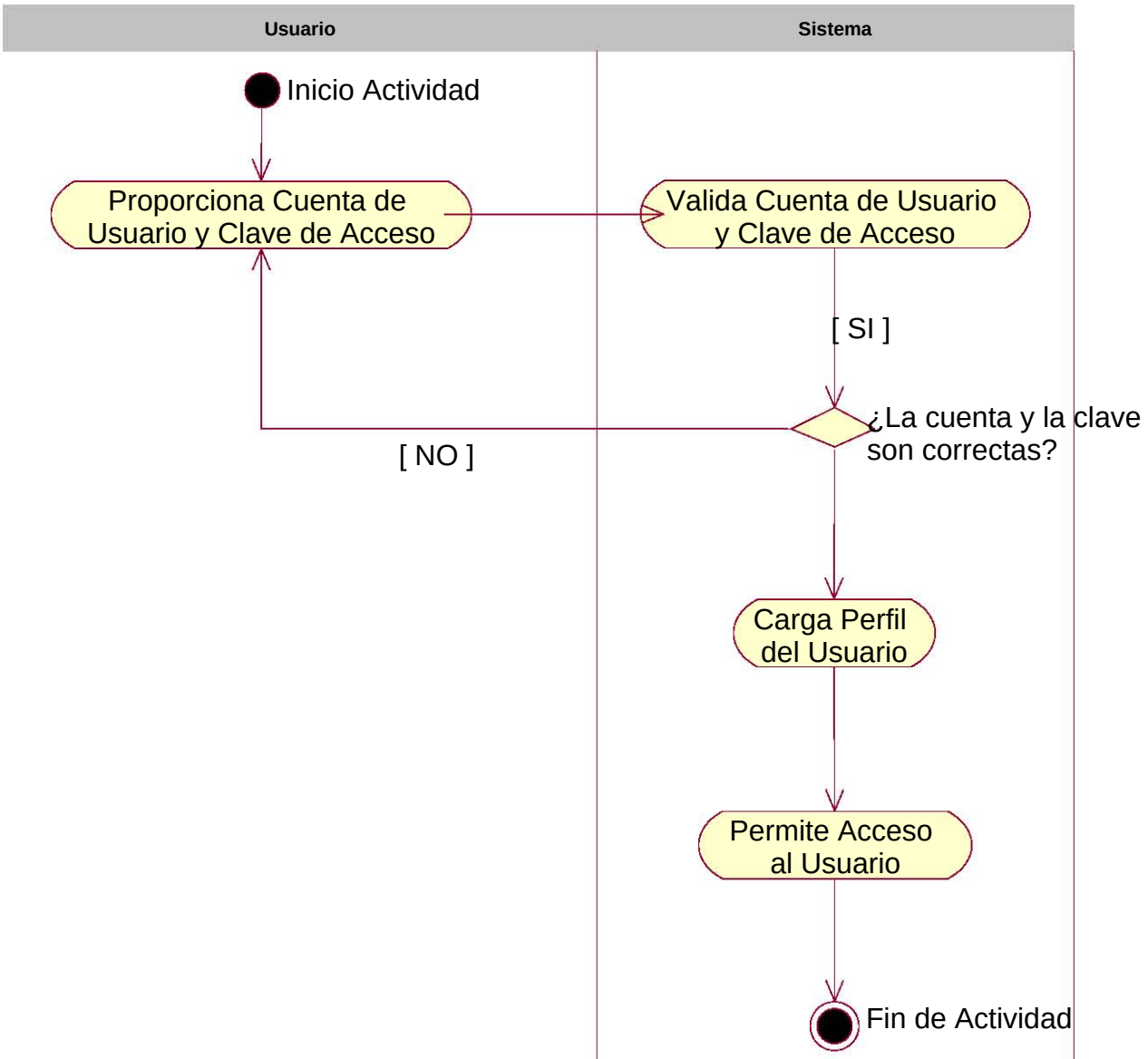

Sistemas de Daños
Sistema de Vida y Gastos Médicos
Sistema
Inicio Actividad
Genera Información de Órdenes de Pago Daños
Genera Información de Órdenes de Pago V y GM
Proceso de Validación e Importación de Información
Prepara Información con Errores
¿Información correcta?
[NO]
[SI]
Estatus de Información de Ordenes de V y GM
Regresa Estatus de Información
Registra Ordenes en el Sistema
Recibe Estatus de Importación de Ordenes de Daños
Fin de Activid

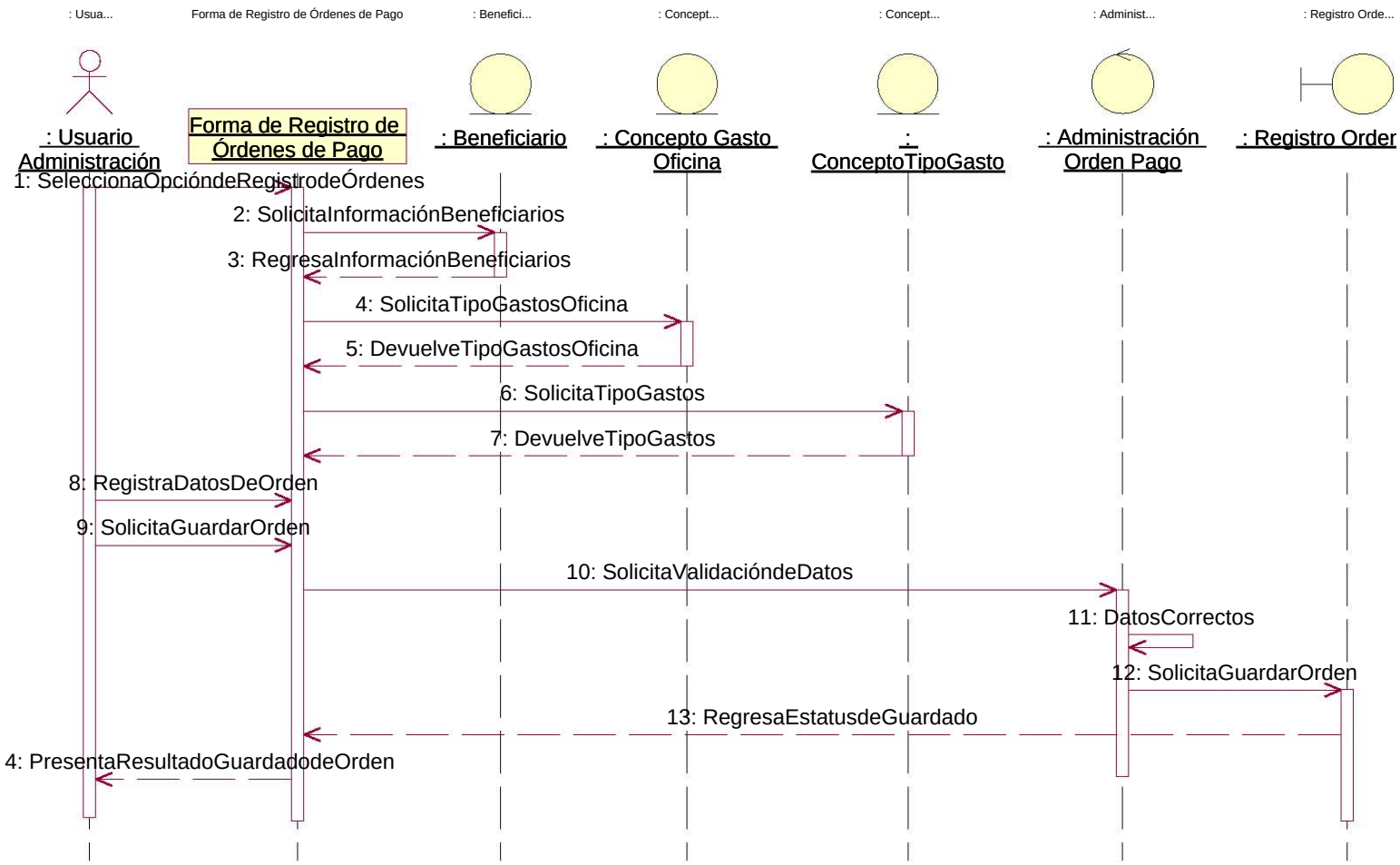

Usuario
Sistema
Inicio Actividad
Selecciona Opción de Registro de Órdenes
Despliega Pantalla de Registro de Órdenes
Despliega Información Beneficiarios
Despliega Información Tipo de Gasto
Despliega Información Tipo de Pago
Ingresa Información de Orden
Solicita Registrar Orden
Solicita corregir información
¿Información Correcta?
[SI]
Acepta orden
Fin de Actividad

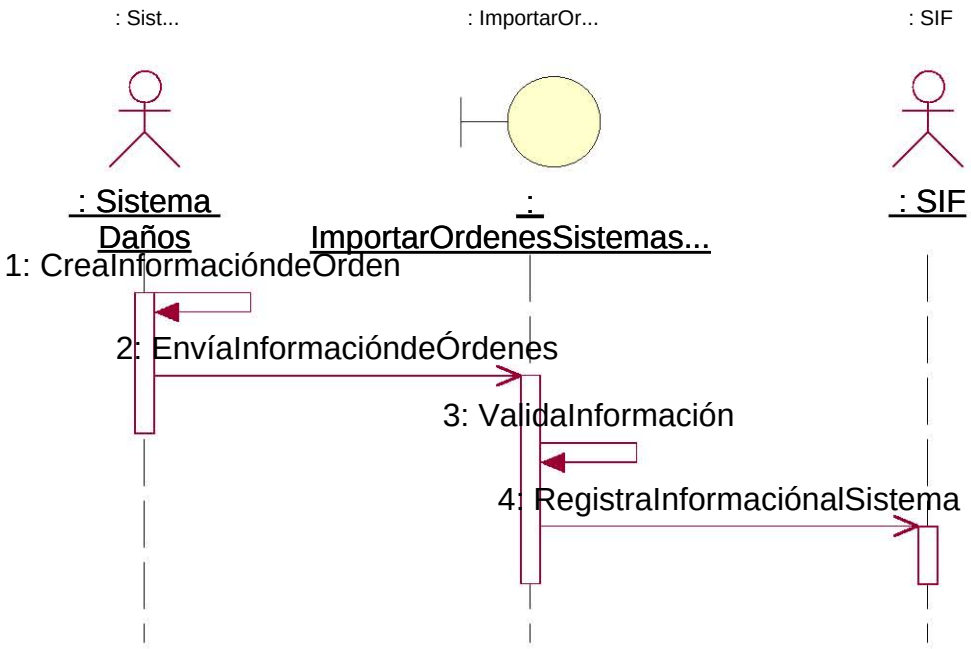

Usuario Sistemas
Sistema
Inicio Actividad
Mantenimiento a Catálogos
Despliega Opciones de Mantenimiento al Catálogo Seleccionado
Realiza Opciones A/M/E
Acepta Opción
Fin de Actividad

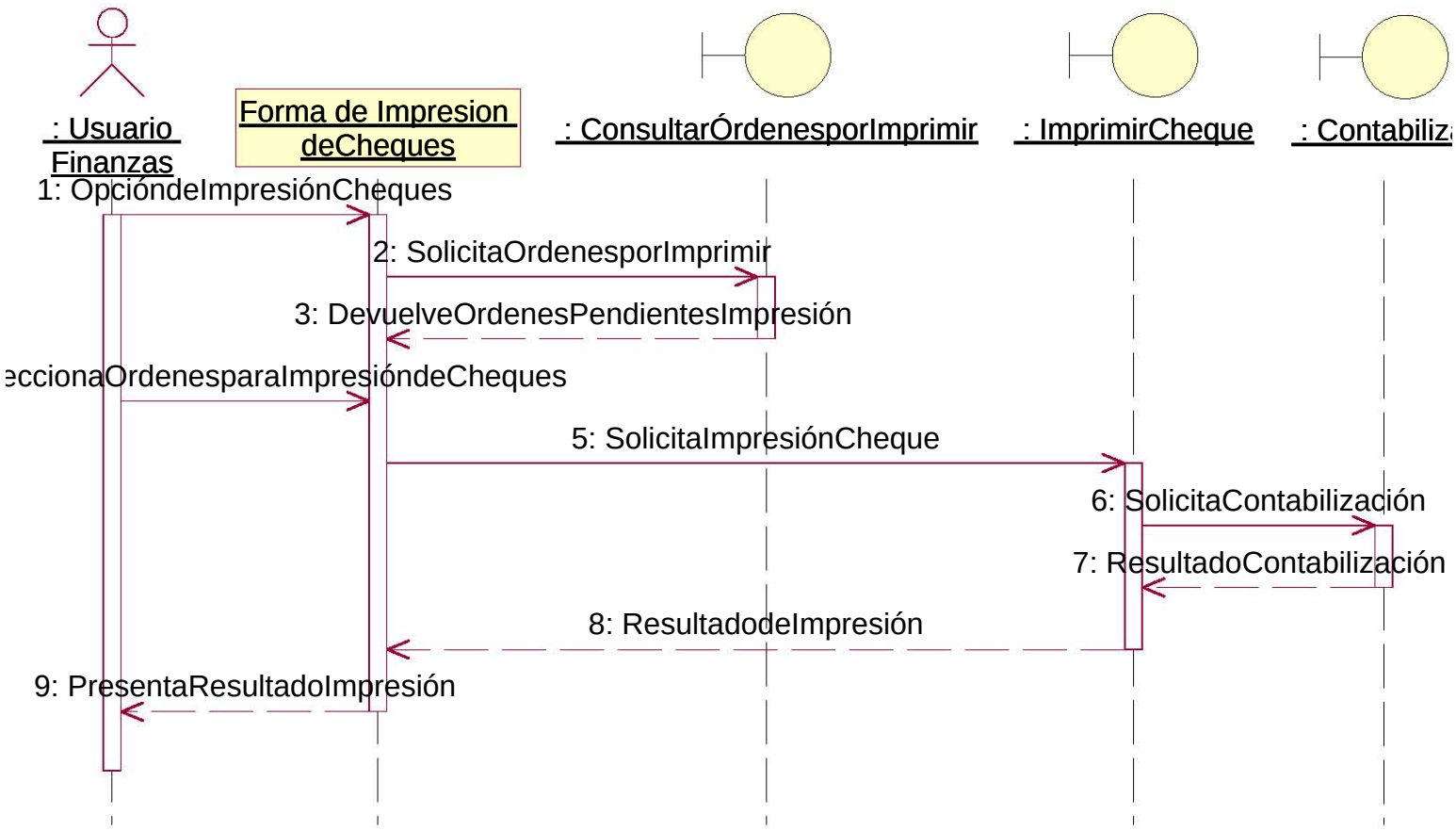

Usuario Sistemas
Sistema
Inicio Actividad
Mantenimiento a Catálogo de Usuarios
Despliega Pantalla de Registro de Usuarios
Realiza Opciones A/M/E
Acepta Opción
Fin de Actividad

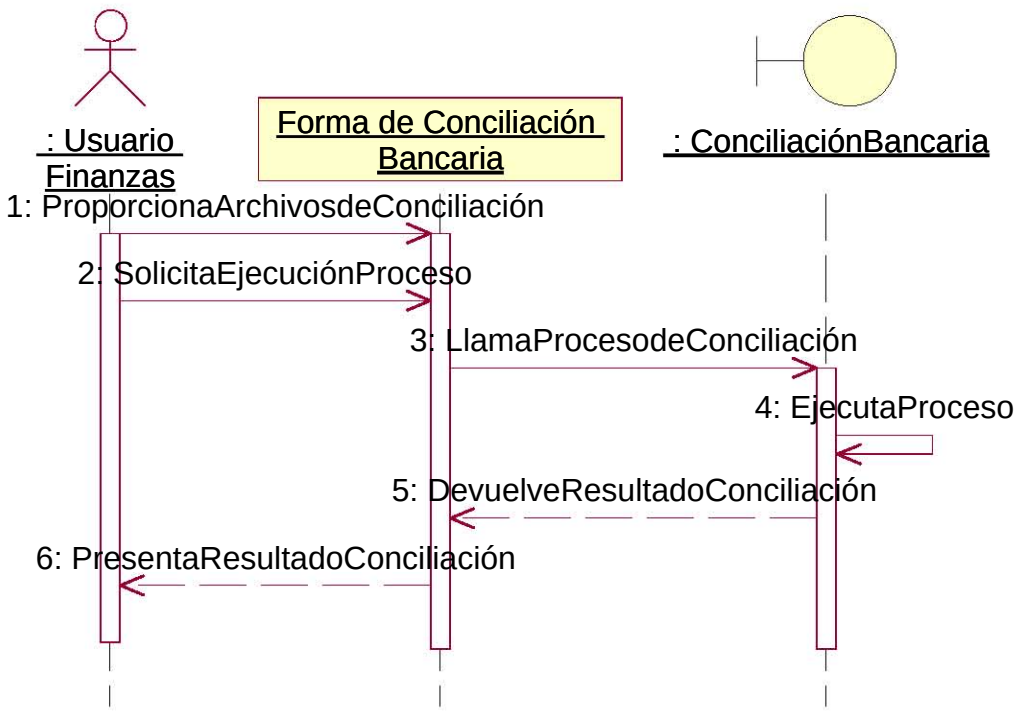

Inicio Actividad
[NO]
Operación del Sistema
¿Registro de Información en el Sistema?
[NO]
¿Selección de Opciones de Menu?
[SI]
[SI]
Registrar Operación del Usuario
Fin de Actividad

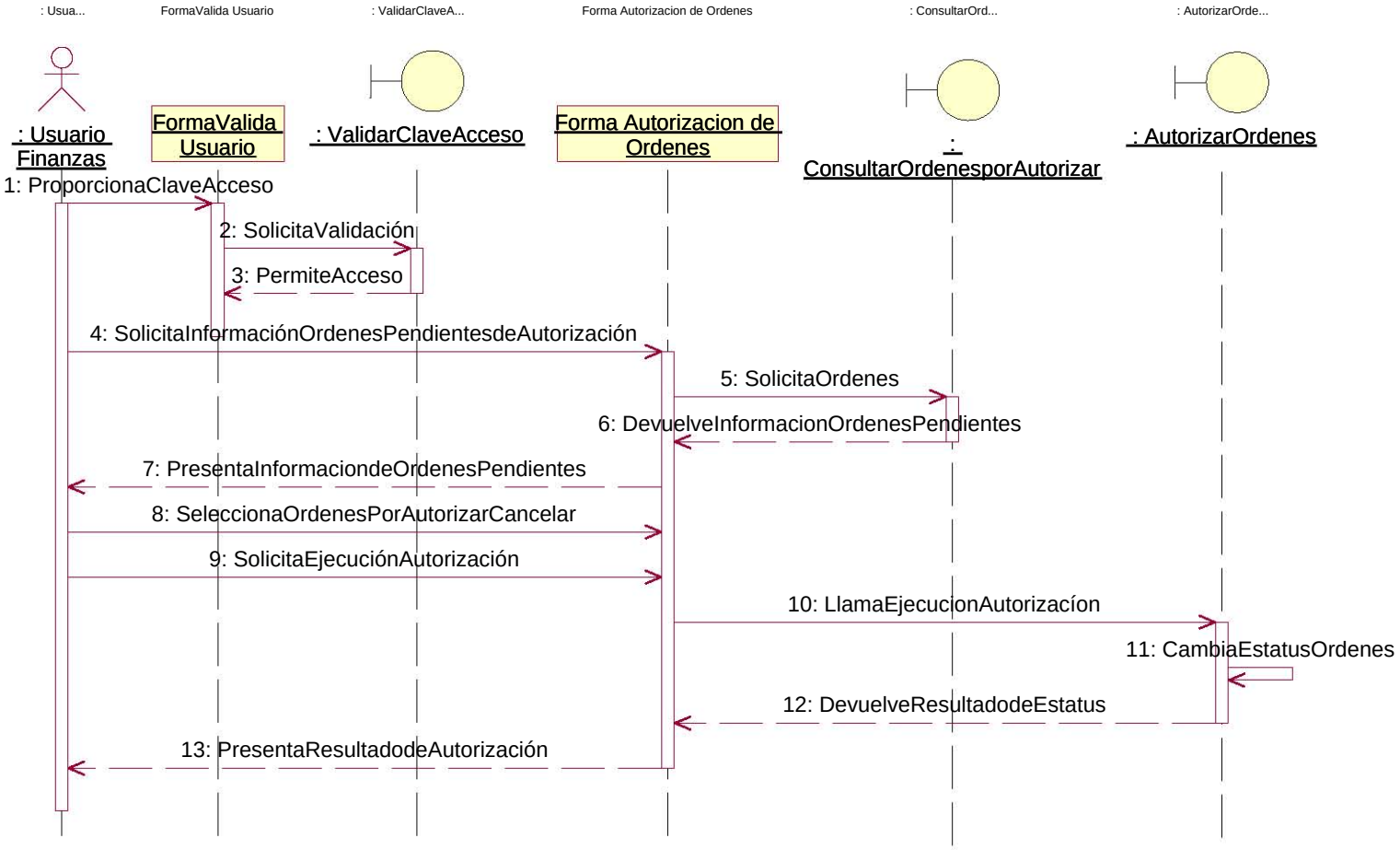

Usuario
Sistema
Inicio Actividad
Proporciona Cuenta de Usuario y Clave de Acceso
Valida Cuenta de Usuario y Clave de Acceso
[SI]
¿La cuenta y la clave son correctas?
[NO]
Carga Perfil del Usuario
Permite Acceso al Usuario
Fin de Actividad

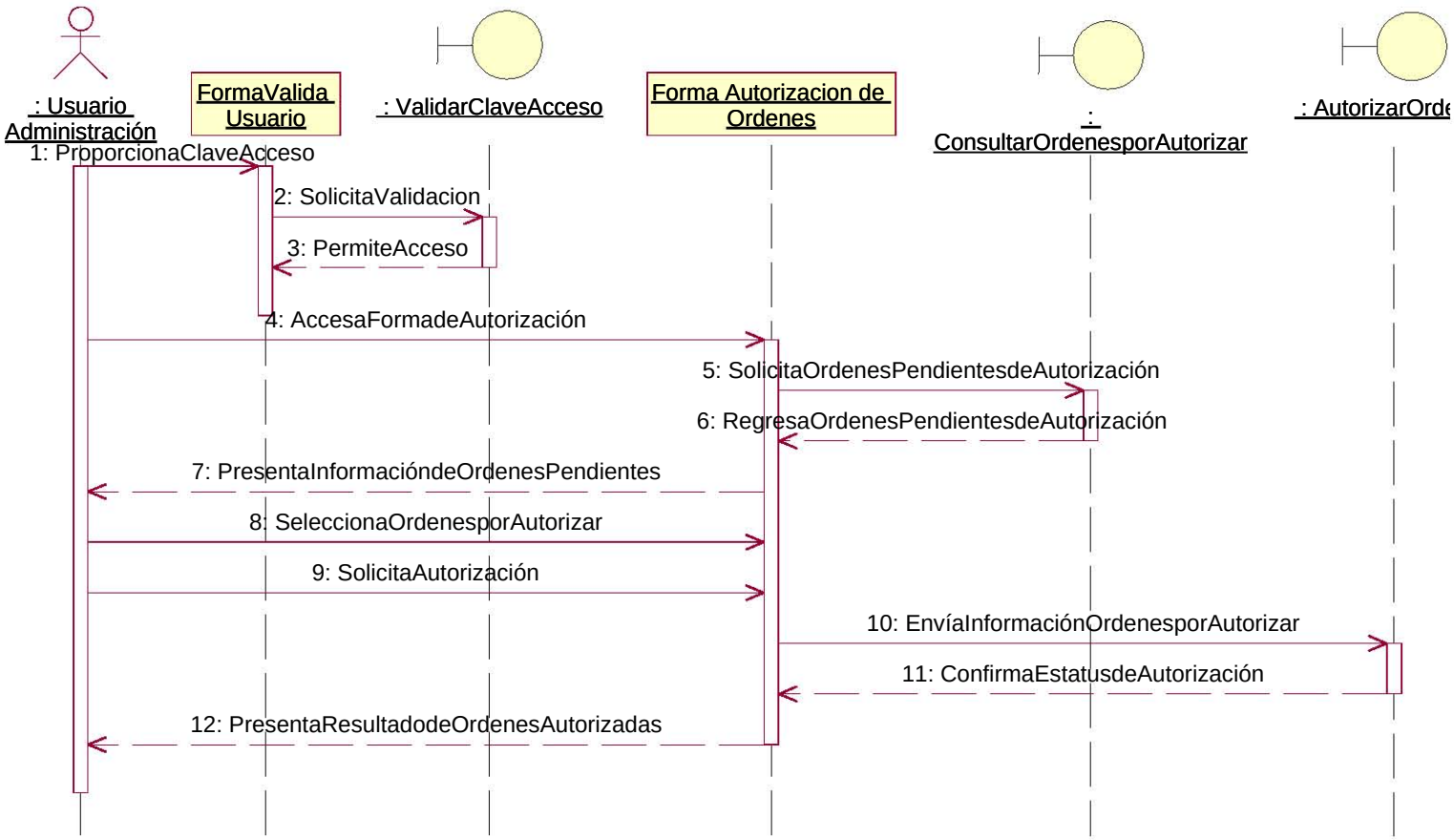

: Usua...
Forma de Registro de Órdenes de Pago
: Benefici...
: Concept...
: Concept...
: Administ...
: Registro Orde...
: Usuario Administración
Forma de Registro de Órdenes de Pago
: Beneficiario
: Concepto Gasto Oficina
: ConceptoTipoGasto
: Administración Orden Pago
: Registro Order
1: SeleccionaOpciónde RegistrodeÓrdenes
2: SolicitaInformaciónBeneficiarios
3: RegresaInformaciónBeneficiarios
4: SolicitaTipoGastosOficina
5: DevuelveTipoGastosOficina
6: SolicitaTipoGastos
7: DevuelveTipoGastos
8: RegistraDatosDeOrden
9: SolicitaGuardarOrden
10: SolicitaValidaciónde Datos
11: DatosCorrectos
12: SolicitaGuardarOrden
13: RegresaEstatusdeGuardado
4: PresentaResultadoGuardadodeOrden

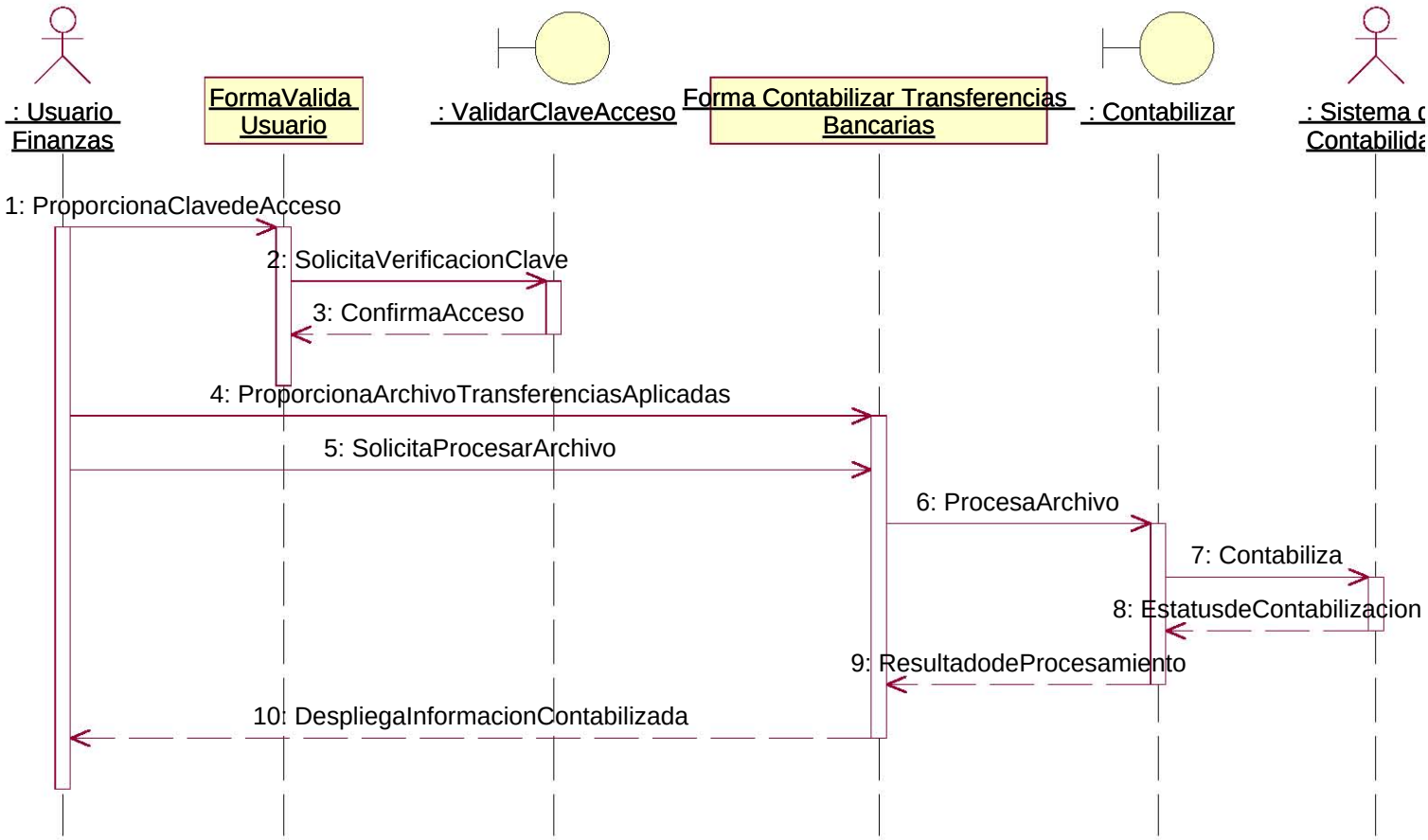

: Sist...
: ImportarOr...
: SIF
: Sistema Daños
: ImportarOrdenesSistemas...
: SIF
1: CreaInformacióndeOrden
2: EnvíaInformacióndeÓrdenes
3: ValidaInformación
4: RegistraInformaciónalSistema


: Usuario Finanzas
Forma de Impresion deCheques
: ConsultarÓrdenesporImprimir
: ImprimirCheque
: Contabiliz
1: OpcióndeImpresiónCheques
2: SolicitaOrdenesporImprimir
3: DevuelveOrdenesPendientesImpresión
eccionaOrdenesparaImpresiónde Cheques
5: SolicitaImpresiónCheque
6: SolicitaContabilización
7: ResultadoContabilización
8: ResultadodeImpresión
9: PresentaResultadoImpresión


: Usuario Finanzas
Forma de Conciliación Bancaria
: ConciliaciónBancaria
1: ProporcionaArchivosdeConciliación
2: SolicitaEjecuciónProceso
3: LlamaProcesodeConciliación
4: EjecutaProceso
5: DevuelveResultadoConciliación
6: PresentaResultadoConciliación


: Usua...
FormaValida Usuario
: ValidarClaveA...
Forma Autorizacion de Ordenes
: ConsultarOrd...
: AutorizarOrde...
: Usuario Finanzas
FormaValida Usuario
: ValidarClaveAcceso
Forma Autorizacion de Ordenes
: ConsultarOrdenesporAutorizar
: AutorizarOrdenes
1: ProporcionaClaveAcceso
2: SolicitaValidación
3: PermiteAcceso
4: SolicitaInformaciónOrdenesPendientesdeAutorización
5: SolicitaOrdenes
6: DevuelveInformacionOrdenesPendientes
7: PresentaInformaciondeOrdenesPendientes
8: SeleccionaOrdenesPorAutorizarCancelar
9: SolicitaEjecuciónAutorización
10: LlamaEjecucionAutorizacíon
11: CambiaEstatusOrdenes
12: DevuelveResultadodeEstatus
13: PresentaResultadodeAutorización


: Usuario Administración
FormaValida Usuario
: ValidarClaveAcceso
Forma Autorizacion de Ordenes
: ConsultarOrdenesporAutorizar
: AutorizarOrde
1: ProporcionaClaveAcceso
2: SolicitaValidacion
3: PermiteAcceso
4: AccesaFormadeAutorización
5: SolicitaOrdenesPendientesdeAutorización
6: RegresaOrdenesPendientesdeAutorización
7: PresentaInformacióndeOrdenesPendientes
8: SeleccionaOrdenesporAutorizar
9: SolicitaAutorización
10: EnvíaInformaciónOrdenesporAutorizar
11: ConfirmaEstatusdeAutorización
12: PresentaResultadodeOrdenesAutorizadas


: Usuario Finanzas
FormaValida Usuario
: ValidarClaveAcceso
Forma Contabilizar Transferencias Bancarias
: Contabilizar
: Sistema d Contabilida
1: ProporcionaClavedeAcceso
2: SolicitaVerificacionClave
3: ConfirmaAcceso
4: ProporcionaArchivoTransferenciasAplicadas
5: SolicitaProcesarArchivo
6: ProcesaArchivo
7: Contabiliza
8: EstatusdeContabilizacion
9: ResultadodeProcesamiento
10: DespliegaInformacionContabilizada

Tipo_Traspaso

	Column Name	Data Type	Length	Allow Nulls
🔑	IDTipo_Traspaso	int	4	
	Descripcion	varchar	30	

Ramos_D

	Colu
	IDRamo_D
	Ramo

umn Name	Data Type	Length	Allow Nulls
	int	4	
	int	4	

Parametros

	Column Name	Data Type	Length	Allow Nulls
	Numero_Copias	int	4	
	Periodo_Cancelacion	int	4	
	Universe_Usuario	varchar	15	
	Universe_Password	varchar	15	
	Universe_Path	varchar	30	
	Universe_Host	varchar	30	
	Edad_Max	smallint	2	
	Edad_Minima	smallint	2	
	Archivo_Cheque	varchar	80	
	Archivo_Caja	varchar	80	
	Archivo_OP	varchar	80	
	EnvioSISEOP	varchar	20	

Oficinas_D

	Column Name	Data Type	Length	Allow Nulls
	IDOficina_D	int	4	
	IDOficina	int	4	

Orden_Pago_SISE

	Column Name
🔑	IDOP
🔑	Ejercicio
🔑	IDMovimiento_SISE
	Asiento
	Caratula
	Fecha

Data Type	Length	Allow Nulls
int	4	
int	4	
int	4	
int	4	
int	4	
datetime	8	

GASTOS_RH

	Column Name	Data Type	Length	Allow Nulls
	FECHA	char	4	✔
	OFICINA	char	2	✔
	AREA	char	2	✔
	DEPARTAMENTO	char	2	✔
	CTA_CONTABLE	char	2	✔
	SUB_CTA	char	4	✔
	MONTO	char	13	✔

Gastos

	Column Name	Data Type	Length	Allow Nulls
🔑	IDGasto	int	4	
	IDFecha	int	4	
	IDCuenta	int	4	
	IDSubcuenta	int	4	
	Importe	money	8	
	Origen	char	3	✔
	IDOficina	int	4	✔

dtproperties

	Column Name	Data Type
🔑	id	int
	objectid	int
🔑	property	varchar
	[value]	varchar
	uvalue	nvarchar
	lvalue	image
	version	int

Prestador_Domicilio

	Column Name	Data Type	Length	Allow Nulls
🔑	IDTipo_Domic	int	4	
🔑	IDPrestador	int	4	
	Calle	varchar	50	
	Exterior	varchar	15	
	Interior	varchar	15	✔
	IDColonia	int	4	
	IDMunicipio	int	4	
	IDCiudad	int	4	
	IDEstado	int	4	

	Length	Allow Nulls
	4	
	4	✔
	64	
	255	✔
	255	✔
	16	✔
	4	

CPMunicipios

	Column Name	Data Type	Length	Allow Nulls
🔑	Clave	int	4	
	Mun_Numero	int	4	
	Descripcion	varchar	50	
	Edo_Numero	int	4	

CP

🔑

Tipo_Tel

	Column Name	Data Type	Length	Allow Nulls
🔑	IDTipo_Tel	int	4	
	Tipo_Tel	varchar	15	

Estados

Column Name	Data Type	Length	Allow Nulls
Clave	int	4	
Edo_Numero	int	4	
Descripcion	varchar	30	

CPColonias

	Column Name	Data Type	Length
🔑	Clave	int	4
	Edo_Numero	int	4
	Mun_Numero	int	4
	Cd_Numero	int	4
	Descripcion	varchar	120
	CP	int	4

Movimiento

	Column Name	Data Type	Length	Allow Nulls
🔑	IDMovimiento	int	4	
	Descripcion	varchar	30	

FK_Bitacora_Movimiento

Allow Nulls

CPCiudades

	Column Name	Data Type	Length	Allow Nulls
🔑	Clave	int	4	
	Cd_Numero	int	4	
	Descripcion	varchar	50	
	Edo_Numero	int	4	

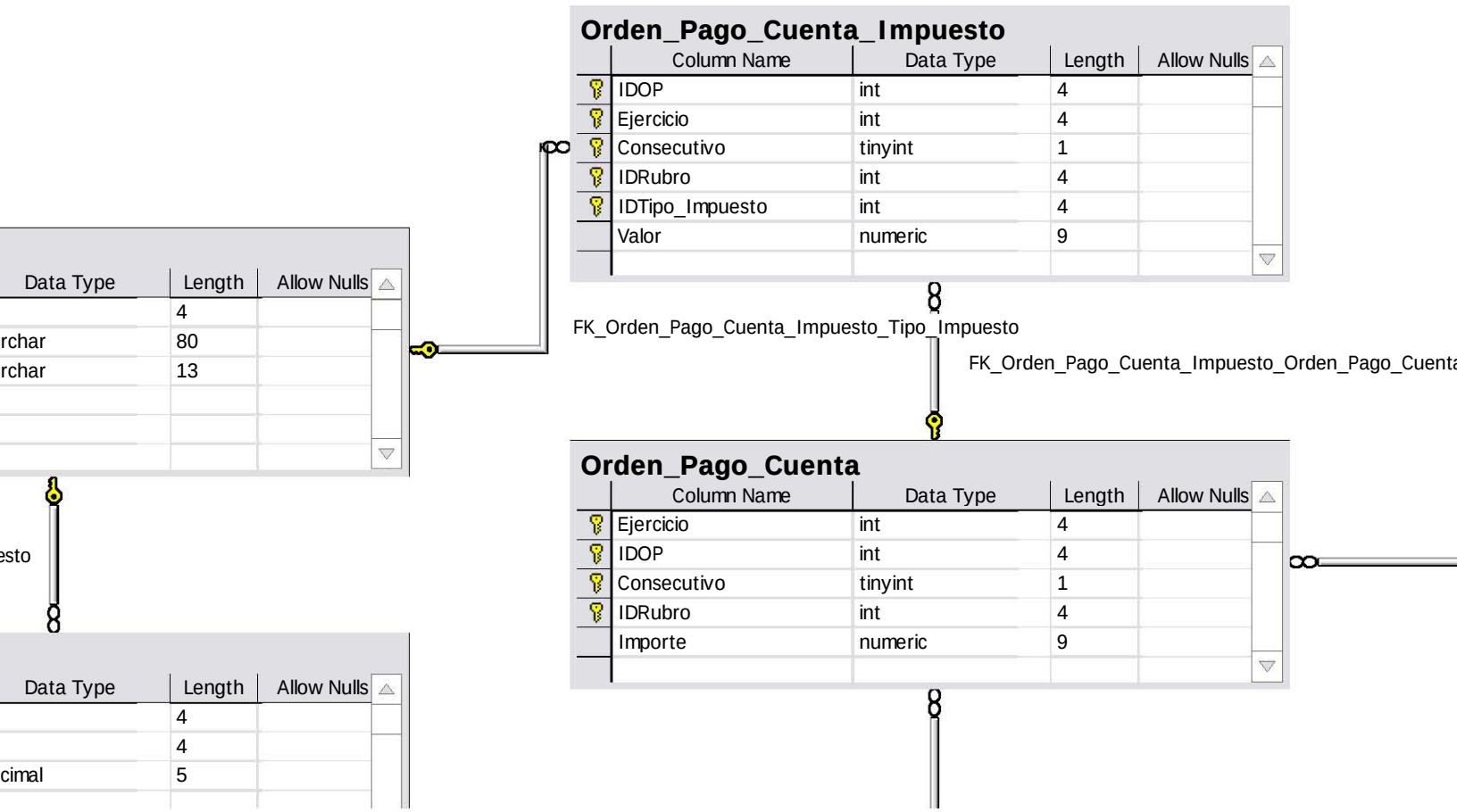
Tipo_Impuesto

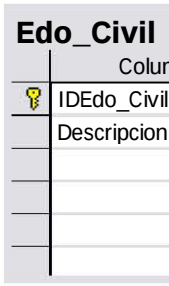
	Column Name	
🔑	IDTipo_Impuesto	int
	Nombre	va
	Cuenta	va

FK_Oficina_Impuesto_Tipo_Impue

Oficina_Impuesto

	Column Name	
🔑	IDOficina	int
🔑	IDTipo_Impuesto	int
	Valor	de


Orden_Pago_Cuenta_Impuesto
Column Name
Data Type
Length
Allow Nulls
IDOP
int
4
Ejercicio
int
4
Consecutivo
tinyint
1
IDRubro
int
4
IDTipo_Impuesto
int
4
Valor
numeric
9
FK_Orden_Pago_Cuenta_Impuesto_Tipo_Impuesto
FK_Orden_Pago_Cuenta_Impuesto_Orden_Pago_Cuenta
Orden_Pago_Cuenta
Column Name
Data Type
Length
Allow Nulls
Ejercicio
int
4
IDOP
int
4
Consecutivo
tinyint
1
IDRubro
int
4
Importe
numeric
9
Data Type
Length
Allow Nulls
4
rchar
80
rchar
13
esto
Data Type
Length
Allow Nulls
4
4
cimal
5



a

FK_Prestador_Domicilio_SISE_Prestadores

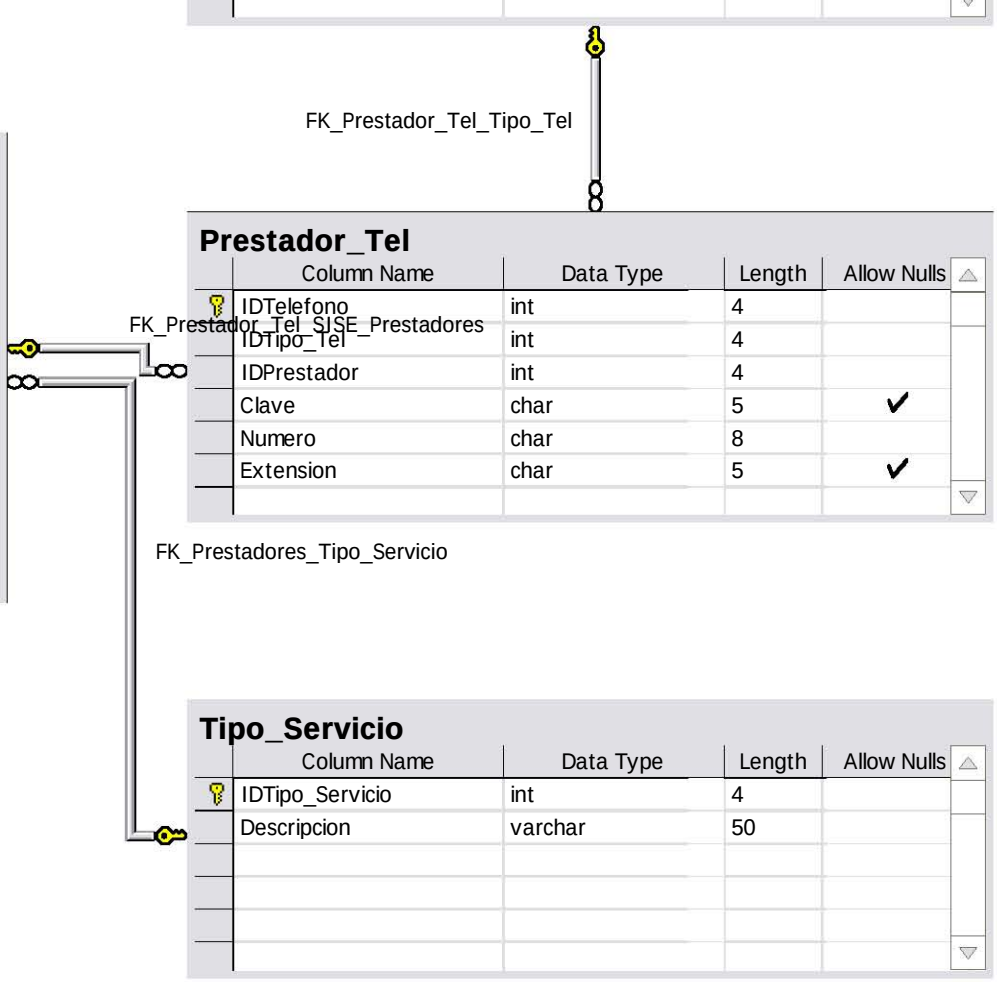
mn Name	Data Type	Length	Allow Nulls
	int	4	
	varchar	20	

FK_SISE_Prestadores_Edo_Civil

Prestadores

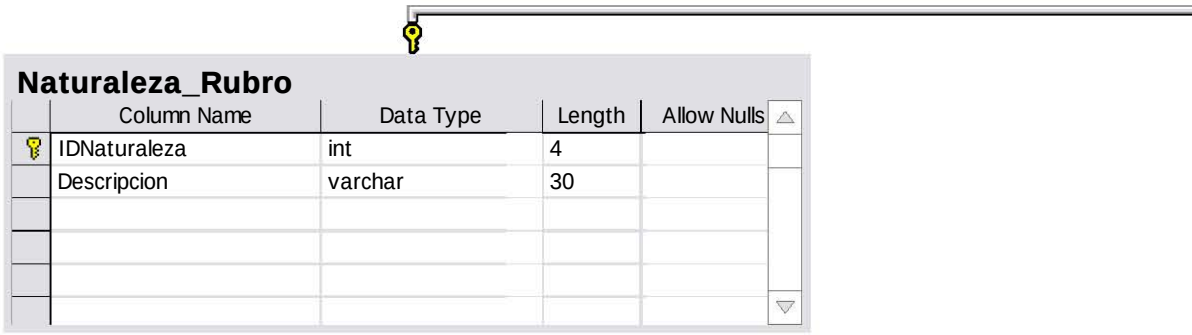
	Column Name	Data Type	Length	Allow Nulls
🔑	IDPrestador	int	4	
	Nombre	varchar	50	
	Paterno	varchar	20	✔
	Materno	varchar	20	✔
	Fisica_Moral	bit	1	
	RFC	char	14	
	CURP	char	18	✔
	Sexo	bit	1	✔
	Fecha_Nac	datetime	8	✔
	IDEdo_Civil	int	4	✔
	IDTipo_Servicio	int	4	

FK_Orden_Pago_Cuenta_Orden_Pago


FK_Prestador_Tel_Tipo_Tel
Prestador_Tel
Column Name
Data Type
Length
Allow Nulls
IDTelefono
int
4
IDTipo_Tel
int
4
IDPrestador
int
4
Clave
char
5
Numero
char
8
Extension
char
5
FK_Prestador_Tel_SJSE_Prestadores
FK_Prestadores_Tipo_Servicio
Tipo_Servicio
Column Name
Data Type
Length
Allow Nulls
IDTipo_Servicio
int
4
Descripcion
varchar
50

Bitacora

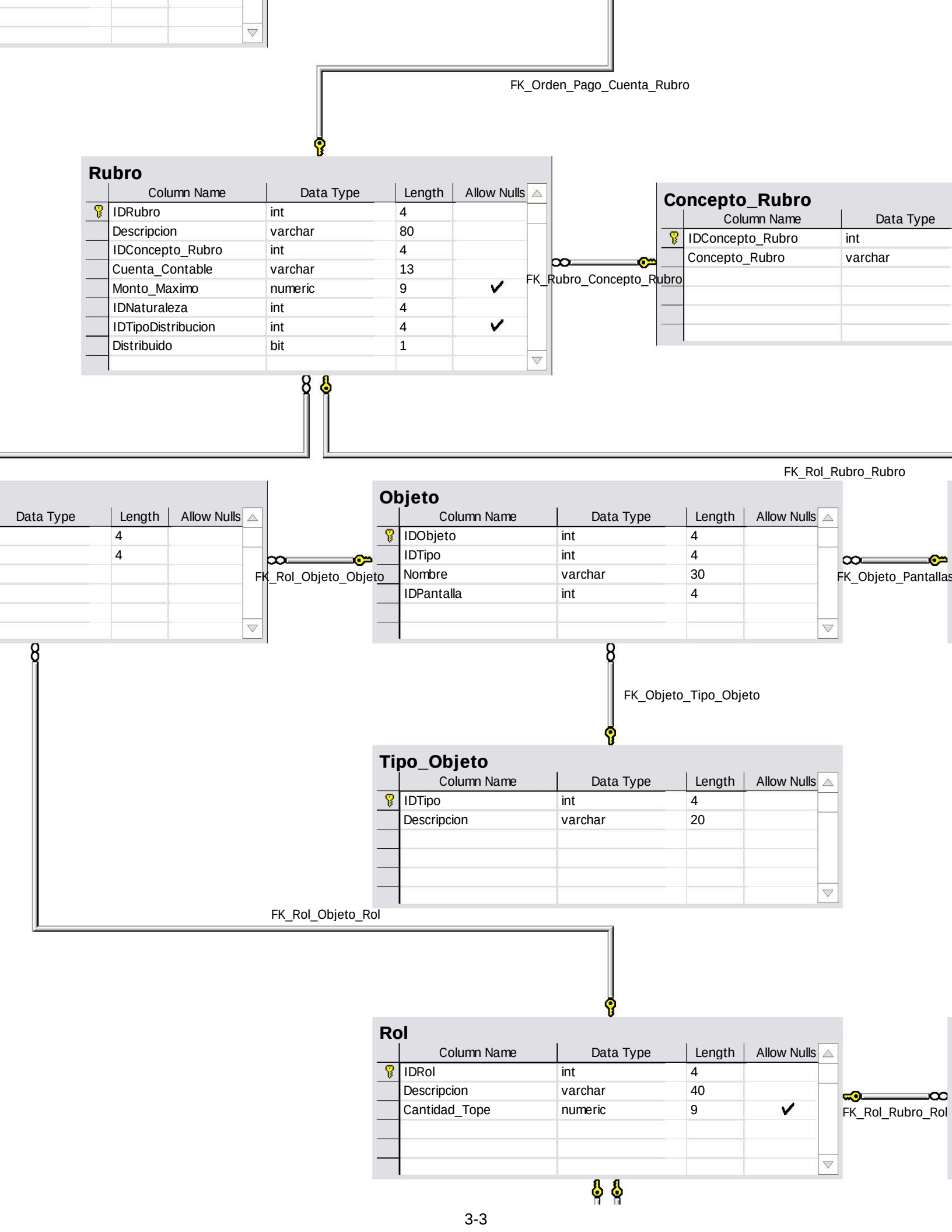
Column Name	Data Type	Length	Allow Nulls
IDBitacora	bigint	8	
IDMovimiento	int	4	
Tabla	varchar	40	
IDEmpleado	int	4	
Fecha	datetime	8	


Naturaleza_Rubro
Column Name
Data Type
Length
Allow Nulls
IDNaturaleza
int
4
Descripcion
varchar
30

FK_Rubro_Naturaleza_Rubro

Rol_Objeto

	Column Name	
🔑	IDObjeto	int
🔑	IDRol	int


FK_Orden_Pago_Cuenta_Rubro
Rubro
Column Name
Data Type
Length
Allow Nulls
IDRubro
int
4
Descripcion
varchar
80
IDConcepto_Rubro
int
4
Cuenta_Contable
varchar
13
Monto_Maximo
numeric
9
IDNaturaleza
int
4
IDTipoDistribucion
int
4
Distribuido
bit
1
FK_Rubro_Concepto_Rubro
Concepto_Rubro
Column Name
Data Type
IDConcepto_Rubro
int
Concepto_Rubro
varchar
FK_Rol_Rubro_Rubro
Data Type
Length
Allow Nulls
4
4
FK_Rol_Objeto_Objeto
Objeto
Column Name
Data Type
Length
Allow Nulls
IDObjeto
int
4
IDTipo
int
4
Nombre
varchar
30
IDPantalla
int
4
FK_Objeto_Pantallas
FK_Objeto_Tipo_Objeto
Tipo_Objeto
Column Name
Data Type
Length
Allow Nulls
IDTipo
int
4
Descripcion
varchar
20
FK_Rol_Objeto_Rol
Rol
Column Name
Data Type
Length
Allow Nulls
IDRol
int
4
Descripcion
varchar
40
Cantidad_Tope
numeric
9
FK_Rol_Rubro_Rol

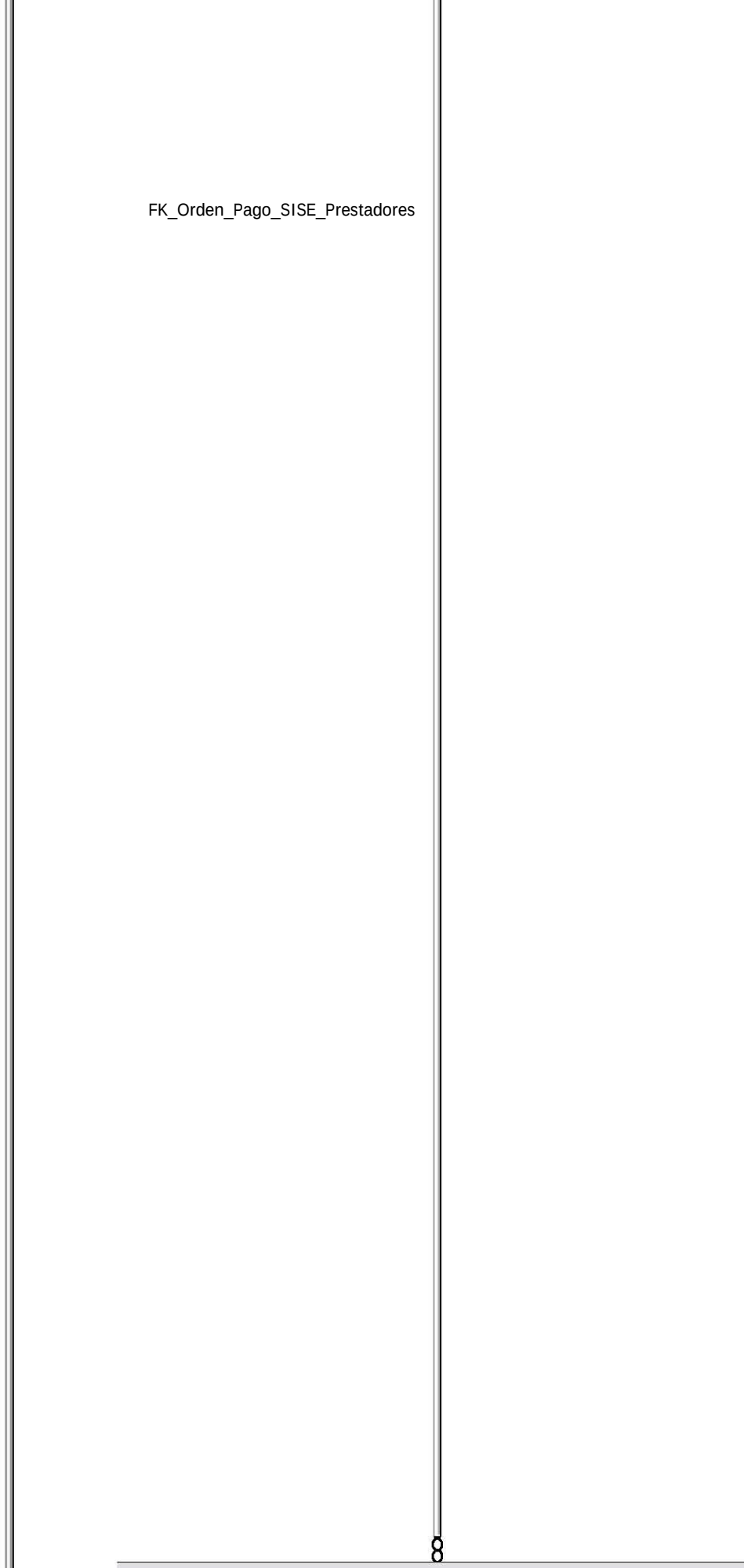
	Length	Allow Nulls
	4	
	50	

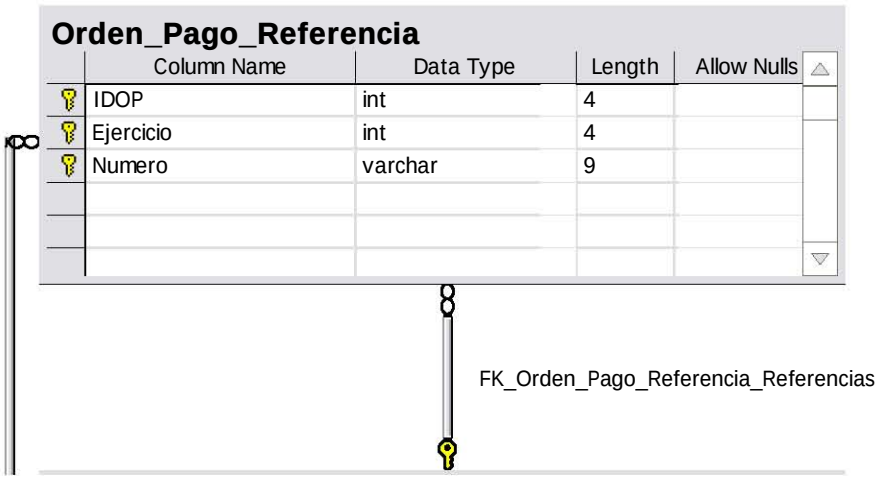
Pantallas

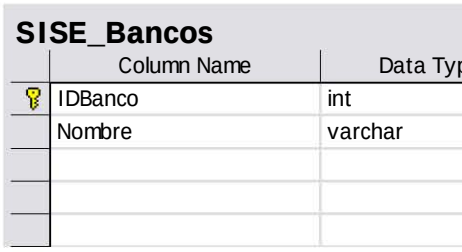
	Column Name	Data Type	Length	Allow Nulls
🔑	IDPantalla	int	4	
	Nombre	varchar	50	

Rol_Rubro

	Column Name	Data Type	Length	Allow Nulls
🔑	IDRol	int	4	
🔑	IDRubro	int	4	


FK_Orden_Pago_SISE_Prestadores
Orden_Pago
Column Name
Data Type
Length
Allow Nulls


Orden_Pago_Referencia
Column Name
Data Type
Length
Allow Nulls
IDOP
int
4
Ejercicio
int
4
Numero
varchar
9
FK_Orden_Pago_Referencia_Referencias


SISE_Bancos
Column Name
Data Typ
IDBanco
int
Nombre
varchar

pe	Length	Allow Nulls
	4	
	30	


Rol_Oficina
Column N
IDOficina
IDRol

Name	Data Type	Length	Allow Nulls
	int	4	
	int	4	

Procesos

	Column Name	Data Type	Length	Allow Nulls
🔑	IDProceso	int	4	
🔑	IDTabla	int	4	
🔑	IDCriterio	int	4	

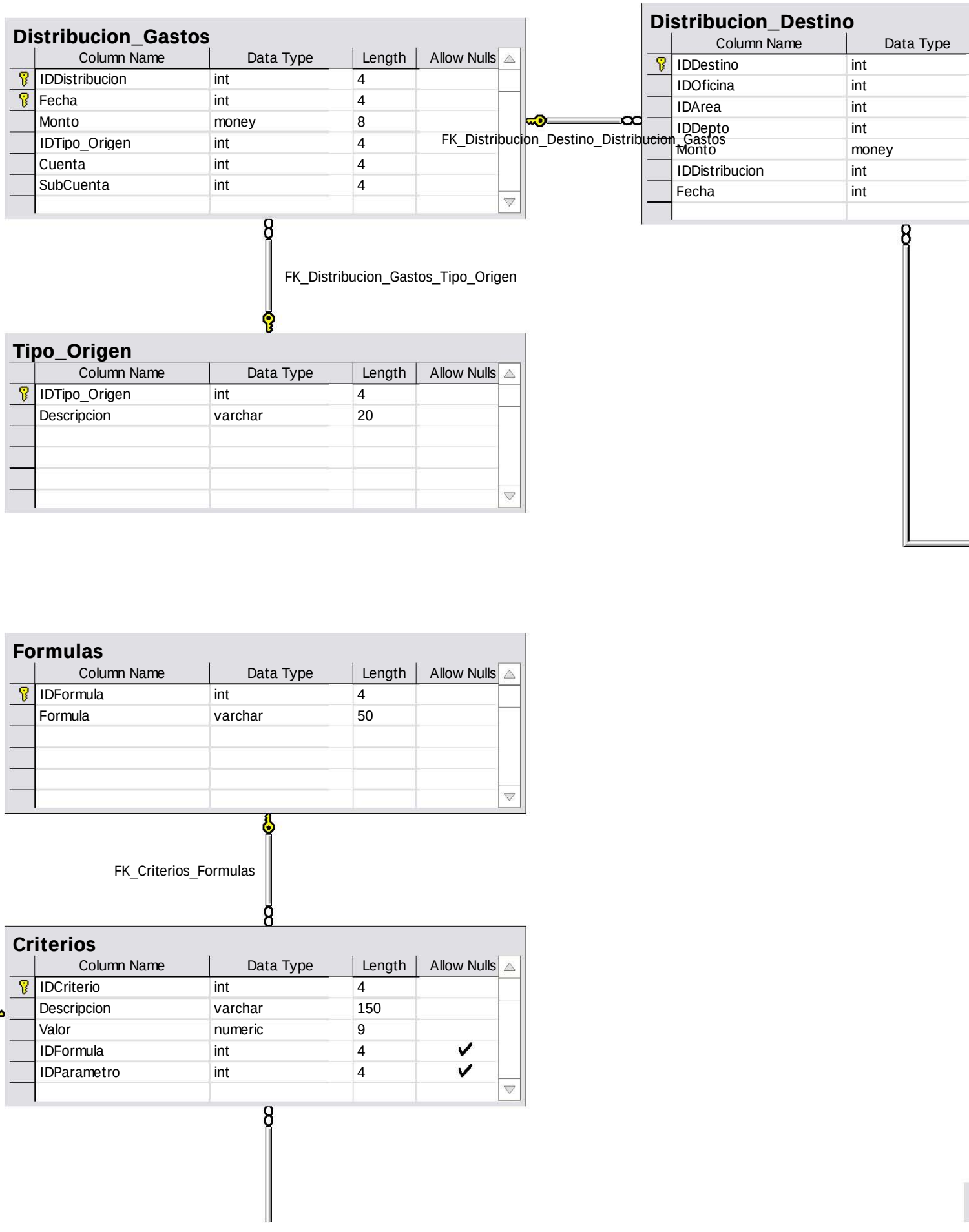
FK_Procesos_

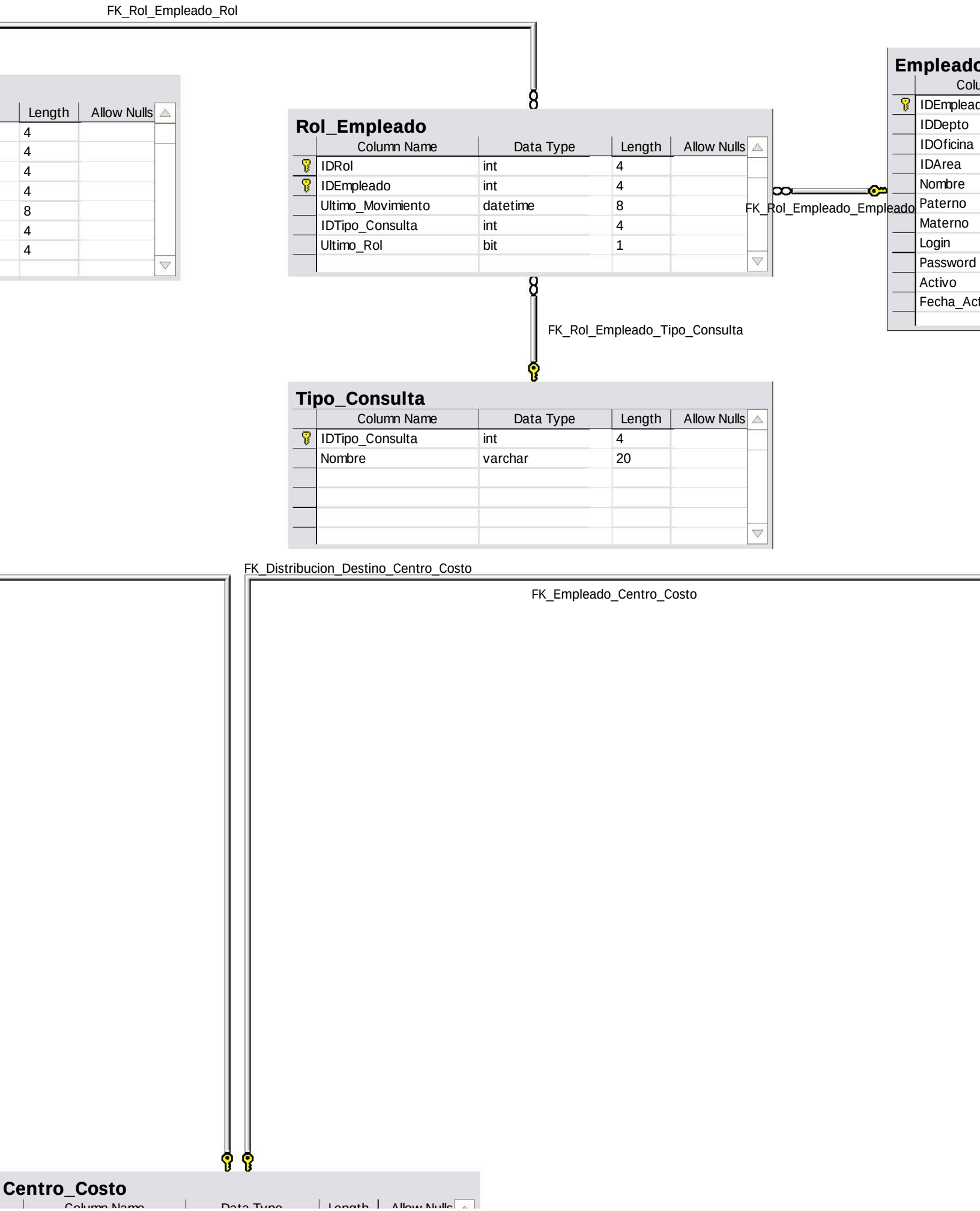
FK_Distribucion_Procesos

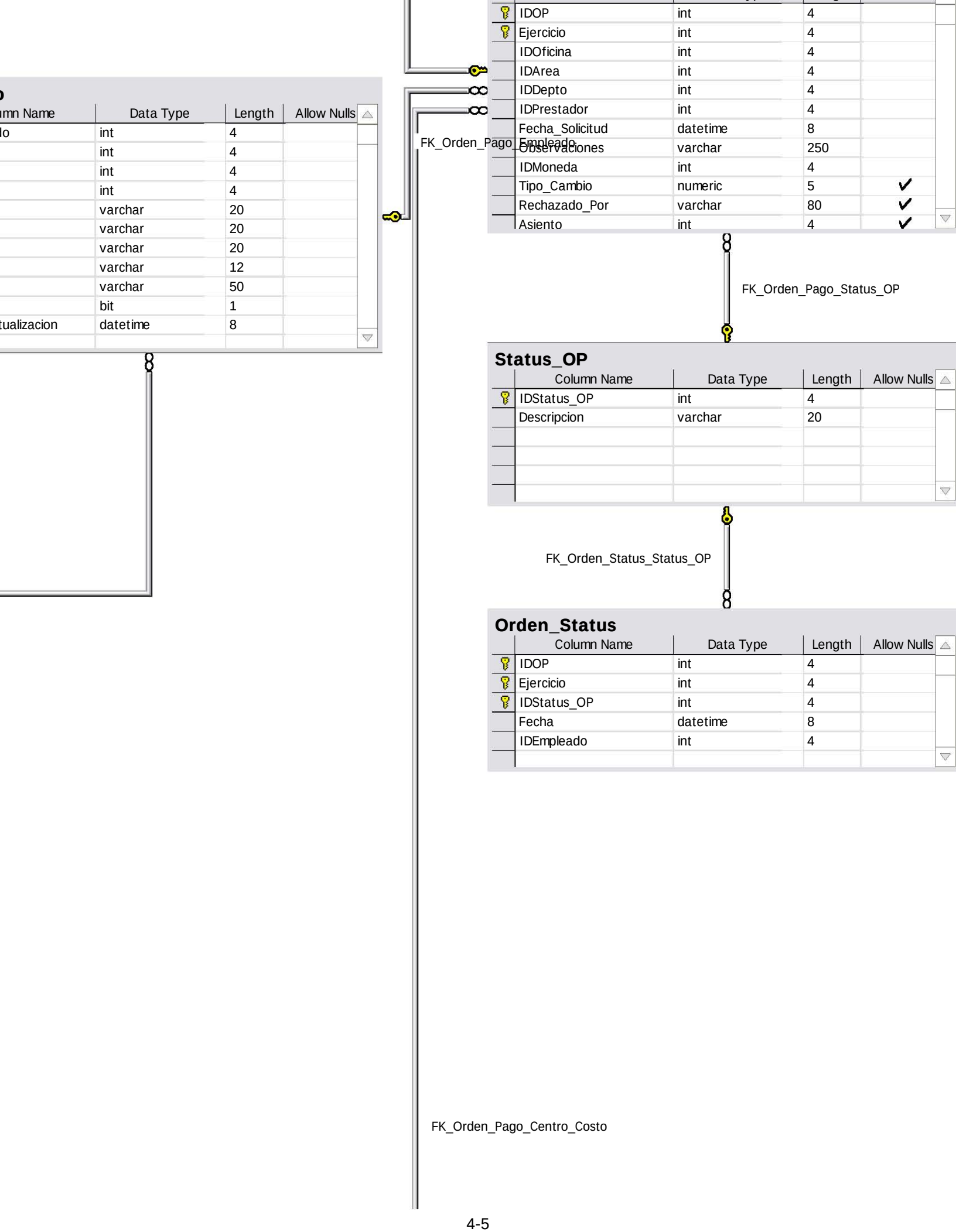
Distribucion

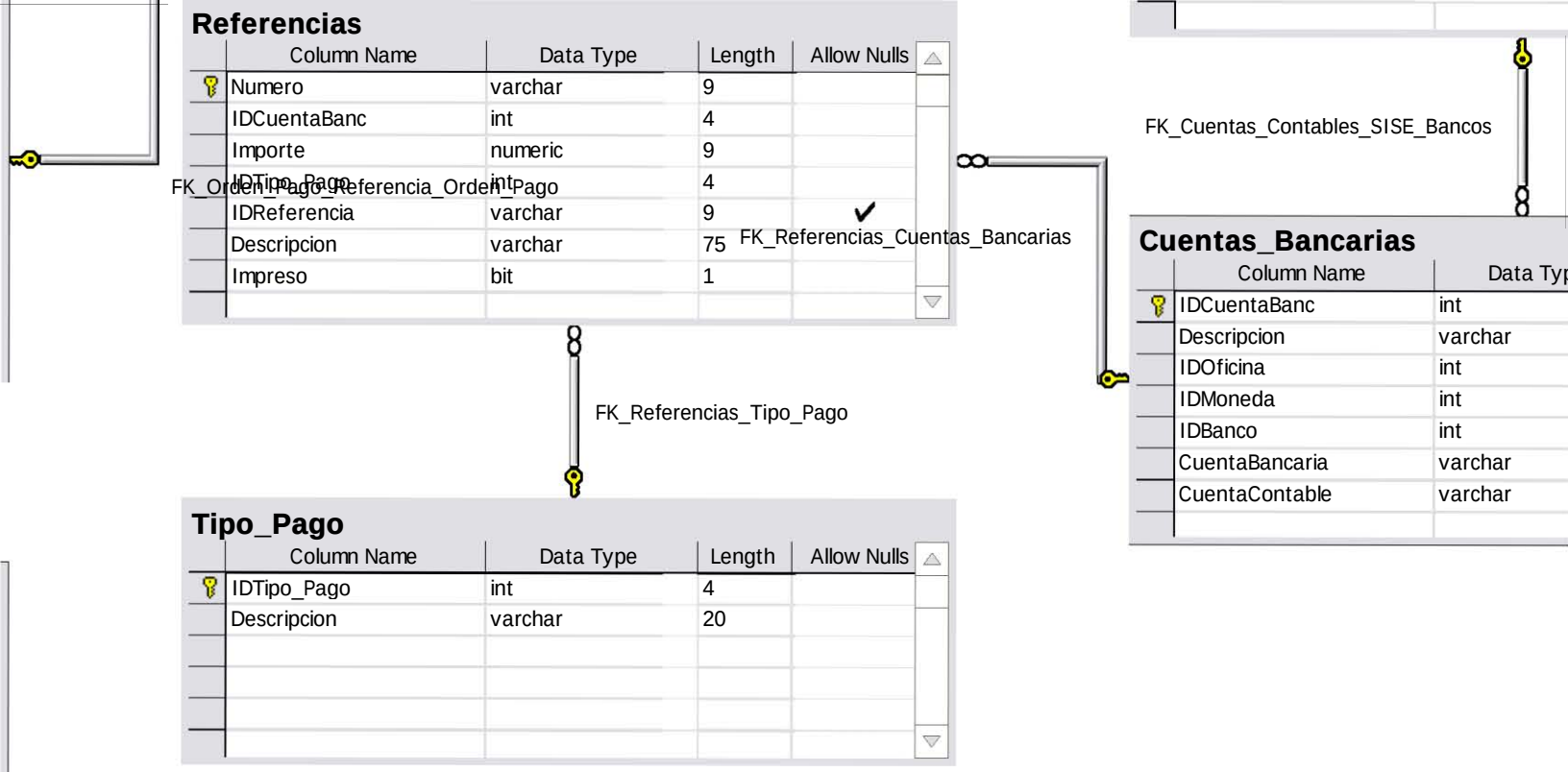
	Column Name	Data Type	Length	Allow Nulls
🔑	IDDistribucion	int	4	
	IDOficina	int	4	
	IDArea	int	4	
	IDDepto	int	4	
	IDProceso	int	4	

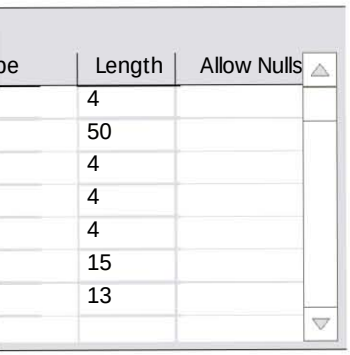
FK_Rol_Oficina_Rol




FK_Rol_Empleado_Rol
Length
Allow Nulls
4
4
4
4
8
4
4
Rol_Empleado
Column Name
Data Type
Length
Allow Nulls
IDRol
int
4
IDEmpleado
int
4
Ultimo_Movimiento
datetime
8
IDTipo_Consulta
int
4
Ultimo_Rol
bit
1
FK_Rol_Empleado_Empleado
Empleado
Colu
IDEmplead
IDDepto
IDOficina
IDArea
Nombre
Paterno
Materno
Login
Password
Activo
Fecha_Act
FK_Rol_Empleado_Tipo_Consulta
Tipo_Consulta
Column Name
Data Type
Length
Allow Nulls
IDTipo_Consulta
int
4
Nombre
varchar
20
FK_Distribucion_Destino_Centro_Costo
FK_Empleado_Centro_Costo
Centro_Costo


o
umn Name
Data Type
Length
Allow Nulls
lo
int
4
int
4
int
4
int
4
varchar
20
varchar
20
varchar
20
varchar
12
varchar
50
bit
1
tualizacion
datetime
8
IDOP
int
4
Ejercicio
int
4
IDOficina
int
4
IDArea
int
4
IDDepto
int
4
IDPrestador
int
4
Fecha_Solicitud
datetime
8
Observaciones
varchar
250
IDMoneda
int
4
Tipo_Cambio
numeric
5
Rechazado_Por
varchar
80
Asiento
int
4
FK_Orden_Pago_Empleado
FK_Orden_Pago_Status_OP
Status_OP
Column Name
Data Type
Length
Allow Nulls
IDStatus_OP
int
4
Descripcion
varchar
20
FK_Orden_Status_Status_OP
Orden_Status
Column Name
Data Type
Length
Allow Nulls
IDOP
int
4
Ejercicio
int
4
IDStatus_OP
int
4
Fecha
datetime
8
IDEmpleado
int
4
FK_Orden_Pago_Centro_Costo


Referencias
Column Name
Data Type
Length
Allow Nulls
Numero
varchar
9
IDCuentaBanc
int
4
Importe
numeric
9
IDTipo_Pago
int
4
IDReferencia
varchar
9
Descripcion
varchar
75
Impreso
bit
1
FK_Orden_Pago_Referencia_Orden_Pago
FK_Referencias_Cuentas_Bancarias
FK_Referencias_Tipo_Pago
FK_Cuentas_Contables_SISE_Bancos
Cuentas_Bancarias
Column Name
Data Typ
IDCuentaBanc
int
Descripcion
varchar
IDOficina
int
IDMoneda
int
IDBanco
int
CuentaBancaria
varchar
CuentaContable
varchar
Tipo_Pago
Column Name
Data Type
Length
Allow Nulls
IDTipo_Pago
int
4
Descripcion
varchar
20


pe
Length
Allow Nulls
4
50
4
4
4
15
13

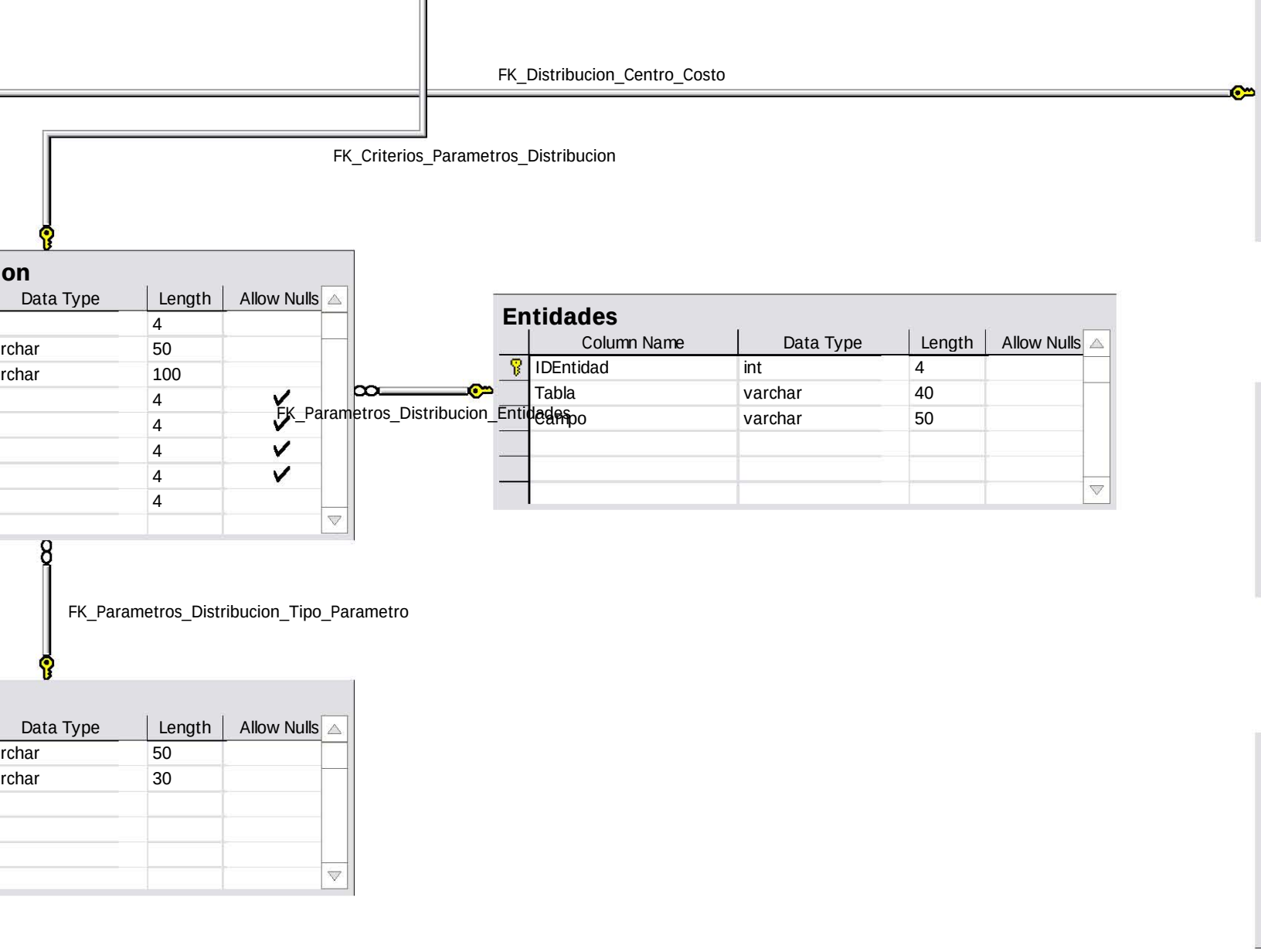
	IDTabla	int	4	
	IDCriterio	int	4	
	IDTipo_Distribucion	int	4	

Parametros_Distribuci

	Column Name	
🔑	IDParametro	int
	IDTipo_Parametro	va
	Descripcion	va
	IDRamo_D	int
	IDOficina_D	int
	IDCuenta	int
	Valor	int
	IDEntidad	int

Tipo_Parametro

	Column Name	
🔑	IDTipo_Parametro	va
	Descripcion	va


FK_Distribucion_Centro_Costo
FK_Criterios_Parametros_Distribucion
on
Data Type
Length
Allow Nulls
4
rchar
50
rchar
100
4
4
4
4
4
FK_Parametros_Distribucion_Entidades
Entidades
Column Name
Data Type
Length
Allow Nulls
IDEntidad
int
4
Tabla
varchar
40
Campo
varchar
50
FK_Parametros_Distribucion_Tipo_Parametro
Data Type
Length
Allow Nulls
rchar
50
rchar
30

Column Name	Data Type	Length	Allow Nulls
IDOficina	int	4	
IDArea	int	4	
IDDepto	int	4	
Nombre_Departamento	varchar	50	
Activo	bit	1	
Tipo_Distribucion	bit	1	
Metros_Cuadrados	float	8	✔
Empleados	int	4	✔

FK_Centro_Costo_Oficina_Areas

Oficina_Areas

Column Name	Data Type	Length	Allow Nulls
IDOficina	int	4	
IDArea	int	4	

FK_Oficina_Areas_Areas

Areas

Column Name	Data Type	Length	Allow Nulls
IDArea	int	4	
Nombre	varchar	40	

CLASE SEGURIDAD

Descripción: Clase utilizada para la validación de Claves de acceso de Usuarios

Public Function blnCambiaPassword(ByVal plngID As Long, ByVal pstrPassword As String, ByVal pstrViejo As String) As Boolean
Descripción: Función utilizada para modificar la clave de acceso del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnCambiaPassword"

    Dim strSql As String
    Dim objBD As SIFIBD.BDAcceso
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strPassword As String

On Error GoTo errHandler

        Set objEncripta = New xmsEncriptador.Encriptador
        'encripto el nuevo password
        strPassword = objEncripta.EncriptaString(pstrPassword)

        'Formo el query
        strSql = "UPDATE " & GSTR_TBL_EMPLEADO & vbCr
        strSql = strSql & " SET " & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" & Trim$(strPassword) &
"'" & vbCr
        strSql = strSql & " ," & GSTR_TBL_EMPLEADO_FLD_STATUS_ACCESO & "=1" & vbCr
        strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngID

        Set objBD = New SIFIBD.BDAcceso

        'Ejecuto el query
        blnCambiaPassword = objBD.blnEjecutaQuery(strSql)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnCambiaPassword = False
    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set objEncripta = Nothing

End Function
```

Public Property Let Empleado(ByVal plngEmpleado As Long)

```
    mlngEmpleado = plngEmpleado
End Property
```

Public Function lngNuevoRol(ByVal pstrNombre As String) As Long
Descripción: Función para agregar una nueva clave de Rol al usuario

```
Dim objBD          As SIFIBD.BDAcceso
```

```
Dim strSql          As String
Dim lngIdRol        As Long

On Error GoTo ErrorHandler

    If Trim$(pstrNombre) <> vbNullString Then

        lngIdRol = objBD.lngGetNewId("Rol", "IdRol")

        If lngIdRol > 0 Then

            Set objBD = New SIFIBD.BDAcceso

            strSql = "Insert Into Rol (Nombre, IdRol) "
            strSql = strSql & "Values ('" & pstrNombre & "', " & _
                                    lngIdRol & ")"


            Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ALTA)


            objBD.blnEjecutaQuery strSql

            Set objBD = Nothing

            lngNuevoRol = lngIdRol
        Else
'            mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"
        End If
    Else
'       mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"
    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

            Set objBD = Nothing

            lngNuevoRol = -1

'            mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"

    End Select
End Function
```

Public Function blnAsignaRolObjeto(ByVal plngObjeto As Long, ByVal plngRol As Long) As Boolean

Descripción: Función utilizada para dar permisos a un objeto de la pantalla a un rol del usuario

```
    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo ErrorHandler

    blnAsignaRolObjeto = False
```

```
    Set objBD = New SIFIBD.BDAcceso


'    strSql = "UPDATE " &  Objeto set IdRol=" & plngRol
'    strSql = strSql & "Where idObjeto =" & plngObjeto

    objBD.blnEjecutaQuery strSql

    Set objBD = Nothing

    blnAsignaRolObjeto = True


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

            Set objBD = Nothing

            blnAsignaRolObjeto = False

            mobjErr.RegistraError Err.Number, Err.Description, "SIFISeguridad.blnAsignaRolObjeto"

    End Select

End Function
```

Public Function lngValidaExistenciaEmpleado(ByVal pstrLogin As String, ByVal pstrPassword As String) As Long

Descripción: Función para validar los datos del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".lngValidaExistenciaEmpleado"
    Const LNG_NO_EXISTE_EMPLEADO As Long = -1
    Const LNG_EMPLEADO_INACTIVO As Long = -2
    Const LNG_PASSWORD_INCORRECTO As Long = -3

    Dim strSql As String

    Dim objBD As SIFIBD.BDAcceso
    Dim rsdEmpleado As ADODB.Recordset
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strPassword As String

On Error GoTo errHandler

    strSql = "SELECT * FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_LOGIN & "='" & LCase$(pstrLogin) & "'" &
vbCr
    'strSQL = strSQL & " AND " & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" &
LCase$(pstrPassword) & "'" & vbCr

    Set objBD = New SIFIBD.BDAcceso


    Set rsdEmpleado = objBD.rsdDevuelveRecordset(strSql)
```

```
    If Not (rsdEmpleado.BOF And rsdEmpleado.EOF) Then
        Set objEncripta = New xmsEncriptador.Encriptador
        strPassword =
objEncripta.DesencriptaString(rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_PASSWORD))

        'Verifico que el password sea correcto
        If UCase$(strPassword) = UCase$(pstrPassword) Then
            lngValidaExistenciaEmpleado = rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_ID)
        Else
            lngValidaExistenciaEmpleado = LNG_PASSWORD_INCORRECTO
        End If

        'Verifico si el empelado está activo o no para el sistema
        If Not rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_ACTIVO) Then

            lngValidaExistenciaEmpleado = LNG_EMPLEADO_INACTIVO
        End If


    Else
        'El empleado no existe
        lngValidaExistenciaEmpleado = LNG_NO_EXISTE_EMPLEADO
    End If

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            lngValidaExistenciaEmpleado = LNG_NO_EXISTE_EMPLEADO
    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set rsdEmpleado = Nothing
    Set objEncripta = Nothing

End Function
```

Public Function rsdObtenPermisosUsuario(plngUsuario As Long) As ADODB.Recordset

Descripción: Función para obtener los permisos de un Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdObtenPermisosUsuario"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo ErrorHandler

    strSql = "SELECT " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE & " AS " &
            GSTR_TBL_OBJETO & ", " & vbCr
    strSql = strSql & GSTR_TBL_PANTALLA & "." & GSTR_TBL_PANTALLA_FLD_NOMBRE & " AS " &
            GSTR_TBL_PANTALLA & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_OBJETO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
```

```
    strSql = strSql & " = " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_OBJETO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_PANTALLA & vbCr
    strSql = strSql & " ON " & GSTR_TBL_OBJETO & "." & GSTR_TBL_PANTALLA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_PANTALLA & "." & GSTR_TBL_PANTALLA_FLD_ID
    strSql = strSql & " WHERE " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
          & "=" & plngUsuario
    strSql = strSql & " AND " & GSTR_TBL_ROL_EMPLEADO & "." &
          GSTR_TBL_ROL_EMPLEADO_ULTIMO_ROL & "=1"

    Set objBD = New SIFIBD.BDAcceso
    Set rsdObtenPermisosUsuario = objBD.rsdDevuelveRecordset(strSql)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else



          Set rsdObtenPermisosUsuario = Nothing

          mobjErr.RegistraError Err.Number, Err.Description, "SIFISeguridad.blnAsignaRolUsuario"

    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function


'
Public Function blnActualizaRol(ByVal pstrDescripcion As String, ByVal pdblMontoMaximo As Double,
          ByVal pblnInserta As Boolean, Optional ByVal plngIDRol As Long) As Boolean
'Descripcion:       Funcion para actualizar un rol

    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaRol"

          Dim strSql            As String
          Dim objBD             As SIFIBD.BDAcceso
          Dim lngIdRol     As Long

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaRol = False

    If pblnInserta = True Then
       'verifica que no exista el nombre del Rol

       If objBD.blnExisteDatoenTabla(GSTR_TBL_ROL_FLD_DESCRIPCION, GSTR_TBL_ROL,
                pstrDescripcion) = False Then

          lngIdRol = objBD.lngGetNewId(GSTR_TBL_ROL, GSTR_TBL_ROL_FLD_ID)
```

```
            strSql = "Insert into " & GSTR_TBL_ROL & " (" & GSTR_TBL_ROL_FLD_ID & ", " & _
                                                GSTR_TBL_ROL_FLD_CANTIDAD & ", " & _
                                                GSTR_TBL_ROL_FLD_DESCRIPCION & ") values(" & _
                                                lngIdRol & ", " & _
                                                pdblMontoMaximo & ", '" & _
                                                pstrDescripcion & "')"

            Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ALTA)

        End If
    Else

        strSql = "delete from " & GSTR_TBL_ROL & " where " & GSTR_TBL_ROL_FLD_ID & "=" & plngIDRol

        Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

    End If

        'verifica que exista una sentencia a ejecutar
    If Len(strSql) > 0 Then
        strSql = UCase(strSql)

        blnActualizaRol = objBD.blnEjecutaQuery(strSql)

    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnActualizaRol = False

            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

CLASE ORDEN DE PAGO

Public Function blnActualizaLoteTransf(ByVal plngOP As Long, ByVal pstrNumero As String, ByVal plngNoLote As Long, ByVal pstrIDCuentaBancaria As String, ByVal pintIDOperacion As String, ByVal plngSucursal As Long) As Boolean

'**Descripción:** Se actualiza en cada una de las OP, transferencias, del archivo (lote)
' en el que se está enviando al banco (liberacion).

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaLoteTransf"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo errHandler


    strSQL = "UPDATE " & GSTR_TBL_REF_TRAN_AUT
    strSQL = strSQL & " SET " & GSTR_TBL_REF_TRAN_AUT_FLD_LOTE & " = " & plngNoLote & ", "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA & " = " &
        pstrIDCuentaBancaria & ", "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_OPERA & " = " & pintIDOperacion & ", " &
        strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL & " = " & plngSucursal
    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & " = " & plngOP
    strSQL = strSQL & " AND " & GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON & " = " & pstrNumero &
        vbCr

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaLoteTransf = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

       Case 0
          'Do nothing
       Case Else
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
          blnActualizaLoteTransf = False
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnDesActualizaLoteTransf(ByVal plngOP As Long, ByVal pstrNumero As String) As Boolean

'Descripción: Se inicializa los valores de las transferencias a Nulos para que puedan ser
'liberadas nuevamente

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnDesActualizaLoteTransf"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso
```

```
    On Error GoTo errHandler


    strSQL = "UPDATE " & GSTR_TBL_REF_TRAN_AUT
    strSQL = strSQL & " SET " & GSTR_TBL_REF_TRAN_AUT_FLD_LOTE & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_OPERA & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL & " = NULL" & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & " = " & plngOP
   ' strSQL = strSQL & " AND " & GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON & " = " & pstrNumero &
            vbCr

    Set objBD = New SIFIBD.BDAcceso

    blnDesActualizaLoteTransf = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnDesActualizaLoteTransf = False
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnActualizaOrdenPago(ByVal pstrNumero As String, ByVal pdblImporte As Double, ByVal plngTipoPago As Long, ByVal plngCuentaBancaria As Long, ByVal pstrDescripcion As String, ByVal pstrNumeroAnterior As String) As Boolean

Descripción: Función para actualizar la información de una Orden de Pago.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaOrdenPago"

    Dim strSQL As String
    Dim strSQLImputacion As String
    Dim objBD As SIFIBD.BDAcceso


On Error GoTo errHandler

    strSQL = "INSERT INTO " & GSTR_TBL_REFERENCIAS & " (" & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPRESO & "," & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_ID & ")" & vbCr

    strSQL = strSQL & " Values (" & vbCr
    strSQL = strSQL & "'" & Trim$(pstrNumero) & "', " & vbCr
```

```
    strSQL = strSQL & plngCuentaBancaria & ", " & vbCr
    strSQL = strSQL & plngTipoPago & ", " & vbCr
    strSQL = strSQL & pdblImporte & "," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrDescripcion)) & "', " & vbCr
    strSQL = strSQL & "0," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrNumeroAnterior)) & "')"

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaOrdenPago = objBD.blnEjecutaQuery(strSQL)



errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            'mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnActualizaOrdenSISE(ByVal plngOrden As Long, ByVal plngEjercicio As Long, ByVal plngCaratula As Long, ByVal plngAsiento As Long, ByVal plngMovimiento As Long) As Boolean

Descripción: Actualizar la información de una orden importada de los sistemas centrales.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaOrdenSISE"
    Const LNG_VALOR_SISE As Long = 1

    Dim strSQL          As String
    Dim objBD           As SIFIBD.BDAcceso
    Dim strTransaccion  As String
    Dim strSQLOP        As String
  '  Dim strSQLMov       As String

On Error GoTo errHandler

    strSQL = strSQL & " BEGIN TRAN OPERA " & vbCr
    strSQL = strSQL & " INSERT INTO " & GSTR_TBL_ORDEN_SISE
    strSQL = strSQL & " (" & GSTR_TBL_ORDEN_SISE_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_SISE_FLD_CARATULA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_SISE_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_MOVIMIENTOS_SISE_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES ( " & plngAsiento
    strSQL = strSQL & ", " & plngCaratula
    strSQL = strSQL & ", '" & Format(GetDateFromServer, "YYYY-MM-DD") & "'"
    strSQL = strSQL & ", " & plngOrden
    strSQL = strSQL & ", " & plngEjercicio
    strSQL = strSQL & ", " & plngMovimiento & ");" & vbCr
    strSQL = strSQL & "UPDATE " & GSTR_TBL_ORDEN_PAGO
    strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" & plngAsiento & ", "
```

```
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula
    strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" & plngOrden
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & "=" & plngEjercicio & ";" &
          vbCr
    strSQL = strSQL & " IF @@ERROR = 0 COMMIT TRAN OPERA ELSE ROLLBACK TRAN OPERA " &
          vbCr

    Set objBD = New SIFIBD.BDAcceso
    blnActualizaOrdenSISE = objBD.blnEjecutaQuery(strSQL)
    Set objBD = Nothing

errHandler:
    Select Case Err.Number

       Case 0
          'Do nothing
       Case Else
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
          blnActualizaOrdenSISE = False
          Call objBD.blnRollBack(strTransaccion)
    End Select
'</CSCustomCode> 1

    'Libero la memoria
    Set objBD = Nothing
End Function
```

Public Function blnActualizaStatus(ByVal plngOrden As Long, ByVal plngEjercicio As Long, ByVal plngStatus As Long, ByVal plngIDEmpleado As Long, Optional ByVal blnEliminar As Boolean) As Boolean

Descripción: Actualiza el estatus de una orden cuando se autoriza o cancela

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaStatus"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String
    Dim strSQLOPStatus As String

On Error GoTo errHandler

    'agrego un registro a la tabla de status de la OP
    strSQL = "INSERT INTO " & GSTR_TBL_ORDEN_STATUS & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", "
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_STATUS_OP_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_ORDEN_STATUS_FLD_FECHA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EMPLEADO_FLD_ID & ") " & vbCr
    strSQL = strSQL & " VALUES " & vbCr
    strSQL = strSQL & "(" & plngEjercicio & ", "
    strSQL = strSQL & plngOrden & ", "
    strSQL = strSQL & plngStatus & ", "
    strSQL = strSQL & "'" & Format(Now, "YYYY-MM-DD h:m:s") & "', " & vbCr
    strSQL = strSQL & plngIDEmpleado & ") " & vbCr


    Set objBD = New SIFIBD.BDAcceso

    If Not blnEliminar Then
```

```
        'Actualizo el último status de la OP
        strSQLOPStatus = "UPDATE " & GSTR_TBL_ORDEN_PAGO & vbCr
        strSQLOPStatus = strSQLOPStatus & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
                         & "=" & plngStatus & vbCr
        strSQLOPStatus = strSQLOPStatus & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" &
                         plngOrden & vbCr
        strSQLOPStatus = strSQLOPStatus & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & "="
                         & plngEjercicio


        If objBD.blnEjecutaQuery(strSQL) Then
           blnActualizaStatus = objBD.blnEjecutaQuery(strSQLOPStatus)
        Else
           blnActualizaStatus = False
        End If

     Else
        blnActualizaStatus = objBD.blnEjecutaQuery(strSQL)

     End If

errHandler:
     Select Case Err.Number

        Case 0
           'Do nothing
        Case Else
           mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
           blnActualizaStatus = False

     End Select
'</CSCustomCode> 1

     'Libero la memoria
     Set objBD = Nothing

End Function
```

Public Function blnGuardarOrdenPagoLIQ(ByVal plngOP As Long, ByVal plngEjercicio As Long, ByVal plngLiq As Long)

Descripción: Se guarda el Recibo asignado a la OP (Devolución de Primas)

```
     Const STR_FUNCION As String = MSTR_MODULO & ".blnGuardarOrdenPagoLIQ"
     Dim strSQL        As String
     Dim objBD         As SIFIBD.BDAcceso

     On Error GoTo ErrorHandler

     Set objBD = New SIFIBD.BDAcceso

     strSQL = "Insert Into " & GSTR_TBL_ORDEN_PAGO_LIQ & " (" & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_IDOP & ", " & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_EJERCICIO & ", " & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_IDLIQ & ")"

     strSQL = strSQL & " Values (" & plngOP & ", " & _
                       plngEjercicio & ", " & _
                       plngLiq & ")"
```

```
    blnGuardarOrdenPagoLIQ = objBD.blnEjecutaQuery(strSQL)


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnGuardarOrdenPagoLIQ = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
End Function
```

Private Function lngObtenIDOP()

Descripción: Función utilizada para obtner un nuevo número de Orden de Pago

```
    Const STR_FUNCION As String = MSTR_MODULO & ".lngObtenIDOP"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String
    Dim rsdOP As ADODB.Recordset

On Error GoTo errHandler

    strSQL = "SELECT MAX(" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ") AS " &
GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", " & vbCr
    strSQL = strSQL & "getdate()" & vbCr
    strSQL = strSQL & " FROM " & GSTR_TBL_ORDEN_PAGO & vbCr


    Set objBD = New SIFIBD.BDAcceso

    Set rsdOP = objBD.rsdDevuelveRecordset(strSQL)

    If Not (rsdOP.EOF And rsdOP.BOF) Then

        If rsdOP(GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO) <> Year(rsdOP(1)) Then
            lngObtenIDOP = 1
        End If

    End If

    If lngObtenIDOP <> 1 Then

        lngObtenIDOP = objBD.lngGetNewId(GSTR_TBL_ORDEN_STATUS,
                                GSTR_TBL_ORDEN_PAGO_FLD_ID)
    End If

errHandler:
    Select Case Err.Number

        Case 0
```

```
        'Do nothing
      Case Else
        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select
'</CSCustomCode> 1

   'Libero la memoria
   Set objBD = Nothing
   Set rsdOP = Nothing

End Function


Public Function NuevaOrden(ByVal plngEjercicio As Long, ByVal plngOficina As Long, ByVal plngArea As
        Long, ByVal plngDepto As Long, ByVal plngBeneficiario As Long, ByVal plngRubro As Long, ByVal
        pstrSolicita As String, ByVal datFechaSolicitud As Date, ByRef plngIDOP As Long, _
            ByVal pstrObservaciones As String, ByVal pstrLeyendaCheque As String, ByVal plngIdMoneda
        As Long, ByVal pdblTipoCambio As Double, ByVal pblnOP As Boolean, Optional ByVal
        plngTraspaso As Long, Optional ByVal plngEjercicioTraspaso As Long, _
            Optional ByVal plngIDBanco As Long, Optional ByVal plngIDPlaza As Long, Optional ByVal
        plngIDSucursal As Long, Optional ByVal pstrHora As String, Optional ByVal pintFormaPago As
        Integer, Optional ByVal pdblImporteTotal As Double = 0) As Boolean


'Descripción:       Funcion Crear una Nueva OP en la BD

   Const STR_FUNCION As String = MSTR_MODULO & ".NuevaOrden"
   Const INT_ID_STATUS_PENDIENTE As Integer = 1

Dim strSQL         As String
Dim objBD          As SIFIBD.BDAcceso
Dim rdsTransOP     As ADODB.Recordset
Dim intIdOp        As Integer
Dim strTansaccion  As String
Dim lngErrorSQL    As Long

On Error GoTo ErrorHandler

   Set objBD = New SIFIBD.BDAcceso

   NuevaOrden = False

   'intIdOp = lngObtenIDOP()
   'plngIDOP = intIdOp

   strSQL = "DECLARE @IDOP       INT " & vbCr
   strSQL = strSQL & "DECLARE @ERRORES    INT" & vbCr
   strSQL = strSQL & "SET NOCOUNT ON " & vbCr
   strSQL = strSQL & "SET @ERRORES = 0 " & vbCr & vbCr
   strSQL = strSQL & "BEGIN TRAN NuevaOrdenPago" & vbCr
   strSQL = strSQL & "SELECT @IDOP = ISNULL(MAX(IDOP),0) + 1 FROM ORDEN_STATUS" & vbCr

   'Orden_Pago
   strSQL = strSQL & "INSERT INTO " & GSTR_TBL_ORDEN_PAGO
   strSQL = strSQL & " ( " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
   strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr
   strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_ID
   strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_ID
   strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
   strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR_FLD_ID
```

```
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_LEYENDACHEQUE
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID & vbCr

strSQL = strSQL & ", " & GSTR_TBL_TRASPASOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_TRASPASOS_FLD_EJERCICIO
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP
'Nuevos campos
strSQL = strSQL & ", " & GSTR_TBL_BANCOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_PLAZA_FLD_ID_PLAZA
strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
strSQL = strSQL & ", " & "IdTipo_Pago"
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_REG_HOR
strSQL = strSQL & ", " & "Importe)" & vbCr

strSQL = strSQL & " Values (" & plngEjercicio
strSQL = strSQL & ", @IDOP "
strSQL = strSQL & ", " & plngOficina
strSQL = strSQL & ", " & plngArea
strSQL = strSQL & ", " & plngDepto & vbCr
strSQL = strSQL & ", " & plngBeneficiario
strSQL = strSQL & ", " & pstrSolicita
strSQL = strSQL & ", '" & Format(datFechaSolicitud, "yyyy-mm-dd Hh:Nn:Ss")
strSQL = strSQL & "', '" & pstrObservaciones
strSQL = strSQL & "', '" & pstrLeyendaCheque
strSQL = strSQL & "', " & pdblTipoCambio
strSQL = strSQL & ", " & INT_ID_STATUS_PENDIENTE
strSQL = strSQL & ", " & plngIdMoneda

'Verifico si se está mandando los datos del traspaso
If plngTraspaso > 0 Then
   strSQL = strSQL & ", " & plngTraspaso
   strSQL = strSQL & ", " & plngEjercicioTraspaso
Else
   strSQL = strSQL & ", NULL"
   strSQL = strSQL & ", NULL"

End If

If pblnOP Then
   strSQL = strSQL & ", 0" & vbCr
Else
   strSQL = strSQL & ", 1" & vbCr
End If

'Nuevos campos
strSQL = strSQL & ", " & plngIDBanco
strSQL = strSQL & ", " & plngIDPlaza
strSQL = strSQL & ", " & plngIDSucursal
strSQL = strSQL & ", " & pintFormaPago
strSQL = strSQL & ", '" & Format(pstrHora, "yyyy-mm-dd h:m:s") & "'"
strSQL = strSQL & ", " & pdblImporteTotal & ")" & vbCr

strSQL = strSQL & "IF @ERRORES = 0 " & vbCr
strSQL = strSQL & "BEGIN" & vbCr
'Orden_Status
```

```
    strSQL = strSQL & "Insert Into " & GSTR_TBL_ORDEN_STATUS
    strSQL = strSQL & " (" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_STATUS_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO_FLD_ID & ")" & vbCr

    strSQL = strSQL & " Values (" & plngEjercicio
    strSQL = strSQL & ", @IDOP"
    strSQL = strSQL & ", 1, '"
    strSQL = strSQL & Format(datFechaSolicitud, "yyyy-mm-dd Hh:Nn:Ss") & "'"
    strSQL = strSQL & ", " & pstrSolicita & ")" & vbCr
    strSQL = strSQL & "END" & vbCr & vbCr


    strSQL = strSQL & "SELECT @IDOP 'IDOP', @ERRORES 'ERROR'" & vbCr
    strSQL = strSQL & "SET NOCOUNT OFF" & vbCr

    'objBD.blnEjecutaQuery (strSQL)
    'strTansaccion = objBD.strBeginTrans

    Set rdsTransOP = objBD.rsdDevuelveRecordset(strSQL)
    If Not rdsTransOP Is Nothing Then
       If Not rdsTransOP.EOF And Not rdsTransOP.BOF Then
          plngIDOP = rdsTransOP.Fields("IDOP")
          lngErrorSQL = rdsTransOP.Fields("ERROR")
       End If
    End If

    If lngErrorSQL = 0 Then
       NuevaOrden = True
       strSQL = "COMMIT TRAN NuevaOrdenPago"
       objBD.blnEjecutaQuery (strSQL)
    Else   'Hubo un error
       NuevaOrden = False
       strSQL = "ROLLBACK TRAN NuevaOrdenPago"
       objBD.blnEjecutaQuery (strSQL)
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          strSQL = "ROLLBACK TRAN NuevaOrdenPago"
          objBD.blnEjecutaQuery (strSQL)

          NuevaOrden = False
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function

Public Function ImputacionOrden(ByVal plngEjercicio As Long, _
                                ByRef plngIDOP() As Long, _
                                ByVal plngCuentaBancaria As Long, _
                                ByVal plngTipoPago As Long, _
```

```
                    ByVal pstrNumero As String, _
                    ByVal pdblImporte As Double, _
                    ByVal plngEmpleado As Long, _
                    ByVal pstrDescripcion As String, _
                    ByVal plngIDTraspaso As Long, _
                    ByVal plngEjercicioTraspaso As Long, _
                    ByVal pblnOP As Boolean) As Boolean


'Descripción:      Funcion para realizar la contabilización de una Orden o mas órdenes

    Const STR_FUNCION As String = MSTR_MODULO & ".ImputacionOrden"

    Const LNG_STATUS_IMPUTADO As Long = 4
    Const LNG_TRASPASO_PERIODO As Long = 1
    Const LNG_STATUS_TRASPASO_PENDIENTE As Long = 1

    Dim strSQL              As String
    Dim objBD               As SIFIBD.BDAcceso
    Dim intIdConsolida      As Integer
    Dim strFecha As String
    Dim strTransaccion As String
    Dim objTraspaso As Traspasos
    Dim blnActualizaOP As Boolean
    Dim strOP As String
    Dim blnOP As Boolean
    Dim lngCont As Long
    Dim blnEjecutaStatus As Boolean


On Error GoTo ErrorHandler

    strFecha = GetDateFromServer

    strSQL = "INSERT INTO " & GSTR_TBL_REFERENCIAS & " (" & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPRESO & ")" & vbCr

    strSQL = strSQL & " Values (" & vbCr
    strSQL = strSQL & "'" & Trim$(pstrNumero) & "', " & vbCr
    strSQL = strSQL & plngCuentaBancaria & ", " & vbCr
    strSQL = strSQL & plngTipoPago & ", " & vbCr
    strSQL = strSQL & pdblImporte & "," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrDescripción)) & "', " & vbCr
    strSQL = strSQL & "0)"


    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans

    'si se pudo ejecutar la insercion
    If objBD.blnEjecutaQuery(strSQL) Then

        For lngCont = 0 To UBound(plngIDOP)
```

```
    strOP = "INSERT INTO " & GSTR_TBL_ORDEN_PAGO_REFERENCIAS & vbCr
    strOP = strOP & "(" & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strOP = strOP & GSTR_TBL_ORDEN_PAGO_FLD_ID & ", " & vbCr
    strOP = strOP & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", " & vbCr
    strOP = strOP & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strOP = strOP & GSTR_TBL_TIPO_PAGO_FLD_ID & ") " & vbCr

    'Valores
    strOP = strOP & " VALUES (" & vbCr
    strOP = strOP & "'" & Trim$(pstrNumero) & "', " & vbCr
    strOP = strOP & plngIDOP(lngCont) & ", " & vbCr
    strOP = strOP & plngEjercicio & ", " & vbCr
    strOP = strOP & plngCuentaBancaria & ", " & vbCr
    strOP = strOP & plngTipoPago & ") " & vbCr

    If objBD.blnEjecutaQuery(strOP) Then
       blnOP = True
    Else
       blnOP = False
       Exit For

    End If


Next

If blnOP Then

    If pblnOP Then
       Set objTraspaso = New Traspasos

       If plngIDTraspaso > 0 Then
          blnActualizaOP = objTraspaso.blnModificarStatusTraspaso(plngIDTraspaso,
              plngEjercicioTraspaso, LNG_STATUS_TRASPASO_PENDIENTE, plngEmpleado,
              LNG_TRASPASO_PERIODO, True, False)
       Else
          blnActualizaOP = True
       End If
    Else
       blnActualizaOP = True
    End If

    If blnActualizaOP Then

       For lngCont = 0 To UBound(plngIDOP)
          If blnActualizaStatus(plngIDOP(lngCont), plngEjercicio, LNG_STATUS_IMPUTADO,
              plngEmpleado) Then
             blnEjecutaStatus = True

          Else
             blnEjecutaStatus = False
             Exit For
          End If

       Next

    End If

End If
```

```
        If blnEjecutaStatus Then
            ImputacionOrden = objBD.blnCommitTrans(strTransaccion)

        Else
            ImputacionOrden = False
            Call objBD.blnRollBack(strTransaccion)
        End If


    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ImputacionOrden = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            Call objBD.blnRollBack(strTransaccion)

    End Select

    Set objBD = Nothing
    Set objTraspaso = Nothing

End Function


Public Function AutorizacionOrden(ByVal plngIDOP As Long, _
                                  ByVal plngEjercicio As Long, _
                                  ByVal pstrComentarios As String, _
                                  ByVal pblnAutorizado As Boolean, _
                                  ByVal plngIDUser As Long) As Boolean
'Descripción:       Funcion para autorizar una OP


    Const STR_FUNCION As String = MSTR_MODULO & ".AutorizacionOrden"

    Const LNG_STATUS_AUTORIZADO As Long = 2
    Const LNG_STATUS_RECHAZADO As Long = 3

Dim strSQL          As String
Dim objBD           As SIFIBD.BDAcceso
Dim lngIDStatus     As Long
Dim strTransaccion As String
Dim blnResultado As Boolean

On Error GoTo ErrorHandler

    If pblnAutorizado Then
        lngIDStatus = LNG_STATUS_AUTORIZADO
    Else
        lngIDStatus = LNG_STATUS_RECHAZADO
    End If


    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans
```

```
    If Not pblnAutorizado Then

        strSQL = "UPDATE " & GSTR_TBL_ORDEN_PAGO & " SET " & _
                        GSTR_TBL_ORDEN_PAGO_FLD_RECHAZADO_POR & " = '" & pstrComentarios & "' "
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & "=" &
            LNG_STATUS_RECHAZADO
        strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " &
            plngEjercicio & " AND " & _
                                GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngIDOP & vbCr




        If objBD.blnEjecutaQuery(strSQL) Then
            blnResultado = True

        Else
            blnResultado = False
        End If

    Else
        blnResultado = True
    End If

    If blnResultado Then

        If pblnAutorizado Then

            If blnActualizaStatus(plngIDOP, plngEjercicio, lngIDStatus, plngIDUser) Then
                AutorizacionOrden = objBD.blnCommitTrans(strTransaccion)


            Else
                objBD.blnRollBack strTransaccion
            End If

        Else
            If blnActualizaStatus(plngIDOP, plngEjercicio, lngIDStatus, plngIDUser, True) Then
                AutorizacionOrden = objBD.blnCommitTrans(strTransaccion)


            Else
                objBD.blnRollBack strTransaccion
            End If

        End If

    Else
        objBD.blnRollBack strTransaccion

    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
```

```
        objBD.blnRollBack strTransaccion

        AutorizacionOrden = False

        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select

    Set objBD = Nothing

End Function


Public Function ConsultaOrden(ByVal plngOrden As Long, _
                              ByVal plngEjercicio As Long) As ADODB.Recordset
'Descripción:       Funcion para realizar una consulta de una OP especifica

    Const STR_FUNCION As String = MSTR_MODULO & ".ConsultaOrden"

        Dim strSQL      As String
        Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT Distinct " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ID_PRESTADOR & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
                GSTR_TBL_OFICINA
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
        GSTR_TBL_AREA
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID & vbCr
    'prestador
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE
    strSQL = strSQL & " + ' ' + ISNULL(" & GSTR_TBL_PRESTADOR & "." &
        GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ' )"
    strSQL = strSQL & " + ' ' + ISNULL(" & GSTR_TBL_PRESTADOR & "." &
        GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ' )" & " AS " & GSTR_TBL_PRESTADOR & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TRASPASOS_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_TRASPASOS_FLD_EJERCICIO

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA & " as " & GSTR_TBL_EMPLEADO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES & vbCr
```

```
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TIPO_PAGO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & " AS MONEDA_INGRESO"
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID
    'strSQL = strSQL & ", SUM( " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE & ")  AS " &
            GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA & ".ID_Gui_Con" & vbCr

    'FROM
    strSQL = strSQL & " FROM " & GSTR_TBL_ORDEN_PAGO & vbCr
    'Orden Pago - Cuenta
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ORDEN_PAGO_CUENTA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID &
            vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & vbCr
    'Movimientos Contables
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MOVTOS_CONTABLES
    strSQL = strSQL & " ON " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO_CUENTA + "." +
            GSTR_TBL_ORDEN_PAGO_FLD_ID
    'Guia Rubros
    strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_GUIA_RUBROS
    strSQL = strSQL & " ON " & GSTR_TBL_GUIA_RUBROS & "." &
            GSTR_TBL_GUIA_RUBROS_FLD_ID_GUI_CON
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_GUIA_RUBROS_FLD_ID_GUI_CON
    'Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." & GSTR_TBL_RUBRO_FLD_ID &
            vbCr
    'Conceptos del Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_CONCEPTO_RUBRO & "." &
GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
```

```
strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
'Proveedor
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_PRESTADOR & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_ID &
    vbCr
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID &
    vbCr
'Moneda
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MONEDA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
'Centros de Costos
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CENTRO_COSTO & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
'Oficinas
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
'Areas
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
'WHERE
strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOrden & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CONCEPTO_RUBRO & "." &
    GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & " = '" &
    GSTR_CONCEPTO_RUBRO_NO_INDICA_IMPUESTO & "'"

'Cláusla Group
strSQL = strSQL & " GROUP BY " & vbCr
strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_ID_PRESTADOR & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID & vbCr
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_MATERNO
    & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TRASPASOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_TRASPASOS_FLD_EJERCICIO
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
```

```
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TIPO_PAGO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA & ".ID_Gui_Con" & vbCr

    'ORDER BY
    strSQL = strSQL & " ORDER BY " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO

    Set ConsultaOrden = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          Set ConsultaOrden = Nothing

          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function


Public Function ConsultaOPImprimir(ByVal plngOrden As Long, _
                        ByVal plngEjercicio As Long) As ADODB.Recordset
'Descripción:    Funcion para realizar una consulta de una OP especifica
'                y obtener los datos necesarios para la impresión
    Const STR_FUNCION As String = MSTR_MODULO & ".ConsultaOPImprimir"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler
```

```
    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_REFERENCIAS_FLD_IMPORTE & " AS IMPORTEOP" & vbCr

    strSQL = strSQL & " FROM " & GSTR_TBL_MOVTOS_CONTABLES

    'Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO & vbCr

    'Orden Pago
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ORDEN_PAGO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & vbCr

    'Moneda
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MONEDA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr

    'Referencias
'   strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_REFERENCIAS & vbCr
'   strSQL = strSQL & " ON " & GSTR_TBL_REFERENCIAS & "." &
            GSTR_TBL_REFERENCIAS_FLD_IDOP
'   strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & vbCr

    strSQL = strSQL & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOrden & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio & vbCr

    strSQL = strSQL & " GROUP BY " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." & G
            STR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
```

```
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_REFERENCIAS_FLD_IMPORTE & vbCr

    strSQL = strSQL & "Order by " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO

    Set ConsultaOPImprimir = objBD.rsdDevuelveRecordset(strSQL)



ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ConsultaOPImprimir = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function ListadoChequesLiberar(ByVal pstrBanco As String) As ADODB.Recordset

Descripción: Funcion para obtener una lista de los cheques
por liberar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoChequesLiberar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso
    Dim strCveBanco As String
    Dim objCatalogo As Catalogos

    On Error GoTo ErrorHandler

    'Obtenemos la Clave del Banco
    Set objCatalogo = New Catalogos
    strCveBanco = objCatalogo.strObtieneCve(GSTR_TBL_BANCOS_FLD_ID, pstrBanco,
            GSTR_TBL_BANCOS, _
```

```
        GSTR_TBL_BANCOS_FLD_DESCRIPCION, False)

Set objBD = New SIFIBD.BDAcceso


strSQL = "SELECT " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_NUMERO
strSQL = strSQL & ", Sum(" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ") As " &
        GSTR_TBL_REFERENCIAS_FLD_IMPORTE & vbCr
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_REF_CON
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE &
        vbCr
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & ".IDConsolida"

strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr

strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA
        & vbCr

strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
        "NOMBRE_OFICINA"
strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
        "NOMBRE_AREA"
strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

strSQL = strSQL + " FROM "
strSQL = strSQL + GSTR_TBL_REFERENCIAS
         'Orden_Pago
strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
strSQL = strSQL + " ON " + GSTR_TBL_REFERENCIAS + "." + GSTR_TBL_REFERENCIAS_FLD_IDOP
        & " = "
strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr

'Rubro
strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_RUBRO & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & vbCr

'Oficina
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & " = "
strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr

'Area
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
```

```
    strSQL = strSQL & " ON " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & " = "
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID & vbCr

    'Centro Costos
    strSQL = strSQL + " INNER JOIN " + GSTR_TBL_CENTRO_COSTO & vbCr
    strSQL = strSQL + " ON " + GSTR_TBL_CENTRO_COSTO + "." + GSTR_TBL_OFICINA_FLD_ID + " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_OFICINA_FLD_ID + " And " & vbCr
    strSQL = strSQL + GSTR_TBL_CENTRO_COSTO + "." + GSTR_TBL_AREA_FLD_ID + " = " +
        GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_AREA_FLD_ID + " AND "
    strSQL = strSQL + GSTR_TBL_CENTRO_COSTO + "." +
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO + " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
        & vbCr


    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & " = 1"   'Cheque
    strSQL = strSQL & " AND (" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_LOTE & " IS NULL "
    strSQL = strSQL & " OR " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_LOTE & " = 0 )" & vbCr
    strSQL = strSQL & " AND (" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & " = 1"
    strSQL = strSQL & " OR " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & "= 0)" & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_RUBRO_FLD_ID & " IN (

Select IDRubro From Rubro " & vbCr
    strSQL = strSQL & " where IDConcepto_Rubro = 1502 and IDBanco = " & Val(strCveBanco) & ")" & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & " = 0
        " & vbCr
    strSQL = strSQL & " AND (" & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & " = 10 "  'Autorizados, no rechazadas, y
Reimpresas
    strSQL = strSQL & " OR " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & "= 7)" & vbCr



    strSQL = strSQL + " Group By "
    strSQL = strSQL & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_NUMERO
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_REF_CON
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & ".IDConsolida" & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
```

```
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA
          & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    Set ListadoChequesLiberar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          Set ListadoChequesLiberar = Nothing

          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing


End Function
```

Public Function ListadoTransferenciasEnviar() As ADODB.Recordset

Descripción: Funcion para obtener una lista de los transferencias por liberar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoTransferenciasEnviar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso


    strSQL = "SELECT " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON
    strSQL = strSQL & ", Sum(" & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & ") As " &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCOS_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & ".CTA_BCO_DES"

          strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_NOMBRE & " AS NOMBRE_PRESTADOR " & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
```

```
            strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
            GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

    strSQL = strSQL & ", IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_NOMBRE & ",' ')" & vbCr
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ')"
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ') AS " & GSTR_TBL_PRESTADOR & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_CUENTAS_BANCARIAS_FLD_CUENTA_BANCARIA
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
"NOMBRE_OFICINA"
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
"NOMBRE_AREA"
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    strSQL = strSQL & "," & GSTR_TBL_BANCOS & ".IDBMR" & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_CVEBANCO
    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_DESCRIPCION
    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & ".CveBancoPlazaBMR"
    strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL & "." & GSTR_TBL_SUCURSAL_FLD_CVEBANCO &
vbCr

    strSQL = strSQL & " FROM " & GSTR_TBL_REF_TRAN_AUT & vbCr

    'Orden_Pago
    strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
    strSQL = strSQL + " ON " + GSTR_TBL_REF_TRAN_AUT + "." +
GSTR_TBL_REF_TRAN_AUT_FLD_IDOP & " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr


    Set ListadoTransferenciasEnviar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ListadoTransferenciasEnviar = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

```
Public Function ListadoConciliacion(ByVal pdtmFechaIni As Date, ByVal pdtmFechaFin As Date, ByVal
        plngIDRubro As Long) As ADODB.Recordset

'Descripción: Funcion para obtener una lista de las OP a conciliar

   Const STR_FUNCION As String = MSTR_MODULO & ".ListadoConciliacion"

   Dim strSQL     As String
   Dim objBD      As SIFIBD.BDAcceso

   On Error GoTo ErrorHandler

   Set objBD = New SIFIBD.BDAcceso


   strSQL = "SELECT " & GSTR_TBL_REFERENCIAS_FLD_ID
        strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT & ", " &
        GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_IDOP
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_NUMERO
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_IMPORTE
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_ESTATUS
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_FECHA_CON


   strSQL = strSQL & " FROM " & GSTR_TBL_REFERENCIAS & vbCr

   strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_ESTATUS & " <> 'C'"
        strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & " = " &
        plngIDRubro & vbCr
   strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT & vbCr
   strSQL = strSQL & " BETWEEN '" & Format(pdtmFechaIni, "yyyy-mm-dd") & " 00:00:00'"
   strSQL = strSQL & " AND '" & Format(pdtmFechaFin, "yyyy-mm-dd") & " 23:59:59'" & vbCr

   strSQL = strSQL & "Order by " & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_NUMERO
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_IDOP

   Set ListadoConciliacion = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set ListadoConciliacion = Nothing

         mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select

   Set objBD = Nothing

End Function


Public Function ListadoConciliacionBancos(ByVal pstrCtaBanca As String) As ADODB.Recordset
'Public Function ListadoConciliacionBancos(ByVal pdtmFechaIni As Date, ByVal pdtmFechaFin As Date,
ByVal pstrCtaBanca As String) As ADODB.Recordset
```

'Descripción: Funcion para obtener una lista de las OP a conciliar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoConciliacionBancos"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso


    strSQL = "SELECT * FROM Registro_Conciliacion_Bancos"

    strSQL = strSQL & " WHERE Estatus_Conciliacion <> 'C' OR Estatus_Conciliacion IS NULL AND

    Set ListadoConciliacionBancos = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ListadoConciliacionBancos = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function ListadoLoteRegenerar(ByVal plngLote As Long) As ADODB.Recordset

Descripción: Funcion para obtener una lista de los Transferencias que pertenencen
' al Lote por Regenerar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoLoteRegenerar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID
            strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
            strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_REFERENCIAS_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
```

```
  strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
  strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
 strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
 GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr

 strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_NUMERO
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & vbCr
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_IMPORTE
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_REF_CON
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE &
 vbCr

  strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
        strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." &
        GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & vbCr

        strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & "
        AS " & "NOMBRE_OFICINA"
  strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
  "NOMBRE_AREA"
  strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
  GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

  strSQL = strSQL + " FROM "
  strSQL = strSQL + GSTR_TBL_REFERENCIAS
  'Orden_Pago
  strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
        strSQL = strSQL + " ON " + GSTR_TBL_REFERENCIAS + "." +
        GSTR_TBL_REFERENCIAS_FLD_IDOP & " = "
  strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr

  'Rubro
  strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_RUBRO & vbCr
  strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & vbCr


  Set ListadoLoteRegenerar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
  Select Case Err.Number
     Case 0
        'do nothing
     Case Else
        Set ListadoLoteRegenerar = Nothing

        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
  End Select

  Set objBD = Nothing
```

End Function

Public Function ListadoLoteTransfRegenerar(ByVal plngLote As Long) As ADODB.Recordset

Descripción: Funcion para obtener una lista de las Transferencias que pertenencen
' al Lote por Regenerar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoLoteTransfRegenerar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

          strSQL = "SELECT " & GSTR_TBL_REF_TRAN_AUT & "." &
          GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON
    strSQL = strSQL & ", Sum(" & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & ") As " &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA 'Cta Origen
    strSQL = strSQL & ", " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL      'Sucursal Origen

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCOS_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & ".CTA_BCO_DES"
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_OTRO_BENEFICIARIO

          strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." &
          GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & " AS CUENTA_ORIGEN "

    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE &
" AS NOMBRE_PRESTADOR " & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
          strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

          strSQL = strSQL & ", IsNull(" & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_NOMBRE & ",' ')" & vbCr
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ')"
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ') AS " & GSTR_TBL_PRESTADOR & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_CUENTAS_BANCARIAS_FLD_CUENTA_BANCARIA
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & vbCr
```

```
   strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr

   strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
"NOMBRE_OFICINA"
   strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
   "NOMBRE_AREA"
   strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
   GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

   strSQL = strSQL & "," & GSTR_TBL_BANCOS & ".IDBMR" & vbCr

   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_CVEBANCO
   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_DESCRIPCION
   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & ".CveBancoPlazaBMR"
         strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL & "." &
         GSTR_TBL_SUCURSAL_FLD_CVEBANCO & vbCr

   strSQL = strSQL & " FROM " & GSTR_TBL_REF_TRAN_AUT & vbCr



   Set ListadoLoteTransfRegenerar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set ListadoLoteTransfRegenerar = Nothing

         mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select

   Set objBD = Nothing

End Function
```

Public Function blnEliminaOrden(ByVal plngOP As Long, ByVal plngEjercicio As Long, ByVal plngEmpleado As Long) As Boolean
Descripción: Funcion para ejecutar Borrar una cuenta asociado a una OP

```
   Const STR_FUNCION As String = MSTR_MODULO & ".blnEliminaOrden"

   Const LNG_STATUS_ELIMINAR As Long = 9

   Dim strSQL      As String
   Dim objBD       As SIFIBD.BDAcceso
   Dim rsDatos     As ADODB.Recordset
   Dim strTransaccion As String
   Dim blnResultado As Boolean
   Dim strCondicion As String

On Error GoTo ErrorHandler

         strCondicion = " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio
         & vbCr
   strCondicion = strCondicion & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOP
```

```
    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans

    If Trim$(strTransaccion) <> vbNullString Then

        'Elimino los impuestos
        strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO_CUENTA_IMPUESTO & vbCr
        strSQL = strSQL & strCondicion

        blnResultado = objBD.blnEjecutaQuery(strSQL)

        'Elimino los movimientos contables
        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_MOVTOS_CONTABLES & vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOP

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If


        'Elimino las cuentas
        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO_CUENTA & vbCr
            strSQL = strSQL & strCondicion

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If



        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO & vbCr
            strSQL = strSQL & strCondicion

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If

        If blnResultado Then

            If blnActualizaStatus(plngOP, plngEjercicio, LNG_STATUS_ELIMINAR, plngEmpleado, True) Then
                objBD.blnCommitTrans strTransaccion
                blnEliminaOrden = True
            Else
                objBD.blnRollBack strTransaccion
                blnEliminaOrden = False
            End If

        End If

    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
```

```
        Case Else
            objBD.blnRollBack strTransaccion

            blnEliminaOrden = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION


    End Select

    Set objBD = Nothing

End Function
```

Public Function blnNuevaOrdenConceptoRubro(ByVal plngOP As Long, ByVal plngConsecutivo As Long, ByVal plngRubro As Long, ByVal pdatFechaSolicitud As Date, ByVal plngMvto As Long, ByVal plngMoneda As Long, ByVal pdblImporte As Double) As Boolean

Descripción: Funcion para Guardar los rubros de cada una de las partidas
de la OP

```
Const STR_FUNCION As String = MSTR_MODULO & ".blnNuevaOrdenConceptoRubro"

Dim strSQL          As String
Dim objBD           As SIFIBD.BDAcceso
Dim lngIDMovimiento As Long

On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    blnNuevaOrdenConceptoRubro = False

    'Valor del IDMovimiento
        lngIDMovimiento = objBD.lngGetNewId(GSTR_TBL_MOVTOS_CONTABLES,
        GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO)


    strSQL = "Insert Into " & GSTR_TBL_MOVTOS_CONTABLES
    strSQL = strSQL & " (" & GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_CONSECUTIVO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE & ")"

    strSQL = strSQL & " Values (" & lngIDMovimiento
    strSQL = strSQL & ", " & plngOP
    strSQL = strSQL & ", " & plngConsecutivo
    strSQL = strSQL & ", " & plngRubro
    strSQL = strSQL & ", '" & Format(pdatFechaSolicitud, "yyyy-mm-dd") & "'"
    strSQL = strSQL & ", " & plngMvto
    strSQL = strSQL & ", " & plngMoneda
    strSQL = strSQL & ", " & pdblImporte & ")"

    blnNuevaOrdenConceptoRubro = objBD.blnEjecutaQuery(strSQL)
```

```
ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnNuevaOrdenConceptoRubro = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function blnActualizaCaratulaMov(ByVal plngOrden As Long, ByVal plngCaratula As Long, ByVal plngAsiento As Long, ByVal pblnCancelacion As Boolean) As Boolean

```
' Esta función actualiza el Asiento y la Carátula en las tablas de Movimientos y Referencias
' Si se está cancelando entonces no se actualiza Referencia y se ingresan nuevos registros en
' Movimientos

    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaCaratulaMov"
    Const LNG_VALOR_SISE As Long = 1

    Dim strSQL          As String
    Dim objBD           As SIFIBD.BDAcceso
    Dim strTransaccion  As String
    Dim strSQLMov       As String
    Dim rdsMovs         As ADODB.Recordset
    Dim intNaturaleza   As Integer

On Error GoTo errHandler

    Set objBD = New SIFIBD.BDAcceso

    If Not pblnCancelacion Then
        strSQLMov = "UPDATE " & GSTR_TBL_MOVTOS_CONTABLES
            strSQLMov = strSQLMov & " SET " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO & "=" &
            plngAsiento & ", " & vbCr
        strSQLMov = strSQLMov & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & "=" &
        plngCaratula & vbCr
        strSQLMov = strSQLMov & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & "=" &
        plngOrden

        strSQL = "UPDATE " & GSTR_TBL_REFERENCIAS & vbCr
        strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" & plngAsiento & vbCr
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula & vbCr
        strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & "=" & plngOrden

    Else
        strSQL = "SELECT * FROM " & GSTR_TBL_MOVTOS_CONTABLES
        strSQL = strSQL & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & "=" & plngOrden

        Set rdsMovs = objBD.rsdDevuelveRecordset(strSQL)
        If Not rdsMovs Is Nothing Then
            If Not rdsMovs.BOF And Not rdsMovs.EOF Then
                With rdsMovs
                    .MoveFirst
                    Do While Not .EOF
```

```
                    intNaturaleza = .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO)
                    If intNaturaleza = 1 Then
                        intNaturaleza = 2
                    Else
                        intNaturaleza = 1
                    End If
            blnNuevaOrdenConceptoRubro plngOrden,
            .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_CONSECUTIVO), _
                        .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO),
            Format(GetDateFromServer, "YYYY-MM-DD"), _
                        intNaturaleza, .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA), _
                        .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE)
                    .MoveNext
                Loop
                'Actualizamos a los nuevos movimientos la Caratula y el Asiento
                strSQLMov = "UPDATE " & GSTR_TBL_MOVTOS_CONTABLES
            strSQLMov = strSQLMov & " SET " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO & "=" &
            plngAsiento & ", " & vbCr
                strSQLMov = strSQLMov & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & "=" &
                plngCaratula & vbCr
                strSQLMov = strSQLMov & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP &
                "=" & plngOrden & vbCr
                strSQLMov = strSQLMov & " AND " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO &
                " Is Null "
                strSQLMov = strSQLMov & " AND " & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA
                & " Is Null "

                strSQL = "UPDATE " & GSTR_TBL_ORDEN_PAGO & vbCr
                    strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" &
                    plngAsiento & vbCr
                strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula
                & vbCr
                strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" & plngOrden

                'blnActualizaCaratulaMov = objBD.blnEjecutaQuery(strSQLMov)
            End With
        End If
    End If
End If

strTransaccion = objBD.strBeginTrans

If objBD.blnEjecutaQuery(strSQL) Then
    If objBD.blnEjecutaQuery(strSQLMov) Then
        blnActualizaCaratulaMov = objBD.blnCommitTrans(strTransaccion)
    Else
        Call objBD.blnRollBack(strTransaccion)
    End If
Else
    Call objBD.blnRollBack(strTransaccion)
End If

errHandler:
    Select Case Err.Number

    Case 0
        'Do nothing
    Case Else
        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
        blnActualizaCaratulaMov = False
```

```
            Call objBD.blnRollBack(strTransaccion)
        End Select

        'Libero la memoria
        Set objBD = Nothing

    End Function
```

CLASE MANEJO DE ERRORES

```
Public Sub RegistraError(ByVal lngErrNum As Long, _
                    ByVal strDescripcion As String, _
                    ByVal pstrModulo As String, _
                    Optional ByVal pstrUserName As String = vbNullString, _
                    Optional ByVal pstrLogin As String = vbNullString, _
                    Optional ByVal plngAbreLog As Long = 0)
'Descripción:       Funcion para obtener la cadena de conexion
'                   de un archivo de texto y guardarla en una
'                   propiedad que es global y accesible de cualquier
'                   lugar del proyecto

   Dim intIDFile       As Long

   intIDFile = FreeFile
   Open App.Path + "\SIF_Errores.log" For Append As #intIDFile

   Print #intIDFile, vbCrLf & vbCrLf
   Print #intIDFile, String(50, "$")
   Print #intIDFile, "NÚMERO DE ERROR  : " & lngErrNum
   Print #intIDFile, "DESCRIPCIÓN      : " & strDescripcion & vbCrLf
   Print #intIDFile, "ORIGEN           : " & pstrModulo
   Print #intIDFile, "FECHA            : " & Now
   Print #intIDFile, String(50, "$")

   Close intIDFile

End Sub

Private Sub CreaLog(pintIdFile As Integer)
'Descripción:       Funcion para obtener abrir el archivo Log y escribir la fecha
'                   y hora del acceso

          Dim intFreeFile As Integer
          Dim strConn As String


   Open App.Path + "\SIF_errores.log" For Append As #pintIdFile

   Print #pintIdFile, vbNullString
   Print #pintIdFile, vbNullString
   Print #pintIdFile,
"++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++"

   Print #pintIdFile, "Se inicio el sistema SIF el dia ", Format(Date, "dddd dd/mm/yyyy"), Time
   Print #pintIdFile, vbNullString

   Close pintIdFile

End Sub
```

CLASE EMPLEADO

Public Function blnNuevoEmpleado(ByVal pstrNombre As String, _
ByVal pstrPaterno As String, _
ByVal pstrMaterno As String, _
ByVal pblnActivo As Boolean, _
ByVal pstrLogin As String, _
ByVal pstrPassword As String, _
ByVal plngOficina As Long, _
ByVal plngArea As Long, _
ByVal plngDepto As Long, _
ByVal pblnUpdate As Boolean, _
ByVal pstrContabilidad As String, _
ByVal pstrLogSISE As String, _
ByVal pstrLogSABE As String, _
ByVal pstrPwdFunEsp As String, _
ByVal plngIDRol As Long, _
ByVal plngIDConsulta As Long, _
ByVal plngIDNivelConsulta As Long, _
Optional ByVal plngID As Long) As Boolean

Descripción: Función para registro de Usuarios al sistema

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnNuevoEmpleado"

    Dim strSql As String
    Dim lngIDEmpleado As Long
    Dim objBD As SIFIBD.BDAcceso
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strContabilidad As String
    Dim strPwdFunEsp As String
    Dim objRolEmpleado As SIFISeguridad.Seguridad
    Dim strSql2 As String
    Dim blnCreaRolOficina As Boolean


On Error GoTo errHandler

    'Mando encriptar el password
    Set objEncripta = New xmsEncriptador.Encriptador

    pstrPassword = objEncripta.EncriptaString(pstrPassword)

    Set objBD = New SIFIBD.BDAcceso

    strSql2 = "INSERT INTO " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSql2 = strSql2 & "(" & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql2 = strSql2 & GSTR_TBL_OFICINA_FLD_ID & ")" & vbCr
    strSql2 = strSql2 & " VALUES (" & plngIDRol & ", " & vbCr
    strSql2 = strSql2 & plngOficina & ")"




    If pblnUpdate Then
        strSql = "UPDATE " & GSTR_TBL_EMPLEADO & vbCr
        strSql = strSql & " SET " & vbCr
        strSql = strSql & GSTR_TBL_OFICINA_FLD_ID & "= " & plngOficina & ", " & vbCr
        strSql = strSql & GSTR_TBL_AREA_FLD_ID & "=" & plngArea & ", " & vbCr
```

```
    strSql = strSql & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & "=" & plngDepto & ", " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_NOMBRE & "='" & UCase$(Trim$(pstrNombre)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PATERNO & "='" & UCase$(Trim$(pstrPaterno)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_MATERNO & "='" & UCase$(Trim$(pstrMaterno)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOGIN & "='" & UCase$(Trim$(pstrLogin)) & "', " &
             vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" & pstrPassword & "', " & vbCr



    If pstrContabilidad <> vbNullString Then
       strContabilidad = objEncripta.EncriptaString(pstrContabilidad)
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & "='" & strContabilidad & "', " &
             vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & "=NULL," & vbCr
    End If

    'Modificacion 05/05/2003 Agregar usuario de SISE y SABE y Pwd de Funciones Especiales

    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SISE & "='" & UCase$(Trim$(pstrLogSISE))
             & "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SABE & "='" & UCase$(Trim$(pstrLogSABE))
             & "', " & vbCr

    If pstrPwdFunEsp <> vbNullString Then
       strPwdFunEsp = objEncripta.EncriptaString(pstrPwdFunEsp)
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & "='" & strPwdFunEsp & "', " &
             vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & "=NULL," & vbCr
    End If


    If pblnActivo Then
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & "=1, " & vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & "=0, " & vbCr
    End If

    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_FECHA_ACTUALIZACION & "='" & Format(Now,
             "YYYY-MM-DD") & "'"
    strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngID

If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOGIN, GSTR_TBL_EMPLEADO,
     Trim$(pstrLogin), GSTR_TBL_EMPLEADO_FLD_ID, plngID) Then
    If objBD.blnBuscaRegistro(Trim$(pstrLogin), GSTR_TBL_EMPLEADO,
             GSTR_TBL_EMPLEADO_FLD_LOGIN, False) Then
       blnNuevoEmpleado = False
    Else
       If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SISE,
             GSTR_TBL_EMPLEADO, Trim$(pstrLogSISE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
     Then

          If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
             GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
            blnNuevoEmpleado = False
```

```
        Else

            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                    GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID,
                    plngID) Then

                If Trim$(pstrLogSABE) <> vbNullString And
                    objBD.blnBuscaRegistro(Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO,
                    GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
                  blnNuevoEmpleado = False
                Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                   blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                End If
            Else
               blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
               blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)


            End If


        End If

    Else
       If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
             GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
     Then

          If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
              GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
            blnNuevoEmpleado = False

          Else
             blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
             blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
          End If
       Else
             blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
             blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
       End If
    End If
    End If
Else
    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SISE,
              GSTR_TBL_EMPLEADO, Trim$(pstrLogSISE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
              Then

          If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
              GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
             blnNuevoEmpleado = False
          Else

             If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                     GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID,
                     plngID) Then

                If Trim$(pstrLogSABE) <> vbNullString And
                     objBD.blnBuscaRegistro(Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO,
                     GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
```

```
                    blnNuevoEmpleado = False
                  Else
                    blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                     blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                  End If
               Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
               End If


            End If

         Else
            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                  GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
          Then

               If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
                  GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
                 blnNuevoEmpleado = False

               Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
               End If
            Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)

            End If
         End If
      End If

      'Actualiza el Perfil del Usuario
         If blnNuevoEmpleado Then
            Set objRolEmpleado = New SIFISeguridad.Seguridad

            objRolEmpleado.Empleado = lngIDEmpleado

            If objRolEmpleado.blnAsignaRolUsuario(plngIDRol, _
                        plngIDConsulta, _
                        plngIDNivelConsulta, _
                        plngID) Then

            End If
            Set objRolEmpleado = Nothing

         End If


   Else

      If objBD.blnBuscaRegistro(Trim$(pstrLogin), GSTR_TBL_EMPLEADO,
          GSTR_TBL_EMPLEADO_FLD_LOGIN, False) Then
         blnNuevoEmpleado = False
      Else
         If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
                  GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
            blnNuevoEmpleado = False
```

```
Else
   If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
         GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
      blnNuevoEmpleado = False
   Else


      lngIDEmpleado = objBD.lngGetNewId(GSTR_TBL_EMPLEADO,
         GSTR_TBL_EMPLEADO_FLD_ID)

      strSql = "INSERT INTO " & GSTR_TBL_EMPLEADO & vbCr
      strSql = strSql & "(" & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_NOMBRE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PATERNO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_MATERNO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOGIN & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PASSWORD & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_FECHA_ACTUALIZACION & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SISE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SABE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & ") " & vbCr

      strSql = strSql & " VALUES (" & vbCr

      strSql = strSql & lngIDEmpleado & ", " & vbCr
      strSql = strSql & plngOficina & ", " & vbCr
      strSql = strSql & plngArea & ", " & vbCr
      strSql = strSql & plngDepto & ", " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrNombre)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrPaterno)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrMaterno)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrLogin)) & "', " & vbCr
      strSql = strSql & "'" & pstrPassword & "', " & vbCr

      If pblnActivo Then
         strSql = strSql & " 1, " & vbCr
      Else
         strSql = strSql & " 0, " & vbCr
      End If

      strSql = strSql & "'" & Format(Now, "YYYY-MM-DD") & "'," & vbCr

      If pstrContabilidad <> vbNullString Then
         strContabilidad = objEncripta.EncriptaString(pstrContabilidad)
         strSql = strSql & "'" & strContabilidad & "', " & vbCr
      Else
         strSql = strSql & "NULL, " & vbCr
      End If

      strSql = strSql & "'" & UCase$(Trim$(pstrLogSISE)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrLogSABE)) & "', " & vbCr

      If pstrPwdFunEsp <> vbNullString Then
         strPwdFunEsp = objEncripta.EncriptaString(pstrPwdFunEsp)
         strSql = strSql & "'" & strPwdFunEsp & "')" & vbCr
```

```
                Else
                    strSql = strSql & "NULL)" & vbCr
                End If


                blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                    If blnNuevoEmpleado Then
                        Set objRolEmpleado = New SIFISeguridad.Seguridad

                        objRolEmpleado.Empleado = lngIDEmpleado

                        If objRolEmpleado.blnAsignaRolUsuario(plngIDRol, _
                                plngIDConsulta, _
                                plngIDNivelConsulta, _
                                lngIDEmpleado) Then

                        End If
                        Set objRolEmpleado = Nothing

                    End If


            End If
        End If
    End If

    End If

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnNuevoEmpleado = False

    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set objEncripta = Nothing

End Function
```

Public Function rsdPerfilEmpleado() As ADODB.Recordset

Descripción: Función para cargar el perfil del Usuario dado su número de clave

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdPerfilEmpleado"

    Dim strSql As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler


    strSql = "SELECT " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
```

```
    strSql = strSql & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_EMPLEADO_ULTIMO_ROL &
                ", " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & ", "
                & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " &
                vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
                & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_DESCRIPCION & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_CANTIDAD & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_TIPO_CONSULTA
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID


    Set objBD = New SIFIBD.BDAcceso

    Set rsdPerfilEmpleado = objBD.rsdDevuelveRecordset(strSql)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
            Set rsdPerfilEmpleado = Nothing

    End Select

    'Libera la memoria
    Set objBD = Nothing

End Function
```

Public Function rsdListaEmpleados(Optional ByVal plngNumEmpleado As Long) As ADODB.Recordset

Descripción: Función para desplegar una lista de Usuarios

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdListaEmpleados"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo errHandler

    strSql = "SELECT " & GSTR_TBL_EMPLEADO & ".*," & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " &
                vbCr
```

```
    ' strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
                    & ", " & vbCr
    strSql = strSql & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_NIVEL_CONSULTA & ", " & vbCr
    strSql = strSql & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_DESCRIPCION_NIVEL & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_DESCRIPCION & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_CANTIDAD & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_TIPO_CONSULTA
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_NIVEL_CONSULTA & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." &
            GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA
    strSql = strSql & " = " & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_NIVEL_CONSULTA


    If Not IsMissing(plngNumEmpleado) Then

        If plngNumEmpleado > 0 Then
            strSql = strSql & vbCr & " WHERE " & GSTR_TBL_EMPLEADO & "." &
            GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngNumEmpleado

        End If

    End If

    strSql = strSql & " Order by " & GSTR_TBL_EMPLEADO & "." &
            GSTR_TBL_EMPLEADO_FLD_NOMBRE & "," & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_PATERNO & "," & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_MATERNO

    Set objBD = New SIFIBD.BDAcceso

    Set rsdListaEmpleados = objBD.rsdDevuelveRecordset(strSql)

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function ObtieneNombreUsuario(ByVal plngIDUser As Long, ByRef plngIDOficina As Long) As String

Descripción: Función para consultar y obtener los datos personales del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneNombreUsuario"

Dim strSql      As String
Dim objBD       As SIFIBD.BDAcceso
Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneNombreUsuario = vbNullString
    plngIDOficina = -1

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_EMPLEADO_FLD_NOMBRE & " + ' ' + " & _
                     GSTR_TBL_EMPLEADO_FLD_PATERNO & " + ' ' + " & _
                     GSTR_TBL_EMPLEADO_FLD_MATERNO & ", " & _
                     GSTR_TBL_OFICINA_FLD_ID & " from " & _
                     GSTR_TBL_EMPLEADO & " where  " & _
                     GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser


    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)
    If Not rsEmpleado Is Nothing Then
       If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
          rsEmpleado.MoveFirst
          ObtieneNombreUsuario = rsEmpleado(0)
          plngIDOficina = rsEmpleado(1)
          rsEmpleado.Close
       End If
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          ObtieneNombreUsuario = vbNullString
          plngIDOficina = -1
          objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function
```

```
Public Function ObtieneRolUsuario(ByVal plngIDUser As Long) As Long
'Descripion:    Funcion para obtener el Rol de un usuario

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneRolUsuario"

Dim strSql      As String
Dim objBD       As SIFIBD.BDAcceso
Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneRolUsuario = -1

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " from " & GSTR_TBL_ROL_EMPLEADO
    strSql = strSql & " where " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser

    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
        If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
            rsEmpleado.MoveFirst
            ObtieneRolUsuario = rsEmpleado.Fields(GSTR_TBL_ROL_FLD_ID)
            rsEmpleado.Close
        End If
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ObtieneRolUsuario = -1
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function


Public Function ObtieneNivelConsulta(ByVal plngIDUser As Long) As Long
'Descripion:    Funcion para obtener el Tipo de consulta que puede realizar

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneNivelConsulta"

            Dim strSql      As String
            Dim objBD       As SIFIBD.BDAcceso
            Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneNivelConsulta = -1

    Set objBD = New SIFIBD.BDAcceso
```

```
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA
    strSql = strSql & " from " & GSTR_TBL_ROL_EMPLEADO
    strSql = strSql & " where " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser

    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
       If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
          rsEmpleado.MoveFirst
          ObtieneNivelConsulta = rsEmpleado.Fields(GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA)
          rsEmpleado.Close
       End If
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          ObtieneNivelConsulta = -1
          objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function


Public Function ObtieneUsuarioDepartamento(ByVal plngIDUser As Long) As String
'Descripción:    Funcion para obtener el departamento de un usuario

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneUsuarioDepartamento"

          Dim strSql      As String
          Dim objBD       As SIFIBD.BDAcceso
          Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneUsuarioDepartamento = vbNullString

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & " from " & _
                    GSTR_TBL_CENTRO_COSTO & ", " & _
                    GSTR_TBL_EMPLEADO & " where " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_OFICINA_FLD_ID & " = " & _
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_AREA_FLD_ID & " = " & _
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & " = " &
_
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO &
                    " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser
```

```
    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
        If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
            rsEmpleado.MoveFirst
            ObtieneUsuarioDepartamento =
             rsEmpleado.Fields(GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO)
            rsEmpleado.Close
        End If
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ObtieneUsuarioDepartamento = vbNullString
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function
```

CLASE DE CATALOGOS

'Declaración del Enum para selección de catálogos

Public Enum enmCatalogos
catTipoTelefono = 0
catRubro = 1
catNaturalezaRubro = 2
catEmpleado = 3
CatCentroCosto = 4
catDepartamento = 5
CatOficina = 6
CatMoneda = 8
CatTipoCambio = 9
CatPrestadores = 10
CatArea = 11
CatOficinasArea = 12
catBanco = 18
catOrdenStatus = 20
catOrdenPago = 21
CatOrdenPagoCuenta = 22
catTipoConsulta = 23
catConceptoRubro = 24
catCuentasBancarias = 27
catOficinaImpuesto = 28
catTipoDomicilio = 29
catProveedorDomicilio = 30
catProveedorTelefono = 31
catEdoCivil = 32
catTipoServicio = 33
catTipoImpuesto = 34
CatOrdenPagoCuentaImpuesto = 35
catTipoPago = 36
catStatusTraspasos = 38
catMovimientosBitacora = 39
catParametrosUniverse = 40
catEstados = 43
catCiudades = 44
catMunicipios = 45
catColonias = 46
catTipoRubro = 50
catRubroTipoRubro = 51

catBancos = 52
catPlaza = 53
catSucursal = 54
catOP_Banca = 55
catCuentasPagadoras = 56
catFiltros = 57
catBancosRubro = 58
catCtasBancariasRubro = 59
catGuiasAutorizadasCtroCostos = 60
catGuiaMonedaTipoPersona = 61
catGuiaRubros = 62
catPlantillasConceptos = 63
catInfBancaria = 64
catLotes = 65
catRangos = 66
catSistBanca = 67
catTipoCuenta = 68
catReportes = 69
catOficina_Impr = 70
catNivelConsulta = 71

End Enum

Public Function blnCreaDato(ByVal plngEmpleado As Long, ByVal catCatalogo As enmCatalogos, ByVal pstrDato1 As String, Optional ByVal pstrDato2 As String, Optional ByVal pstrDato3 As String, Optional ByVal pstrDato4 As String, Optional ByVal pstrDato5 As String, Optional ByVal pstrDato6 As String, Optional ByVal pstrDato7 As String, Optional ByVal pstrDato8 As String, Optional ByVal pstrDato9 As String) As Boolean

Descripción: función que guarda un dato determinado en una tabla especifica

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnCreaDato"


    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso
    Dim lngID As Long
    Dim dtmFecha As Date

On Error GoTo errHandler


mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)


    strSQL = ""
    mstrConcepto = ""

    Select Case catCatalogo
       Case 1      'Rubros

          'Nuevo Rubro
          Set objBD = New SIFIBD.BDAcceso
          'Verifica que el rubro no este ya dado de alta para el mismo concepto
          If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_DESCRIPCION, _
                              GSTR_TBL_RUBRO, _
                              pstrDato1, _
                              GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, _
                              pstrDato2) Then

             'Valido que no exista la cuenta contable
             If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE, _
                                 GSTR_TBL_RUBRO, _
                                 pstrDato3) Then

                'Valido que no exista la cuenta bancaria
                If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA, _
                                    GSTR_TBL_RUBRO, _
                                    pstrDato6) Then

                   lngID = objBD.lngGetNewId(GSTR_TBL_RUBRO, GSTR_TBL_RUBRO_FLD_ID)
                   strSQL = "INSERT INTO " & GSTR_TBL_RUBRO & vbCr
                   strSQL = strSQL & "(" & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "," & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_MONTO_MAX & "," & vbCr
                   strSQL = strSQL & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
                   'Agregadas 15-Mayo-2004
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_MONEDA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_BANCO & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_TIPOCUENTA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & ")"

                   strSQL = strSQL & " VALUES (" & vbCr
                   strSQL = strSQL & lngID & ", " & vbCr
```

```
                strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "', " & vbCr
                strSQL = strSQL & pstrDato3 & ", " & vbCr
                strSQL = strSQL & CLng(pstrDato2) & "," & vbCr

                If Trim$(pstrDato4) <> vbNullString Then
                   strSQL = strSQL & CDbl(pstrDato4) & "," & vbCr
                Else
                   strSQL = strSQL & "NULL," & vbCr
                End If

                strSQL = strSQL & CLng(pstrDato5) & "," & vbCr

                '15-Mayo-2004
                If Trim$(pstrDato6) <> vbNullString Then   'Cuenta Bancaria
                   strSQL = strSQL & "'" & UCase$(Trim$(pstrDato6)) & "', " & vbCr
                Else
                   strSQL = strSQL & "NULL," & vbCr
                End If

                If Trim$(pstrDato7) <> vbNullString Then  'IDMoneda
                   strSQL = strSQL & CDbl(pstrDato7) & ", "
                Else
                   strSQL = strSQL & "NULL" & ", "
                End If

                'ID banco
                strSQL = strSQL & CDbl(pstrDato8) & ", "

                'IDTipoCuenta
                strSQL = strSQL & CDbl(pstrDato9) & ", "

                'ID_Activo: cuando se crea está con estatus Activo: 1
                strSQL = strSQL & "1)"

             End If

          End If

       End If

       mstrConcepto = GSTR_TBL_RUBRO_FLD_DESCRIPCION & "=  " & UCase$(Trim$(pstrDato1)) & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & "= " & pstrDato3 & " " &
                      vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_MONTO_MAX & "= " & pstrDato4 & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & "=  " & pstrDato6 & " " &
                      vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_ID_MONEDA & "= " & pstrDato7 & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_ID_BANCO & "= " & pstrDato8

       Call RegistrarBitacora(GSTR_TBL_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
         mlngIdOficina)

    Case 9
       dtmFecha = CDate(GetDateFromServer)

       strSQL = "INSERT INTO " & GSTR_TBL_TIPO_CAMBIO & vbCr
       strSQL = strSQL & " (" & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_MES & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ID & ") " & vbCr
       strSQL = strSQL & " VALUES (" & vbCr

       strSQL = strSQL & Year(dtmFecha) & ", " & vbCr
       strSQL = strSQL & Month(dtmFecha) & ", " & vbCr
       strSQL = strSQL & Format(CDbl(pstrDato1), "##.0000") & ", " & vbCr
       strSQL = strSQL & CDbl(pstrDato2) & ") " & vbCr
```

```
    Set objBD = New SIFIBD.BDAcceso

Case 11     'Catálogo de áreas

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de areas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_AREA_FLD_NOMBRE, _
                            GSTR_TBL_AREA, _
                            pstrDato1) = False Then
        lngID = objBD.lngGetNewId(GSTR_TBL_AREA, GSTR_TBL_AREA_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_AREA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_AREA_FLD_NOMBRE & ")" & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_AREA_FLD_NOMBRE & "= " & pstrDato1

        Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    End If

Case 15 'Pantalla

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de areas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_PANTALLA_FLD_NOMBRE, _
                            GSTR_TBL_PANTALLA, _
                            pstrDato1) = False Then
        lngID = objBD.lngGetNewId(GSTR_TBL_PANTALLA, GSTR_TBL_PANTALLA_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_PANTALLA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_PANTALLA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PANTALLA_FLD_NOMBRE & ")" & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_PANTALLA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1))

        Call RegistrarBitacora(GSTR_TBL_PANTALLA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    End If

        'strSQL = "select " & GSTR_TBL_PANTALLA_FLD_ID & ", " & _
                    GSTR_TBL_PANTALLA_FLD_NOMBRE & " from " & _
                    GSTR_TBL_PANTALLA & " Order by " & _
                    GSTR_TBL_PANTALLA_FLD_NOMBRE

Case 23     'Catálogo de Tipos de Consulta de información

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso
```

```
    lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_CONSULTA, GSTR_TBL_TIPO_CONSULTA_FLD_ID)

    strSQL = "INSERT INTO " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & ")" & vbCr

    strSQL = strSQL & " VALUES (" & vbCr
    strSQL = strSQL & lngID & ", " & vbCr
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

    mstrConcepto = GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1))

    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
      mstrConcepto, mlngIdOficina)

Case 24     'Catálogo de Conceptos de Grupo


    Set objBD = New SIFIBD.BDAcceso

    If objBD.blnBuscaRegistro(pstrDato2, GSTR_TBL_CONCEPTO_RUBRO,
      GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, True) Then
        blnCreaDato = False
        Exit Function
    End If

    If objBD.blnBuscaRegistro(pstrDato1, GSTR_TBL_CONCEPTO_RUBRO,
      GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO, False) Then
        blnCreaDato = False
        Exit Function
    Else
        'Obtengo el consecutivo del ID
        lngID = objBD.lngGetNewId(GSTR_TBL_CONCEPTO_RUBRO,
                GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE)

        strSQL = "INSERT INTO " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
        strSQL = strSQL & " (" & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & ") " & vbCr
        strSQL = strSQL & " VALUES (" & lngID & ", " & vbCr
        strSQL = strSQL & Val(pstrDato2) & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "', " & vbCr
        strSQL = strSQL & "'" & pstrDato3 & "')"

        mstrConcepto = GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "= " & pstrDato2 & " " & vbCr
        mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "= " &
                       UCase$(Trim$(pstrDato1)) & " " & vbCr
        mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & "= " &
                       pstrDato3

        Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                mstrConcepto, mlngIdOficina)

    End If

Case 25     'RolRubro
    strSQL = "INSERT INTO ROL_PLANTILLA (ID_GUI_CON,IDROL) " & vbCr
    strSQL = strSQL & " VALUES (" & pstrDato1 & ", " & CLng(pstrDato2) & ")"

    mstrConcepto = GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "= " & pstrDato1 & " "
                       & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_ROL_FLD_ID & "= " & CLng(pstrDato2)
    Call RegistrarBitacora("ROL_PLANTILLA", plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    Set objBD = New SIFIBD.BDAcceso
```

```
Case 26     'rol - Oficina
    strSQL = "INSERT INTO " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_OFICINA_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES (" & CLng(pstrDato1) & ", " & vbCr
    strSQL = strSQL & CLng(pstrDato2) & ")"


    mstrConcepto = GSTR_TBL_ROL_FLD_ID & "= " & CLng(pstrDato1) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_OFICINA_FLD_ID & "= " & CLng(pstrDato2)

    Call RegistrarBitacora(GSTR_TBL_ROL_OFICINAS, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
      mstrConcepto, mlngIdOficina)

    Set objBD = New SIFIBD.BDAcceso

Case 28     'Oficina - Impuesto
    strSQL = "INSERT INTO " & GSTR_TBL_OFICINA_IMPUESTO & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_OFICINA_IMPUESTO_FLD_VALOR & ") " & vbCr

    'Valores
    strSQL = strSQL & " VALUES (" & vbCr
    'IDOficina
    strSQL = strSQL & pstrDato1 & ", " & vbCr
    'IDImpuesto
    strSQL = strSQL & pstrDato2 & ", " & vbCr
    'Valor
    strSQL = strSQL & pstrDato3 & ") "



    Call RegistrarBitacora(GSTR_TBL_OFICINA_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)


    Set objBD = New SIFIBD.BDAcceso

Case 31     'Proveedor telefono
    Set objBD = New SIFIBD.BDAcceso

    lngID = objBD.lngGetNewId(GSTR_TBL_PRESTADOR_TEL, GSTR_TBL_PRESTADOR_TEL_FLD_ID)

    strSQL = "INSERT INTO " & GSTR_TBL_PRESTADOR_TEL & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_PRESTADOR_TEL_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_TIPO_TELEFONO_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_CLAVE & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_NUMERO & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_EXTENSION & ") "

    strSQL = strSQL & " VALUES (" & vbCr
    'ID
    strSQL = strSQL & lngID & ", "
    'Id Tipo Telefono
    strSQL = strSQL & CLng(pstrDato1) & ", "
    'Id Prestador
    strSQL = strSQL & CLng(pstrDato2) & ", "
    'clave
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato3)) & "',"
    'Número
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato4)) & "',"
    'extensión
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato5)) & "')"
```

```
    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "= " & CLng(pstrDato2) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_CLAVE & "= " &
                        UCase$(Trim$(pstrDato3)) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_NUMERO & "= " &
                        UCase$(Trim$(pstrDato4)) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_EXTENSION & "= " &
                        UCase$(Trim$(pstrDato5))



    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_TEL, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                        mstrConcepto, mlngIdOficina)

Case 34     'Impuestos
    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_TIPO_IMPUESTO_FLD_NOMBRE, _
                        GSTR_TBL_TIPO_IMPUESTO, _
                        pstrDato1) Then

        lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_IMPUESTO, GSTR_TBL_TIPO_IMPUESTO_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_TIPO_IMPUESTO & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_CUENTA & ") " & vbCr
        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato1)) & "'," & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato2)) & "')"


        Call RegistrarBitacora(GSTR_TBL_TIPO_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)

    End If

Case 36     'Tipo Pago
    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION, _
                        GSTR_TBL_TIPO_PAGO, _
                        pstrDato1) Then

        lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_PAGO, GSTR_TBL_TIPO_PAGO_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_TIPO_PAGO & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION & ")" & vbCr

        'Valores
        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION & "= " & Trim$(UCase$(pstrDato1))

        Call RegistrarBitacora(GSTR_TBL_TIPO_PAGO, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                mstrConcepto, mlngIdOficina)

    End If

Case 51     'Rubro - Tipo Rubro
    strSQL = "INSERT INTO " & GSTR_TBL_RUBRO_TIPO_RUBRO & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_RUBRO_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES " & vbCr
    strSQL = strSQL & "(" & pstrDato1 & ", " & vbCr
```

```
    strSQL = strSQL & pstrDato2 & ")" & vbCr

    Call RegistrarBitacora(GSTR_TBL_RUBRO_TIPO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)

    Set objBD = New SIFIBD.BDAcceso

Case 52 'Catálogo de Bancos
    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de Bancos
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_BANCOS_FLD_DESCRIPCION, _
                            GSTR_TBL_BANCOS, _
                            pstrDato1, GSTR_TBL_BANCOS_FLD_ACTIVO, 1) Then
        lngID = objBD.lngGetNewId(GSTR_TBL_BANCOS, GSTR_TBL_BANCOS_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_BANCOS & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_BANCOS_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_DESCRIPCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_ID_IMPRESION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_IDBMR & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_ACTIVO & ") " & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "',"
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato2)) & "',"
        strSQL = strSQL & UCase$(Trim$(pstrDato3)) & ", "
        strSQL = strSQL & "1)"


        mstrConcepto = GSTR_TBL_BANCOS_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1))
        Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                                mlngIdOficina)
    End If

Case 53 'Catálogo de Plazas
    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de Plazas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_PLAZA_FLD_CVEBANCO, _
                            GSTR_TBL_PLAZA, _
                            pstrDato4, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, _
                            pstrDato2, GSTR_TBL_PLAZA_FLD_ID_PLAZA, pstrDato3,
                             GSTR_TBL_PLAZA_FLD_ACTIVO, 1) Then

        blnCreaDato = False
        Exit Function
    End If

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_PLAZA_FLD_DESCRIPCION, _
                            GSTR_TBL_PLAZA, _
                            pstrDato1, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, _
                            pstrDato2, GSTR_TBL_PLAZA_FLD_ACTIVO, 1) Then
        lngID = objBD.lngGetNewId(GSTR_TBL_PLAZA, GSTR_TBL_PLAZA_FLD_ID_PLAZA, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, pstrDato2)

        strSQL = "INSERT INTO " & GSTR_TBL_PLAZA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_PLAZA_FLD_ID_BANCO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_ID_PLAZA & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_DESCRIPCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_CVEBANCO & ", " & vbCr
```

```
            strSQL = strSQL & GSTR_TBL_PLAZA_FLD_CVEBANCO_BMR & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_PLAZA_FLD_ACTIVO & ") " & vbCr
            strSQL = strSQL & " VALUES (" & vbCr
            strSQL = strSQL & Val(pstrDato2) & ", " & vbCr
            strSQL = strSQL & lngID & ", " & vbCr
            strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "'," & vbCr
            strSQL = strSQL & pstrDato4 & ","
            strSQL = strSQL & pstrDato5 & ",1)"

            mstrConcepto = GSTR_TBL_PLAZA_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1)) & " " & vbCr
            mstrConcepto = mstrConcepto & GSTR_TBL_PLAZA_FLD_CVEBANCO & "= " & pstrDato4

            Call RegistrarBitacora(GSTR_TBL_PLAZA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                    mlngIdOficina)
        End If



errHandler:
    Select Case Err.Number

        Case 0

        Case Else
            Call mobjError.RegistraError(Err.Number, Err.Description, STR_FUNCION)
            blnCreaDato = False
            Err.Clear
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnEliminaDato(ByVal plngEmpleado As Long, ByVal enmCatalogo As enmCatalogos, ByVal plngID As Long, Optional ByVal plngValor As Long, Optional ByVal plngValor2 As Long, Optional ByVal pstrDato1 As String) As Boolean

Descripción: Función para eliminar un registro en una tabla determinada

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnEliminaDato"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String

On Error GoTo errHandler
mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)



    Select Case enmCatalogo
        Case 1  'Rubro
            strSQL = "UPDATE " & GSTR_TBL_RUBRO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=0"
            strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO_FLD_ID & "=" & plngID

            mstrConcepto = GSTR_TBL_RUBRO_FLD_ID & "=" & plngID

            Call RegistrarBitacora(GSTR_TBL_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                    mlngIdOficina)

        Case 10     'Proveedores

            strSQL = " DELETE " & GSTR_TBL_PRESTADOR_TEL & vbCrLf
            strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID & vbCrLf
```

```
    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)


    strSQL = " DELETE " & GSTR_TBL_PRESTADOR_DOMICILIO & vbCrLf
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & vbCrLf

    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)


    strSQL = " DELETE " & GSTR_TBL_PRESTADOR & vbCrLf
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & vbCrLf


    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_PRESTADOR, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)

Case 11     'Areas
    strSQL = "DELETE " & GSTR_TBL_AREA & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_AREA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_AREA_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                        mlngIdOficina)

Case 23     'Tipo Consulta
    strSQL = "DELETE " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)

Case 24     'Concepto del Rubro

    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnBuscaRegistro(plngID, GSTR_TBL_RUBRO, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, True)
      Then


        strSQL = "DELETE " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
        strSQL = strSQL & " WHERE " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & plngID

        mstrConcepto = GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & plngID
        Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)
    Else
            blnEliminaDato = False
            Exit Function
    End If

Case 25     'Rol - Rubro
    strSQL = "DELETE ROL_PLANTILLA " & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "=" &
                plngID
    strSQL = strSQL & " AND " & GSTR_TBL_ROL_FLD_ID & "=" & plngValor

    mstrConcepto = GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "=" & plngID & " " &
                        vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_ROL_FLD_ID & "=" & plngValor
```

```
    Call RegistrarBitacora("ROL_PLANTILLA", plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                        mlngIdOficina)

Case 26     'Rol - Oficinas
    strSQL = "DELETE " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_OFICINA_FLD_ID & "=" & plngValor

    mstrConcepto = GSTR_TBL_ROL_FLD_ID & "=" & plngID
    mstrConcepto = mstrConcepto & GSTR_TBL_OFICINA_FLD_ID & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_ROL_OFICINAS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                        mstrConcepto, mlngIdOficina)

Case 28     'Oficina - Impuesto
    strSQL = "DELETE " & GSTR_TBL_OFICINA_IMPUESTO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_OFICINA_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & "=" & plngValor



    Call RegistrarBitacora(GSTR_TBL_OFICINA_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case 30     'Prestador - Domicilio
    strSQL = "DELETE " & GSTR_TBL_PRESTADOR_DOMICILIO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_DOMICILIO_FLD_ID & "=" & plngValor


    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_DOMICILIO_FLD_ID & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_DOMICILIO, plngEmpleado,
                                MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto, mlngIdOficina)

Case 31     'Prestador - Telefono
    strSQL = "DELETE " & GSTR_TBL_PRESTADOR_TEL & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_TEL, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 34     'Tipo Impuesto
    strSQL = "DELETE " & GSTR_TBL_TIPO_IMPUESTO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_TIPO_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case 36     'Tipo Pago
    strSQL = "DELETE " & GSTR_TBL_TIPO_PAGO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_PAGO_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_TIPO_PAGO_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_TIPO_PAGO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 51     'Rubro - Tipo Rubro
    strSQL = "DELETE " & GSTR_TBL_RUBRO_TIPO_RUBRO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO_FLD_ID & "=" & plngID & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_RUBRO_FLD_ID & "=" & plngValor & vbCr


    mstrConcepto = GSTR_TBL_RUBRO_FLD_ID & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_RUBRO_FLD_ID & "=" & plngValor & vbCr
```

```
    Call RegistrarBitacora(GSTR_TBL_RUBRO_TIPO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                           mstrConcepto, mlngIdOficina)

Case 52     'Bancos_TE

    strSQL = "UPDATE " & GSTR_TBL_BANCOS & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_BANCOS_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_BANCOS_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_BANCOS_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                           mlngIdOficina)

Case 53     'Plazas
    strSQL = "UPDATE " & GSTR_TBL_PLAZA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_PLAZA_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_PLAZA_FLD_ID_PLAZA & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_PLAZA_FLD_ID_BANCO & "=" & plngValor


     mstrConcepto = GSTR_TBL_PLAZA_FLD_ID_PLAZA & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_PLAZA_FLD_ID_BANCO & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                           mlngIdOficina)

Case 54     'Sucursal
    strSQL = "UPDATE " & GSTR_TBL_SUCURSAL & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_SUCURSAL_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_SUCURSAL_FLD_ID_BANCO & "=" & plngValor
    strSQL = strSQL & " AND " & GSTR_TBL_SUCURSAL_FLD_ID_PLAZA & "=" & plngValor2

    mstrConcepto = GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_SUCURSAL_FLD_ID_BANCO & "=" & plngValor & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_SUCURSAL_FLD_ID_PLAZA & "=" & plngValor2

    Call RegistrarBitacora(GSTR_TBL_SUCURSAL, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                           mstrConcepto, mlngIdOficina)


Case 55     'Operación Bancaria
    strSQL = "UPDATE " & GSTR_TBL_OP_BANCA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_OP_BANCA_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_OP_BANCA_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_OP_BANCA_FLD_ID_BANCO & "=" & plngValor

    mstrConcepto = GSTR_TBL_OP_BANCA_FLD_ID & "=" & plngID & " "
    mstrConcepto = GSTR_TBL_OP_BANCA_FLD_ID_BANCO & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_OP_BANCA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 56  'Cuentas Bancarias


Case 57  'Filtros de búsquedas
    strSQL = "DELETE " & GSTR_TBL_FIL_OP & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_FIL_OP_FLD_IDR_FILTRO & "=" & plngID


    Call RegistrarBitacora(GSTR_TBL_FIL_OP, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case Else
    mobjError.RegistraError 203, GSTR_ERROR_203, STR_FUNCION
```

```
    End Select


    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnEliminaDato = False
            Err.Clear
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnModificaDato(ByVal plngEmpleado As Long, ByVal catCatalogo As enmCatalogos, ByVal plngID As Long, ByVal pstrDato1 As String, Optional ByVal pstrDato2 As String, Optional ByVal pstrDato3 As String, Optional ByVal pstrDato4 As String, Optional ByVal pstrDato5 As String, Optional ByVal pstrDato6 As String, Optional ByVal pstrDato7 As String, Optional ByVal pstrDato8 As String, Optional ByVal pstrDato9 As String) As Boolean

Descripción: función general para eliminar un registro de la Base de datos

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnModificaDato"

    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler
mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)



    Select Case catCatalogo

        Case 11     'Areas
            strSQL = "UPDATE " & GSTR_TBL_AREA & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_AREA_FLD_NOMBRE & "='" & UCase$(Trim$(pstrDato1)) & "'" &
                     vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_AREA_FLD_ID & "=" & plngID

            mstrConcepto = strSQL = strSQL & " SET " & GSTR_TBL_AREA_FLD_NOMBRE & "= " &
                           UCase$(Trim$(pstrDato1))
            Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR, mstrConcepto,
                                   mlngIdOficina)

        Case 13 'Tipo Objeto
            strSQL = "UPDATE " & GSTR_TBL_TIPO_OBJETO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION & "='" &
                     UCase$(Trim$(pstrDato1)) & "'" & vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_OBJETO_FLD_ID & "=" & plngID

            mstrConcepto = GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1))
            Call RegistrarBitacora(GSTR_TBL_TIPO_OBJETO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                           mstrConcepto, mlngIdOficina)

        Case 14 'Rol
```

```
    strSQL = "UPDATE " & GSTR_TBL_ROL & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_ROL_FLD_DESCRIPCION & "='" & UCase$(Trim$(pstrDato1)) & "'" &
                        vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_CANTIDAD & "=" & CDbl(pstrDato2) & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_ROL_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1)) & " "
    mstrConcepto = GSTR_TBL_ROL_FLD_CANTIDAD & "= " & CDbl(pstrDato2)

    Call RegistrarBitacora(GSTR_TBL_ROL, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR, mstrConcepto,
                        mlngIdOficina)

Case 15 'Pantalla
    strSQL = "UPDATE " & GSTR_TBL_PANTALLA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_PANTALLA_FLD_NOMBRE & "='" & UCase$(Trim$(pstrDato1)) & "'"
                & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PANTALLA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_PANTALLA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1)) & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_PANTALLA_FLD_ID & "= " & plngID

    Call RegistrarBitacora(GSTR_TBL_PANTALLA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                                    mstrConcepto, mlngIdOficina)

Case 17 'Objeto
    strSQL = "UPDATE " & GSTR_TBL_OBJETO
    strSQL = strSQL & " SET " & GSTR_TBL_OBJETO_FLD_NOMBRE & " = '" & UCase$(Trim$(pstrDato1)) & "' " &
                        vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_OBJETO_FLD_ID & " = " & plngID & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_PANTALLA_FLD_ID & " = " & CLng(pstrDato3) & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_OBJETO_FLD_ID & " = " & CLng(pstrDato2)


    mstrConcepto = GSTR_TBL_OBJETO_FLD_NOMBRE & " = " & UCase$(Trim$(pstrDato1)) & " " & vbCr

    Call RegistrarBitacora(GSTR_TBL_OBJETO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                        mstrConcepto, mlngIdOficina)

Case 23     'Modifica Tipo Consulta
    strSQL = "UPDATE " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "='" &
                UCase$(Trim$(pstrDato1)) & "'" & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID

    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "= " &
                        UCase$(Trim$(pstrDato1))

    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                                    mstrConcepto, mlngIdOficina)

Case 24     'Concepto del Rubro

    Set objBD = New SIFIBD.BDAcceso

     If Not objBD.blnExisteDatoenTabla(GSTR_TBL_CONCEPTO_RUBRO_FLD_ID,
                GSTR_TBL_CONCEPTO_RUBRO, pstrDato1, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE,
     plngID) Then

     If objBD.blnBuscaRegistro(pstrDato1, GSTR_TBL_CONCEPTO_RUBRO,
     GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, False) Then
         blnModificaDato = False
         Exit Function
       End If
    End If
```

```
            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO,
            GSTR_TBL_CONCEPTO_RUBRO, pstrDato2, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE, plngID)
            Then

            If objBD.blnBuscaRegistro(pstrDato2, GSTR_TBL_CONCEPTO_RUBRO,
            GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO, False) Then
                blnModificaDato = False
                Exit Function
              End If
          End If

          strSQL = "UPDATE " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "='" &
            UCase$(Trim$(pstrDato2)) & "'" & vbCr
          strSQL = strSQL & ", " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & pstrDato1 & vbCr
          strSQL = strSQL & " WHERE " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & "=" & plngID

            mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "= " &
            UCase$(Trim$(pstrDato2)) & " " & vbCr
          mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "= " & pstrDato1 & vbCr

            Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
            mstrConcepto, mlngIdOficina)



        Case Else
            mobjError.RegistraError 202, GSTR_ERROR_201, STR_FUNCION
            blnModificaDato = False
            Exit Function
    End Select

    If objBD Is Nothing Then
        Set objBD = New SIFIBD.BDAcceso
    End If

    If Trim$(strSQL) <> vbNullString Then
        blnModificaDato = objBD.blnEjecutaQuery(strSQL)
    End If

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnModificaDato = False
            Err.Clear
    End Select

End Function
```

Public Function LoadCatalogo(ByVal catCatalogo As enmCatalogos, _
Optional ByVal plngIdDato1 As Long = 0, _
Optional ByVal plngIdDato2 As Long = 0, _
Optional ByVal plngIdDato3 As Long = 0, _
Optional ByVal plngIdDato4 As Long = 0, _
Optional ByVal plngIdDato5 As Long = 0, _
Optional ByVal plngIdDato6 As Long = 0, _
Optional ByVal plngIdDato7 As Long = 0, _
Optional ByVal plngIDDato8 As Long) As ADODB.Recordset

Descripción: Función General para cargar la información de un catálogo específico en un recordset y

```
                poder deplegarlo en un objeto

    Const STR_FUNCION As String = MSTR_MODULO & ".LoadCatalogo"

    Const LNG_ACTIVO As Long = 1

    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler

    Select Case catCatalogo

        Case 0 'Tipo Telefono
            strSQL = "SELECT " & GSTR_TBL_TIPO_TELEFONO_FLD_ID & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_TIPO_TELEFONO_FLD_TIPO & vbCr
            strSQL = strSQL & " FROM " & GSTR_TBL_TIPO_TELEFONO & vbCr
            strSQL = strSQL & " ORDER BY " & GSTR_TBL_TIPO_TELEFONO_FLD_TIPO

        Case 1 'Rubro
            strSQL = "SELECT " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_MONTO_MAX & ", " & vbCr
              strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ",
              " & vbCr
            strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." &
            GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr


            strSQL = strSQL & " CONVERT(bigint," & GSTR_TBL_RUBRO & "." &
            GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ") AS " &
            GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
              strSQL = strSQL & GSTR_TBL_NATURALEZA & "." & GSTR_TBL_NATURALEZA_DESCRIPCION & " AS " &
              GSTR_TBL_NATURALEZA & ", " & vbCr

            '15-Mayo-2004
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_TIPOCUENTA & "," & vbCr

            strSQL = strSQL & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr

            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO & ", " & vbCr

              strSQL = strSQL & GSTR_TBL_BANCOS & "." & GSTR_TBL_BANCOS_FLD_DESCRIPCION & " AS " &
              GSTR_TBL_RUBRO_FLD_ID_NOMBRE_BANCO & vbCr


            strSQL = strSQL & " FROM " & GSTR_TBL_RUBRO & vbCr

            'Concepto del rubro
            strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
              strSQL = strSQL & "=" & GSTR_TBL_CONCEPTO_RUBRO & "." &
              GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & vbCr

            'Naturaleza del Rubro
            strSQL = strSQL & " INNER JOIN " & GSTR_TBL_NATURALEZA & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID
            strSQL = strSQL & "=" & GSTR_TBL_NATURALEZA & "." & GSTR_TBL_NATURALEZA_FLD_ID & vbCr

            'Bancos
            strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_BANCOS & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
            strSQL = strSQL & " = " & GSTR_TBL_BANCOS & "." & GSTR_TBL_BANCOS_FLD_ID & vbCr
```

```
    'Moneda
    strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_MONEDA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
    strSQL = strSQL & "=" & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr



    If plngIdDato1 > 0 Then
        strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ROL_RUBRO & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID

        strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_ROL_FLD_ID & "=" &
      plngIdDato1 & vbCr
        strSQL = strSQL & " AND " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "="
      & plngIdDato2 & vbCr
        '15-Mayo-2004 Solo muestra los activos (no eliminados lógicamente)
        strSQL = strSQL & " AND (" & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=1 OR
      " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & " IS NULL )" & vbCr
    Else
        '15-Mayo-2004 Solo muestra los activos (no eliminados lógicamente)
        strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=1
      OR " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & " IS NULL " & vbCr
    End If




    strSQL = strSQL & " ORDER BY " & GSTR_TBL_CONCEPTO_RUBRO & "." &
      GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION



Case 2
    strSQL = "SELECT " & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_NATURALEZA_DESCRIPCION & vbCr
    strSQL = strSQL & " FROM " & GSTR_TBL_NATURALEZA & vbCr
    strSQL = strSQL & "ORDER BY " & GSTR_TBL_NATURALEZA_DESCRIPCION

Case 3 'Empleados
    strSQL = "select " & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO_FLD_NOMBRE & vbCr
    strSQL = strSQL & " + ' ' + " & GSTR_TBL_EMPLEADO_FLD_PATERNO & vbCr
    strSQL = strSQL & " + ' ' + " & GSTR_TBL_EMPLEADO_FLD_MATERNO & " as " &
      GSTR_TBL_EMPLEADO_FLD_NOMBRE & vbCr
    strSQL = strSQL & " from " & GSTR_TBL_EMPLEADO & vbCr
    strSQL = strSQL & " where " & GSTR_TBL_EMPLEADO_FLD_ACTIVO & " = " & LNG_ACTIVO & vbCr
    strSQL = strSQL & " order by " & GSTR_TBL_EMPLEADO_FLD_NOMBRE

Case 4 'Centro de Costos

    If plngIdDato1 = 0 Then
        strSQL = "SELECT " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
      & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_TIPO_DISTRIBUCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_EMPLEADOS & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_METROS_CUADRADOS & ", " & vbCr
```

```
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
    GSTR_TBL_AREA & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & vbCr
      strSQL = strSQL & " FROM " & GSTR_TBL_CENTRO_COSTO & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
      strSQL = strSQL & " WHERE " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " = 1 " & vbCr
      strSQL = strSQL & " ORDER BY " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & ",
    " & vbCr
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO
  Else
      strSQL = "SELECT " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
    & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_TIPO_DISTRIBUCION & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_EMPLEADOS & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_METROS_CUADRADOS & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
    GSTR_TBL_AREA & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & vbCr
      strSQL = strSQL & " FROM " & GSTR_TBL_CENTRO_COSTO & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
      strSQL = strSQL & " WHERE " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " <> 0 " & vbCr
      strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " <> 3 " & vbCr
      strSQL = strSQL & " ORDER BY " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & ",
    " & vbCr
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO
  End If

Case 5 'Departamentos
  strSQL = "select " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & ", " & vbCr
  strSQL = strSQL & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr
  strSQL = strSQL & " from " & GSTR_TBL_CENTRO_COSTO & vbCr
  strSQL = strSQL & " Where " & GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " = " & LNG_ACTIVO & vbCr
  strSQL = strSQL & " ORDER BY " & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO

Case 6 'Oficinas
  If plngIdDato1 > 0 Then
    strSQL = "select " & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & " from " & GSTR_TBL_OFICINA
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO
    strSQL = strSQL & " where " & GSTR_TBL_OFICINA_FLD_ID & " = " & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & "and " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIdDato1
  Else
```

```
        strSQL = "select " & GSTR_TBL_OFICINA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_OFICINA
        strSQL = strSQL & " Order by " & GSTR_TBL_OFICINA_FLD_NOMBRE
    End If



Case 8 'Monedas
    If plngIdDato1 > 0 Then
        strSQL = "select " & GSTR_TBL_MONEDA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_SIGNO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ABREV & vbCr
        strSQL = strSQL & " from " & GSTR_TBL_MONEDA & vbCr
        strSQL = strSQL & " where " & GSTR_TBL_MONEDA_FLD_ID & " = " & plngIdDato1 & vbCr
        strSQL = strSQL & " order by " & GSTR_TBL_MONEDA_FLD_NOMBRE
    Else
        strSQL = "select " & GSTR_TBL_MONEDA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_SIGNO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ABREV & vbCr
        strSQL = strSQL & " from " & GSTR_TBL_MONEDA & vbCr
        strSQL = strSQL & " order by " & GSTR_TBL_MONEDA_FLD_NOMBRE
    End If


Case 9 'Tipo de Cambio
    If plngIdDato1 >= 0 And plngIdDato2 > 0 And plngIdDato3 > 0 Then
        strSQL = "select " & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_MES
        strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_CAMBIO
        strSQL = strSQL & " where " & GSTR_TBL_MONEDA_FLD_ID & " = " & plngIdDato1
        strSQL = strSQL & " AND " & GSTR_TBL_TIPO_CAMBIO_FLD_MES & " = " & plngIdDato2
        strSQL = strSQL & " AND " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO & " = " & plngIdDato3
    Else
        strSQL = "select " & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_MES
        strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_CAMBIO

    End If

Case 10 'Prestadores
    strSQL = "SELECT " & GSTR_TBL_PRESTADOR_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_MATERNO & ", " & vbCr

    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & " + ' ' + " & vbCr
    strSQL = strSQL & " ISNULL(" & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", ' ' ) + ' ' + " & vbCr
    strSQL = strSQL & " ISNULL(" & GSTR_TBL_PRESTADOR_FLD_MATERNO & ", ' ') as  " &
      GSTR_TBL_PRESTADOR & ", " & vbCr

    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_FISICA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_RFC & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CURP & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_SEXO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_FECHA_NAC & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_SERVICIO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EDO_CIVIL_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDBANCO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDPLAZA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDSUCURSAL & ", " & vbCr
```

```
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CTA_BANCARIA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CTA_CLABE & "," & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & "," & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & vbCr



    strSQL = strSQL & " FROM " & GSTR_TBL_PRESTADOR & vbCr

    If plngIdDato1 > 0 Then
       strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngIdDato1 & vbCr
       If plngIdDato2 > 0 Then _
          strSQL = strSQL & " AND NOT " & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & " Is Null" &
     vbCr
    Else
       If plngIdDato2 > 0 Then _
          strSQL = strSQL & " WHERE NOT " & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & " Is Null"
     & vbCr
    End If

    strSQL = strSQL & " ORDER BY " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

Case 11 'Areas
    strSQL = "select " & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & " from " & GSTR_TBL_AREA
    strSQL = strSQL & " Order by " & GSTR_TBL_AREA_FLD_NOMBRE

Case 12 'Oficinas-Areas
        strSQL = "SELECT " & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & vbCr
        strSQL = strSQL & " FROM " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & vbCr
        strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
        strSQL = strSQL & " = " & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
        strSQL = strSQL & " ORDER BY" & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID

Case 13 'Tipo Objeto
        strSQL = "select " & GSTR_TBL_TIPO_OBJETO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_OBJETO
        strSQL = strSQL & " Order by " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION

Case 14 'Rol
        strSQL = "select " & GSTR_TBL_ROL_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_DESCRIPCION
        strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_CANTIDAD
        strSQL = strSQL & " from " & GSTR_TBL_ROL
        strSQL = strSQL & " Order by " & GSTR_TBL_ROL_FLD_DESCRIPCION

Case 15 'Pantalla
        strSQL = "select " & GSTR_TBL_PANTALLA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_PANTALLA_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_PANTALLA
        strSQL = strSQL & " Order by " & GSTR_TBL_PANTALLA_FLD_NOMBRE

Case 16 'Rol-Objeto
        strSQL = "SELECT " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_OBJETO_FLD_ID_PWD & ", " &
       vbCr
```

```
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_PANTALLA_FLD_ID & vbCr
        strSQL = strSQL & " FROM " & GSTR_TBL_ROL_OBJETO & vbCr
        strSQL = strSQL & " RIGHT JOIN " & GSTR_TBL_OBJETO & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & vbCr
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE

    Case 17 'Objeto
        strSQL = "select " & GSTR_TBL_OBJETO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_OBJETO_FLD_NOMBRE
        strSQL = strSQL & ", " & GSTR_TBL_PANTALLA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_OBJETO_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_OBJETO
        strSQL = strSQL & " Order by " & GSTR_TBL_OBJETO_FLD_NOMBRE

    Case 18 'Banco
        strSQL = "select " & GSTR_TBL_BANCO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_BANCO_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_BANCO
        strSQL = strSQL & " Order by " & GSTR_TBL_BANCO_FLD_NOMBRE

    Case 19 'Status_OP
        strSQL = "SELECT " & GSTR_TBL_STATUS_OP_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_DESCRIPCION
        strSQL = strSQL & " FROM " & GSTR_TBL_STATUS_OP
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_STATUS_OP_FLD_DESCRIPCION


    Case 20 'Orden_Status
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_STATUS_FLD_FECHA
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_STATUS
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 21 'Orden_Pago
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
        strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_RUBRO_FLD_ID
'       strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_DESCRIPCION
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
        strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_PAGO
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 22 'Orden_Pago_Cuenta
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        'strSQL = strSQL & ", " & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_GUIAS_AUT_CC_FLD_ID_GUI_CON
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_PAGO_CUENTA
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 23 'TipoConsulta
        strSQL = "SELECT " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & vbCr
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & vbCr
        strSQL = strSQL & " FROM " & GSTR_TBL_TIPO_CONSULTA & vbCr
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
```

```
        Case 24 'Concepto de rubro
           strSQL = "SELECT " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO
           strSQL = strSQL & " FROM " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
           strSQL = strSQL & " ORDER BY " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID

        Case 25     'Rol - Rubro
           strSQL = "SELECT " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
                    strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." &
             GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & vbCr
           strSQL = strSQL & " FROM " & GSTR_TBL_ROL_RUBRO & vbCr
           strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
           strSQL = strSQL & " ON " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
           strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
           strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
             strSQL = strSQL & " ON " & GSTR_TBL_CONCEPTO_RUBRO & "." &
             GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
           strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID



        Case Else
           mobjError.RegistraError 202, GSTR_ERROR_202, STR_FUNCION

     End Select

     'Cargo el recordset
     Set objBD = New SIFIBD.BDAcceso

     Set LoadCatalogo = objBD.rsdDevuelveRecordset(strSQL)

errHandler:
     Select Case Err.Number
        Case 0

        Case Else
           mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

     End Select

     'Libero la memoria
     Set objBD = Nothing

End Function
```

Public Sub RegistrarBitacora(ByVal pstrTabla As String, ByVal plngEmpleado As Long, ByVal plngMovimiento As Long, Optional ByVal pstrConcepto As String, Optional ByVal plngOficina As Long)

Descripción: Función para registrar los movimientos de la operación del sistema de un usuario determinado.

```
     Const STR_FUNCION As String = MSTR_MODULO & ".RegistrarBitacora"

     Dim strSQL As String
     Dim lngID As Long
     Dim strFecha As String
     Dim strHora As String
     Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler

     'strFecha = GetDateFromServer
     strFecha = Format(GetDateFromServer, "yyyy-mm-dd")
```

```
    strHora = Format(GetDateFromServer, "Hh:Mm")

    Set objBD = New SIFIBD.BDAcceso

    lngID = objBD.lngGetNewId(GSTR_TBL_BITACORA, GSTR_TBL_BITACORA_FLD_ID)


    strSQL = "INSERT INTO " & GSTR_TBL_BITACORA & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_BITACORA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_TABLA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_FECHA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_MOVIMIENTO_FLD_ID & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_HORA & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_CONCEPTO & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_IDOFICINA & ") " & vbCr

    strSQL = strSQL & " VALUES (" & vbCr
    strSQL = strSQL & lngID & ", " & vbCr
    strSQL = strSQL & "'" & UCase$(Trim$(pstrTabla)) & "', " & vbCr
    strSQL = strSQL & "'" & strFecha & "'," & vbCr
    strSQL = strSQL & plngEmpleado & ", " & vbCr
    strSQL = strSQL & plngMovimiento & ", " & vbCr
    strSQL = strSQL & "'" & strHora & "', " & vbCr
    strSQL = strSQL & "'" & pstrConcepto & "'," & vbCr
    strSQL = strSQL & plngOficina & ")" & vbCr


    Call objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select
'</CSCustomCode> 1

    'Libero la memoria
    Set objBD = Nothing

End Sub
```

CLASE DE ACCESO A DATOS

Public Function blnBuscaRegistro(ByVal pstrDato As String, ByVal pstrTabla As String, ByVal pstrCampo As String, ByVal pblnCampoNumerico As Boolean) As Boolean

Descripción: Función para buscar un registro dado un valor, una tabla y un campo determinado.
Regresa Verdadero en caso de lograr la busqueda. Falso si no la encuentra.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnBuscaRegistro"

    Dim strSQL As String
    Dim rsdRecordset As ADODB.Recordset

On Error GoTo errHandler

    strSQL = "SELECT * " & vbCr
    strSQL = strSQL & " FROM " & pstrTabla & vbCr

    If pblnCampoNumerico Then
        strSQL = strSQL & " WHERE " & pstrCampo & "=" & pstrDato
    Else
        strSQL = strSQL & " WHERE " & pstrCampo & "='" & pstrDato & "'"
    End If

    Set rsdRecordset = rsdDevuelveRecordset(strSQL)

    If rsdRecordset.RecordCount > 0 Then
        'si existe por lo menos un registro
        blnBuscaRegistro = True
    Else
        'No existe el registro
        blnBuscaRegistro = False
    End If

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnBuscaRegistro = False
    End Select

    'Libero la memoria
    Set rsdRecordset = Nothing

End Function
```

Public Function blnConectaBD() As Boolean

Descripción: Funcion que abre la conexion a la BD

```
Dim objLog      As xmsError.xmsManejaError

    On Error GoTo ErrorHandler

    'Si no se ha asignado la propiedad de la cadena de conexion
```

```
    'la asigna
    If STR_CONN_STRING <> vbNullString Then
        If adoConnection.State = False Then
            With adoConnection
                .CursorLocation = adUseServer
                .Open STR_CONN_STRING, mstrUser, mstrPassword
            End With
            blnConectaBD = True
        Else
            blnConectaBD = True
        End If
    Else
        blnConectaBD = True
    End If
ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnConectaBD = False

            Set objLog = New xmsError.xmsManejaError

            objLog.RegistraError 2, "No se pudo crear una conexion a la BD" &                    vbCrLf    &    ""    &
STR_CONN_STRING, "BDAcceso.blnConectaBD"

            Set objLog = Nothing

'           Resume Next
'           objErr.RegistraError Err.Number,  vbCrLf & Err.Description,
            "SIFIBD.blnConectaBD"
    End Select
End Function
```

Public Function blnEjecutaQuery(ByVal pstrSQL As String) As Boolean

Descripción: Funcion ejecuta una cadena SQL

```
Dim adoCmd     As ADODB.Command
Dim objLog     As xmsError.xmsManejaError

    On Error GoTo ErrorHandler

    If blnConectaBD Then
        Set adoCmd = New ADODB.Command

        With adoCmd
            .CommandText = pstrSQL
            .CommandType = adCmdText
            .ActiveConnection = adoConnection
            .Execute
        End With
        blnEjecutaQuery = True

        Set adoCmd = Nothing
    Else
        blnEjecutaQuery = False
    End If
```

```
ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         blnEjecutaQuery = False

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 3, "No se pudo ejecutar la consulta : " &                vbCrLf  &  pstrSQL,
"BDAcceso.blnEjecutaQuery"

         Set objLog = Nothing

   End Select
End Function
```

Public Function rsdDevuelveRecordset(ByVal pstrSQL As String) As ADODB.Recordset

Descripion: Funcion para ejecutar una cadena de conexion y devuelve un

```
Dim adoRecordset    As ADODB.Recordset
Dim objLog          As xmsError.xmsManejaError

   On Error GoTo ErrorHandler

   If blnConectaBD = True Then
      Set adoRecordset = New ADODB.Recordset
      With adoRecordset
         .CursorLocation = adUseClient
         .CursorType = adOpenStatic
         .LockType = adLockBatchOptimistic
         .Source = pstrSQL
         .ActiveConnection = adoConnection
         .Open
         .ActiveConnection = Nothing
      End With
      Set rsdDevuelveRecordset = adoRecordset

      Set adoRecordset = Nothing

   End If


ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set rsdDevuelveRecordset = Nothing

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 4, "No se pudo ejecutar la consulta : " &                vbCrLf  &  pstrSQL,
"BDAcceso.rsdDevuelveRecordset"

         Set objLog = Nothing

'          Resume Next
```

```
    End Select
End Function


Public Function lngGetNewId(ByVal pstrTable As String, ByVal pstrField As String, _
                    Optional ByVal pstrDatoFiltrar1 As String, _
                    Optional ByVal plngValorFiltrar1 As Long, _
                    Optional ByVal pstrDatoFiltrar2 As String, _
                    Optional ByVal plngValorFiltrar2 As Long) As Double
```

Descripción: Funcion para ejecutar una cadena de conexion y devuelve

```
Dim strSQL          As String
Dim adoRecordset    As ADODB.Recordset
Dim objLog          As xmsError.xmsManejaError

On Error GoTo ErrorHandler

    Set adoRecordset = New ADODB.Recordset


    strSQL = "select max(" & pstrField & ") from " & pstrTable

    'Campo1 a filtar y valor1 a considerar
    If plngValorFiltrar1 > 0 Then
        strSQL = strSQL & " where " & pstrDatoFiltrar1 & " = " &
            plngValorFiltrar1
    End If

    'Campo2 a filtar y valor2 a considerar
    If plngValorFiltrar2 > 0 Then
        strSQL = strSQL & " and " & pstrDatoFiltrar2 & " = " & plngValorFiltrar2
    End If

    Set adoRecordset = rsdDevuelveRecordset(strSQL)


    lngGetNewId = -1

    If Not adoRecordset.EOF And Not adoRecordset.BOF Then
        If IsNumeric(adoRecordset(0)) Then
            lngGetNewId = CDbl(adoRecordset(0)) + 1
        Else
            lngGetNewId = 1
        End If
    End If

    Set adoRecordset = Nothing

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set adoRecordset = Nothing

            Set objLog = New xmsError.xmsManejaError

            objLog.RegistraError 5, "No se pudo obtener un nuevo ID de la Tabla                    :    **    "    &
pstrTable & " **  del campo : ** " & pstrField & " **                                ", "BDAcceso.lngGetNewId"
```

```
        Set objLog = Nothing

'       objErr.RegistraError Err.Number,  vbCrLf & Err.Description,
        "SIFISeguridad.lngGetNewIdUsuario"
        lngGetNewId = -1
'       Resume Next
   End Select
End Function
```

Public Function blnDeleteId(ByVal pstrTable As String, ByVal pstrField As String, ByVal pintValue As Integer) As Boolean

Descripción: Funcion para ejecutar una cadena de conexion y elimina
' el ID de la tabla enviado

```
Dim strSQL      As String
Dim objLog      As xmsError.xmsManejaError

On Error GoTo ErrorHandler


   strSQL = "delete from " & pstrTable & " Where " & pstrField & " = " &  pintValue

   blnDeleteId = blnEjecutaQuery(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         blnDeleteId = False

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 5, "No se pudo Eliminar el Registro de la Tabla : ** " & pstrTable & "
          **  del campo : ** " & pstrField & " **  con valor = " & pintValue,
          "BDAcceso.lngGetNewId"

         Set objLog = Nothing

   End Select
End Function
```

Public Function blnExisteDatoenTabla(ByVal pstrFld As String, ByVal pstrTable As String, ByVal pstrValuetoSearch As String, Optional ByVal pstrFld2 As String = vbNullString, Optional ByVal pstrValuetoSearchFld2 As String = vbNullString, Optional ByVal pstrFld3 As String = vbNullString, Optional ByVal pstrValuetoSearchFld3 As String = vbNullString, Optional ByVal pstrFld4 As String = vbNullString, Optional ByVal pstrValuetoSearchFld4 As String = vbNullString) As Boolean

Descripción: Funcion para determinar si existe un dato

```
Const STR_FUNCION As String = MSTR_MODULO & ".blnExisteDatoenTabla"
```

```
Dim strSQL          As String
Dim rsDatos         As ADODB.Recordset

    On Error GoTo ErrorHandler
    Set rsDatos = New ADODB.Recordset

    blnExisteDatoenTabla = False

    'el primer campo siempre debe de ser de tipo string, dado que ese esta      buscando   un   nombre   o
         descripcion repetido
    strSQL = "Select " & pstrFld & _
           " From " & pstrTable & _
           " Where " & pstrFld & " = '" & pstrValuetoSearch & "' "

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld2 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld2 & " = " & pstrValuetoSearchFld2 & " "
    End If

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld3 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld3 & " = " & pstrValuetoSearchFld3 & " "
    End If

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld4 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld4 & " = " & pstrValuetoSearchFld4 & " "
    End If

    Set rsDatos = rsdDevuelveRecordset(strSQL)

    If Not rsDatos Is Nothing Then
        With rsDatos
            If Not (.BOF) And Not (.EOF) Then
                If .RecordCount > 0 Then
                    blnExisteDatoenTabla = True
                End If
            End If
            .Close
        End With
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnExisteDatoenTabla = False
            objErr.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set rsDatos = Nothing

End Function
```

Private Function OpenConnString()

Descripción: Funcion para obtener la cadena de conexion
' de un archivo de texto y guardarla en una
' propiedad que es global y accesible de cualquier
' lugar del proyecto

```
    Dim strConn     As String
    Dim objReg As RegKey
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strBD As String
    Dim strServidor As String
    Dim strConexion As String
    Dim lngReg As Long

On Error GoTo OpenSourceErr

    Set objReg = RegKeyFromString(GSTR_REGISTRY_RUTA)

    For lngReg = 1 To objReg.Values.Count

      If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_PASSWORD)Then

          If Trim$(objReg.Values(lngReg).Value) <> vbNullString Then
            Set objEncripta = New xmsEncriptador.Encriptador

            mstrPassword = objEncripta.DesencriptaString(Trim$(objReg.Values(lngReg).Value))


          End If

        End If

        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_USER) Then
           mstrUser = objReg.Values(lngReg).Value
        End If

        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_BD) Then
           strBD = objReg.Values(lngReg).Value
        End If


        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_SERVIDOR) Then
           strServidor = objReg.Values(lngReg).Value
        End If


        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_CONEXION) Then
           strConexion = objReg.Values(lngReg).Value
        End If

    Next

    strConn = strConexion

    If Trim$(strBD) <> vbNullString Then
       strConn = strConn & ";Initial Catalog=" & Trim$(strBD)
    End If

    If Trim$(strServidor) <> vbNullString Then
```

```
        strConn = strConn & ";Data Source=" & Trim$(strServidor)
    End If

    mstrConn = strConn


OpenSourceErr:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

    End Select

    'Libero la memoria
    Set objReg = Nothing
    Set objEncripta = Nothing

End Function
```

CLASE DE ACCESO A DATOS

Public Function blnBuscaRegistro(ByVal pstrDato As String, ByVal pstrTabla As String, ByVal pstrCampo As String, ByVal pblnCampoNumerico As Boolean) As Boolean

Descripción: Función para buscar un registro dado un valor, una tabla y un campo determinado.
Regresa Verdadero en caso de lograr la busqueda. Falso si no la encuentra.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnBuscaRegistro"

    Dim strSQL As String
    Dim rsdRecordset As ADODB.Recordset

On Error GoTo errHandler

    strSQL = "SELECT * " & vbCr
    strSQL = strSQL & " FROM " & pstrTabla & vbCr

    If pblnCampoNumerico Then
        strSQL = strSQL & " WHERE " & pstrCampo & "=" & pstrDato
    Else
        strSQL = strSQL & " WHERE " & pstrCampo & "='" & pstrDato & "'"
    End If

    Set rsdRecordset = rsdDevuelveRecordset(strSQL)

    If rsdRecordset.RecordCount > 0 Then
        'si existe por lo menos un registro
        blnBuscaRegistro = True
    Else
        'No existe el registro
        blnBuscaRegistro = False
    End If

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnBuscaRegistro = False
    End Select

    'Libero la memoria
    Set rsdRecordset = Nothing

End Function
```

Public Function blnConectaBD() As Boolean

Descripción: Funcion que abre la conexion a la BD

```
Dim objLog      As xmsError.xmsManejaError

    On Error GoTo ErrorHandler

    'Si no se ha asignado la propiedad de la cadena de conexion
```

```
    'la asigna
    If STR_CONN_STRING <> vbNullString Then
        If adoConnection.State = False Then
            With adoConnection
                .CursorLocation = adUseServer
                .Open STR_CONN_STRING, mstrUser, mstrPassword
            End With
            blnConectaBD = True
        Else
            blnConectaBD = True
        End If
    Else
        blnConectaBD = True
    End If
ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnConectaBD = False

            Set objLog = New xmsError.xmsManejaError

            objLog.RegistraError 2, "No se pudo crear una conexion a la BD" &                    vbCrLf    &    ""    &
STR_CONN_STRING, "BDAcceso.blnConectaBD"

            Set objLog = Nothing

'           Resume Next
'           objErr.RegistraError Err.Number,  vbCrLf & Err.Description,
            "SIFIBD.blnConectaBD"
    End Select
End Function
```

Public Function blnEjecutaQuery(ByVal pstrSQL As String) As Boolean

Descripción: Funcion ejecuta una cadena SQL

```
Dim adoCmd     As ADODB.Command
Dim objLog     As xmsError.xmsManejaError

    On Error GoTo ErrorHandler

    If blnConectaBD Then
        Set adoCmd = New ADODB.Command

        With adoCmd
            .CommandText = pstrSQL
            .CommandType = adCmdText
            .ActiveConnection = adoConnection
            .Execute
        End With
        blnEjecutaQuery = True

        Set adoCmd = Nothing
    Else
        blnEjecutaQuery = False
    End If
```

```
ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         blnEjecutaQuery = False

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 3, "No se pudo ejecutar la consulta : " &                vbCrLf  &  pstrSQL,
"BDAcceso.blnEjecutaQuery"

         Set objLog = Nothing

   End Select
End Function
```

Public Function rsdDevuelveRecordset(ByVal pstrSQL As String) As ADODB.Recordset

Descripion: Funcion para ejecutar una cadena de conexion y devuelve un

```
Dim adoRecordset    As ADODB.Recordset
Dim objLog          As xmsError.xmsManejaError

   On Error GoTo ErrorHandler

   If blnConectaBD = True Then
      Set adoRecordset = New ADODB.Recordset
      With adoRecordset
         .CursorLocation = adUseClient
         .CursorType = adOpenStatic
         .LockType = adLockBatchOptimistic
         .Source = pstrSQL
         .ActiveConnection = adoConnection
         .Open
         .ActiveConnection = Nothing
      End With
      Set rsdDevuelveRecordset = adoRecordset

      Set adoRecordset = Nothing

   End If


ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set rsdDevuelveRecordset = Nothing

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 4, "No se pudo ejecutar la consulta : " &                vbCrLf  &  pstrSQL,
"BDAcceso.rsdDevuelveRecordset"

         Set objLog = Nothing

'          Resume Next
```

```
    End Select
End Function


Public Function lngGetNewId(ByVal pstrTable As String, ByVal pstrField As String, _
                    Optional ByVal pstrDatoFiltrar1 As String, _
                    Optional ByVal plngValorFiltrar1 As Long, _
                    Optional ByVal pstrDatoFiltrar2 As String, _
                    Optional ByVal plngValorFiltrar2 As Long) As Double
```

Descripción: Funcion para ejecutar una cadena de conexion y devuelve

```
Dim strSQL          As String
Dim adoRecordset    As ADODB.Recordset
Dim objLog          As xmsError.xmsManejaError

On Error GoTo ErrorHandler

    Set adoRecordset = New ADODB.Recordset


    strSQL = "select max(" & pstrField & ") from " & pstrTable

    'Campo1 a filtar y valor1 a considerar
    If plngValorFiltrar1 > 0 Then
        strSQL = strSQL & " where " & pstrDatoFiltrar1 & " = " &
            plngValorFiltrar1
    End If

    'Campo2 a filtar y valor2 a considerar
    If plngValorFiltrar2 > 0 Then
        strSQL = strSQL & " and " & pstrDatoFiltrar2 & " = " & plngValorFiltrar2
    End If

    Set adoRecordset = rsdDevuelveRecordset(strSQL)


    lngGetNewId = -1

    If Not adoRecordset.EOF And Not adoRecordset.BOF Then
        If IsNumeric(adoRecordset(0)) Then
            lngGetNewId = CDbl(adoRecordset(0)) + 1
        Else
            lngGetNewId = 1
        End If
    End If

    Set adoRecordset = Nothing

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set adoRecordset = Nothing

            Set objLog = New xmsError.xmsManejaError

            objLog.RegistraError 5, "No se pudo obtener un nuevo ID de la Tabla                    :    **    "    &
pstrTable & " **  del campo : ** " & pstrField & " **                                ", "BDAcceso.lngGetNewId"
```

```
        Set objLog = Nothing

'       objErr.RegistraError Err.Number,  vbCrLf & Err.Description,
        "SIFISeguridad.lngGetNewIdUsuario"
        lngGetNewId = -1
'       Resume Next
   End Select
End Function
```

Public Function blnDeleteId(ByVal pstrTable As String, ByVal pstrField As String, ByVal pintValue As Integer) As Boolean

Descripción: Funcion para ejecutar una cadena de conexion y elimina
' el ID de la tabla enviado

```
Dim strSQL      As String
Dim objLog      As xmsError.xmsManejaError

On Error GoTo ErrorHandler


   strSQL = "delete from " & pstrTable & " Where " & pstrField & " = " &  pintValue

   blnDeleteId = blnEjecutaQuery(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         blnDeleteId = False

         Set objLog = New xmsError.xmsManejaError

         objLog.RegistraError 5, "No se pudo Eliminar el Registro de la Tabla : ** " & pstrTable & "
          **  del campo : ** " & pstrField & " **  con valor = " & pintValue,
          "BDAcceso.lngGetNewId"

         Set objLog = Nothing

   End Select
End Function
```

Public Function blnExisteDatoenTabla(ByVal pstrFld As String, ByVal pstrTable As String, ByVal pstrValuetoSearch As String, Optional ByVal pstrFld2 As String = vbNullString, Optional ByVal pstrValuetoSearchFld2 As String = vbNullString, Optional ByVal pstrFld3 As String = vbNullString, Optional ByVal pstrValuetoSearchFld3 As String = vbNullString, Optional ByVal pstrFld4 As String = vbNullString, Optional ByVal pstrValuetoSearchFld4 As String = vbNullString) As Boolean

Descripción: Funcion para determinar si existe un dato

```
Const STR_FUNCION As String = MSTR_MODULO & ".blnExisteDatoenTabla"
```

```
Dim strSQL          As String
Dim rsDatos         As ADODB.Recordset

    On Error GoTo ErrorHandler
    Set rsDatos = New ADODB.Recordset

    blnExisteDatoenTabla = False

    'el primer campo siempre debe de ser de tipo string, dado que ese esta      buscando   un   nombre   o
         descripcion repetido
    strSQL = "Select " & pstrFld & _
           " From " & pstrTable & _
           " Where " & pstrFld & " = '" & pstrValuetoSearch & "' "

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld2 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld2 & " = " & pstrValuetoSearchFld2 & " "
    End If

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld3 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld3 & " = " & pstrValuetoSearchFld3 & " "
    End If

    'los siguientes campos seran de tipo integer, dado que pueden ser id
    If pstrValuetoSearchFld4 <> vbNullString Then
      strSQL = strSQL & " and " & pstrFld4 & " = " & pstrValuetoSearchFld4 & " "
    End If

    Set rsDatos = rsdDevuelveRecordset(strSQL)

    If Not rsDatos Is Nothing Then
        With rsDatos
            If Not (.BOF) And Not (.EOF) Then
                If .RecordCount > 0 Then
                    blnExisteDatoenTabla = True
                End If
            End If
            .Close
        End With
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnExisteDatoenTabla = False
            objErr.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set rsDatos = Nothing

End Function
```

Private Function OpenConnString()

Descripción: Funcion para obtener la cadena de conexion
' de un archivo de texto y guardarla en una
' propiedad que es global y accesible de cualquier
' lugar del proyecto

```
    Dim strConn     As String
    Dim objReg As RegKey
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strBD As String
    Dim strServidor As String
    Dim strConexion As String
    Dim lngReg As Long

On Error GoTo OpenSourceErr

    Set objReg = RegKeyFromString(GSTR_REGISTRY_RUTA)

    For lngReg = 1 To objReg.Values.Count

      If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_PASSWORD)Then

          If Trim$(objReg.Values(lngReg).Value) <> vbNullString Then
            Set objEncripta = New xmsEncriptador.Encriptador

            mstrPassword = objEncripta.DesencriptaString(Trim$(objReg.Values(lngReg).Value))


          End If

        End If

        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_USER) Then
           mstrUser = objReg.Values(lngReg).Value
        End If

        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_BD) Then
           strBD = objReg.Values(lngReg).Value
        End If


        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_SERVIDOR) Then
           strServidor = objReg.Values(lngReg).Value
        End If


        If UCase$(objReg.Values(lngReg).Name) = UCase$(GSTR_REGISTRY_CONEXION) Then
           strConexion = objReg.Values(lngReg).Value
        End If

    Next

    strConn = strConexion

    If Trim$(strBD) <> vbNullString Then
       strConn = strConn & ";Initial Catalog=" & Trim$(strBD)
    End If

    If Trim$(strServidor) <> vbNullString Then
```

```
        strConn = strConn & ";Data Source=" & Trim$(strServidor)
    End If

    mstrConn = strConn


OpenSourceErr:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

    End Select

    'Libero la memoria
    Set objReg = Nothing
    Set objEncripta = Nothing

End Function
```

CLASE DE CATALOGOS

'Declaración del Enum para selección de catálogos

Public Enum enmCatalogos
catTipoTelefono = 0
catRubro = 1
catNaturalezaRubro = 2
catEmpleado = 3
CatCentroCosto = 4
catDepartamento = 5
CatOficina = 6
CatMoneda = 8
CatTipoCambio = 9
CatPrestadores = 10
CatArea = 11
CatOficinasArea = 12
catBanco = 18
catOrdenStatus = 20
catOrdenPago = 21
CatOrdenPagoCuenta = 22
catTipoConsulta = 23
catConceptoRubro = 24
catCuentasBancarias = 27
catOficinaImpuesto = 28
catTipoDomicilio = 29
catProveedorDomicilio = 30
catProveedorTelefono = 31
catEdoCivil = 32
catTipoServicio = 33
catTipoImpuesto = 34
CatOrdenPagoCuentaImpuesto = 35
catTipoPago = 36
catStatusTraspasos = 38
catMovimientosBitacora = 39
catParametrosUniverse = 40
catEstados = 43
catCiudades = 44
catMunicipios = 45
catColonias = 46
catTipoRubro = 50
catRubroTipoRubro = 51

catBancos = 52
catPlaza = 53
catSucursal = 54
catOP_Banca = 55
catCuentasPagadoras = 56
catFiltros = 57
catBancosRubro = 58
catCtasBancariasRubro = 59
catGuiasAutorizadasCtroCostos = 60
catGuiaMonedaTipoPersona = 61
catGuiaRubros = 62
catPlantillasConceptos = 63
catInfBancaria = 64
catLotes = 65
catRangos = 66
catSistBanca = 67
catTipoCuenta = 68
catReportes = 69
catOficina_Impr = 70
catNivelConsulta = 71

End Enum

Public Function blnCreaDato(ByVal plngEmpleado As Long, ByVal catCatalogo As enmCatalogos, ByVal pstrDato1 As String, Optional ByVal pstrDato2 As String, Optional ByVal pstrDato3 As String, Optional ByVal pstrDato4 As String, Optional ByVal pstrDato5 As String, Optional ByVal pstrDato6 As String, Optional ByVal pstrDato7 As String, Optional ByVal pstrDato8 As String, Optional ByVal pstrDato9 As String) As Boolean

Descripción: función que guarda un dato determinado en una tabla especifica

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnCreaDato"


    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso
    Dim lngID As Long
    Dim dtmFecha As Date

On Error GoTo errHandler


mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)


    strSQL = ""
    mstrConcepto = ""

    Select Case catCatalogo
       Case 1      'Rubros

          'Nuevo Rubro
          Set objBD = New SIFIBD.BDAcceso
          'Verifica que el rubro no este ya dado de alta para el mismo concepto
          If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_DESCRIPCION, _
                              GSTR_TBL_RUBRO, _
                              pstrDato1, _
                              GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, _
                              pstrDato2) Then

             'Valido que no exista la cuenta contable
             If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE, _
                                 GSTR_TBL_RUBRO, _
                                 pstrDato3) Then

                'Valido que no exista la cuenta bancaria
                If Not objBD.blnExisteDatoenTabla(GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA, _
                                    GSTR_TBL_RUBRO, _
                                    pstrDato6) Then

                   lngID = objBD.lngGetNewId(GSTR_TBL_RUBRO, GSTR_TBL_RUBRO_FLD_ID)
                   strSQL = "INSERT INTO " & GSTR_TBL_RUBRO & vbCr
                   strSQL = strSQL & "(" & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "," & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_MONTO_MAX & "," & vbCr
                   strSQL = strSQL & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
                   'Agregadas 15-Mayo-2004
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_MONEDA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_BANCO & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_TIPOCUENTA & ", " & vbCr
                   strSQL = strSQL & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & ")"

                   strSQL = strSQL & " VALUES (" & vbCr
                   strSQL = strSQL & lngID & ", " & vbCr
```

```
                strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "', " & vbCr
                strSQL = strSQL & pstrDato3 & ", " & vbCr
                strSQL = strSQL & CLng(pstrDato2) & "," & vbCr

                If Trim$(pstrDato4) <> vbNullString Then
                   strSQL = strSQL & CDbl(pstrDato4) & "," & vbCr
                Else
                   strSQL = strSQL & "NULL," & vbCr
                End If

                strSQL = strSQL & CLng(pstrDato5) & "," & vbCr

                '15-Mayo-2004
                If Trim$(pstrDato6) <> vbNullString Then   'Cuenta Bancaria
                   strSQL = strSQL & "'" & UCase$(Trim$(pstrDato6)) & "', " & vbCr
                Else
                   strSQL = strSQL & "NULL," & vbCr
                End If

                If Trim$(pstrDato7) <> vbNullString Then  'IDMoneda
                   strSQL = strSQL & CDbl(pstrDato7) & ", "
                Else
                   strSQL = strSQL & "NULL" & ", "
                End If

                'ID banco
                strSQL = strSQL & CDbl(pstrDato8) & ", "

                'IDTipoCuenta
                strSQL = strSQL & CDbl(pstrDato9) & ", "

                'ID_Activo: cuando se crea está con estatus Activo: 1
                strSQL = strSQL & "1)"

             End If

          End If

       End If

       mstrConcepto = GSTR_TBL_RUBRO_FLD_DESCRIPCION & "=  " & UCase$(Trim$(pstrDato1)) & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & "= " & pstrDato3 & " " &
                      vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_MONTO_MAX & "= " & pstrDato4 & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & "=  " & pstrDato6 & " " &
                      vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_ID_MONEDA & "= " & pstrDato7 & " " & vbCr
       mstrConcepto = mstrConcepto & GSTR_TBL_RUBRO_FLD_ID_BANCO & "= " & pstrDato8

       Call RegistrarBitacora(GSTR_TBL_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
         mlngIdOficina)

    Case 9
       dtmFecha = CDate(GetDateFromServer)

       strSQL = "INSERT INTO " & GSTR_TBL_TIPO_CAMBIO & vbCr
       strSQL = strSQL & " (" & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_MES & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR & ", " & vbCr
       strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ID & ") " & vbCr
       strSQL = strSQL & " VALUES (" & vbCr

       strSQL = strSQL & Year(dtmFecha) & ", " & vbCr
       strSQL = strSQL & Month(dtmFecha) & ", " & vbCr
       strSQL = strSQL & Format(CDbl(pstrDato1), "##.0000") & ", " & vbCr
       strSQL = strSQL & CDbl(pstrDato2) & ") " & vbCr
```

```
    Set objBD = New SIFIBD.BDAcceso

Case 11     'Catálogo de áreas

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de areas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_AREA_FLD_NOMBRE, _
                            GSTR_TBL_AREA, _
                            pstrDato1) = False Then
        lngID = objBD.lngGetNewId(GSTR_TBL_AREA, GSTR_TBL_AREA_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_AREA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_AREA_FLD_NOMBRE & ")" & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_AREA_FLD_NOMBRE & "= " & pstrDato1

        Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    End If

Case 15 'Pantalla

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de areas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_PANTALLA_FLD_NOMBRE, _
                            GSTR_TBL_PANTALLA, _
                            pstrDato1) = False Then
        lngID = objBD.lngGetNewId(GSTR_TBL_PANTALLA, GSTR_TBL_PANTALLA_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_PANTALLA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_PANTALLA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PANTALLA_FLD_NOMBRE & ")" & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_PANTALLA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1))

        Call RegistrarBitacora(GSTR_TBL_PANTALLA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    End If

        'strSQL = "select " & GSTR_TBL_PANTALLA_FLD_ID & ", " & _
                    GSTR_TBL_PANTALLA_FLD_NOMBRE & " from " & _
                    GSTR_TBL_PANTALLA & " Order by " & _
                    GSTR_TBL_PANTALLA_FLD_NOMBRE

Case 23     'Catálogo de Tipos de Consulta de información

    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso
```

```
    lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_CONSULTA, GSTR_TBL_TIPO_CONSULTA_FLD_ID)

    strSQL = "INSERT INTO " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & ")" & vbCr

    strSQL = strSQL & " VALUES (" & vbCr
    strSQL = strSQL & lngID & ", " & vbCr
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "')"

    mstrConcepto = GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1))

    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
      mstrConcepto, mlngIdOficina)

Case 24     'Catálogo de Conceptos de Grupo


    Set objBD = New SIFIBD.BDAcceso

    If objBD.blnBuscaRegistro(pstrDato2, GSTR_TBL_CONCEPTO_RUBRO,
      GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, True) Then
        blnCreaDato = False
        Exit Function
    End If

    If objBD.blnBuscaRegistro(pstrDato1, GSTR_TBL_CONCEPTO_RUBRO,
      GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO, False) Then
        blnCreaDato = False
        Exit Function
    Else
        'Obtengo el consecutivo del ID
        lngID = objBD.lngGetNewId(GSTR_TBL_CONCEPTO_RUBRO,
                GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE)

        strSQL = "INSERT INTO " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
        strSQL = strSQL & " (" & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & ") " & vbCr
        strSQL = strSQL & " VALUES (" & lngID & ", " & vbCr
        strSQL = strSQL & Val(pstrDato2) & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "', " & vbCr
        strSQL = strSQL & "'" & pstrDato3 & "')"

        mstrConcepto = GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "= " & pstrDato2 & " " & vbCr
        mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "= " &
                       UCase$(Trim$(pstrDato1)) & " " & vbCr
        mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & "= " &
                       pstrDato3

        Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                mstrConcepto, mlngIdOficina)

    End If

Case 25     'RolRubro
    strSQL = "INSERT INTO ROL_PLANTILLA (ID_GUI_CON,IDROL) " & vbCr
    strSQL = strSQL & " VALUES (" & pstrDato1 & ", " & CLng(pstrDato2) & ")"

    mstrConcepto = GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "= " & pstrDato1 & " "
                       & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_ROL_FLD_ID & "= " & CLng(pstrDato2)
    Call RegistrarBitacora("ROL_PLANTILLA", plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                mlngIdOficina)

    Set objBD = New SIFIBD.BDAcceso
```

```
Case 26     'rol - Oficina
    strSQL = "INSERT INTO " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_OFICINA_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES (" & CLng(pstrDato1) & ", " & vbCr
    strSQL = strSQL & CLng(pstrDato2) & ")"


    mstrConcepto = GSTR_TBL_ROL_FLD_ID & "= " & CLng(pstrDato1) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_OFICINA_FLD_ID & "= " & CLng(pstrDato2)

    Call RegistrarBitacora(GSTR_TBL_ROL_OFICINAS, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
      mstrConcepto, mlngIdOficina)

    Set objBD = New SIFIBD.BDAcceso

Case 28     'Oficina - Impuesto
    strSQL = "INSERT INTO " & GSTR_TBL_OFICINA_IMPUESTO & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_OFICINA_IMPUESTO_FLD_VALOR & ") " & vbCr

    'Valores
    strSQL = strSQL & " VALUES (" & vbCr
    'IDOficina
    strSQL = strSQL & pstrDato1 & ", " & vbCr
    'IDImpuesto
    strSQL = strSQL & pstrDato2 & ", " & vbCr
    'Valor
    strSQL = strSQL & pstrDato3 & ") "



    Call RegistrarBitacora(GSTR_TBL_OFICINA_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)


    Set objBD = New SIFIBD.BDAcceso

Case 31     'Proveedor telefono
    Set objBD = New SIFIBD.BDAcceso

    lngID = objBD.lngGetNewId(GSTR_TBL_PRESTADOR_TEL, GSTR_TBL_PRESTADOR_TEL_FLD_ID)

    strSQL = "INSERT INTO " & GSTR_TBL_PRESTADOR_TEL & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_PRESTADOR_TEL_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_TIPO_TELEFONO_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_CLAVE & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_NUMERO & ", "
    strSQL = strSQL & GSTR_TBL_PRESTADOR_TEL_FLD_EXTENSION & ") "

    strSQL = strSQL & " VALUES (" & vbCr
    'ID
    strSQL = strSQL & lngID & ", "
    'Id Tipo Telefono
    strSQL = strSQL & CLng(pstrDato1) & ", "
    'Id Prestador
    strSQL = strSQL & CLng(pstrDato2) & ", "
    'clave
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato3)) & "',"
    'Número
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato4)) & "',"
    'extensión
    strSQL = strSQL & "'" & UCase$(Trim$(pstrDato5)) & "')"
```

```
    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "= " & CLng(pstrDato2) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_CLAVE & "= " &
                        UCase$(Trim$(pstrDato3)) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_NUMERO & "= " &
                        UCase$(Trim$(pstrDato4)) & " " & vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_PRESTADOR_TEL_FLD_EXTENSION & "= " &
                        UCase$(Trim$(pstrDato5))



    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_TEL, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                        mstrConcepto, mlngIdOficina)

Case 34     'Impuestos
    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_TIPO_IMPUESTO_FLD_NOMBRE, _
                        GSTR_TBL_TIPO_IMPUESTO, _
                        pstrDato1) Then

        lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_IMPUESTO, GSTR_TBL_TIPO_IMPUESTO_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_TIPO_IMPUESTO & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_IMPUESTO_FLD_CUENTA & ") " & vbCr
        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato1)) & "'," & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato2)) & "')"


        Call RegistrarBitacora(GSTR_TBL_TIPO_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)

    End If

Case 36     'Tipo Pago
    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION, _
                        GSTR_TBL_TIPO_PAGO, _
                        pstrDato1) Then

        lngID = objBD.lngGetNewId(GSTR_TBL_TIPO_PAGO, GSTR_TBL_TIPO_PAGO_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_TIPO_PAGO & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION & ")" & vbCr

        'Valores
        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & Trim$(UCase$(pstrDato1)) & "')"

        mstrConcepto = GSTR_TBL_TIPO_PAGO_FLD_DESCRIPCION & "= " & Trim$(UCase$(pstrDato1))

        Call RegistrarBitacora(GSTR_TBL_TIPO_PAGO, plngEmpleado, MSTR_MOVIMIENTO_ALTA,
                mstrConcepto, mlngIdOficina)

    End If

Case 51     'Rubro - Tipo Rubro
    strSQL = "INSERT INTO " & GSTR_TBL_RUBRO_TIPO_RUBRO & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_RUBRO_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES " & vbCr
    strSQL = strSQL & "(" & pstrDato1 & ", " & vbCr
```

```
    strSQL = strSQL & pstrDato2 & ")" & vbCr

    Call RegistrarBitacora(GSTR_TBL_RUBRO_TIPO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ALTA)

    Set objBD = New SIFIBD.BDAcceso

Case 52 'Catálogo de Bancos
    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de Bancos
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_BANCOS_FLD_DESCRIPCION, _
                            GSTR_TBL_BANCOS, _
                            pstrDato1, GSTR_TBL_BANCOS_FLD_ACTIVO, 1) Then
        lngID = objBD.lngGetNewId(GSTR_TBL_BANCOS, GSTR_TBL_BANCOS_FLD_ID)

        strSQL = "INSERT INTO " & GSTR_TBL_BANCOS & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_BANCOS_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_DESCRIPCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_ID_IMPRESION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_IDBMR & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_BANCOS_FLD_ACTIVO & ") " & vbCr

        strSQL = strSQL & " VALUES (" & vbCr
        strSQL = strSQL & lngID & ", " & vbCr
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "',"
        strSQL = strSQL & "'" & UCase$(Trim$(pstrDato2)) & "',"
        strSQL = strSQL & UCase$(Trim$(pstrDato3)) & ", "
        strSQL = strSQL & "1)"


        mstrConcepto = GSTR_TBL_BANCOS_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1))
        Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                                mlngIdOficina)
    End If

Case 53 'Catálogo de Plazas
    'Obtengo el consecutivo del ID
    Set objBD = New SIFIBD.BDAcceso

    'se verifica que no exista el dato en la tabla de Plazas
    'si no existe se puede insertar, de otra forma no se intenta la insercion
    If objBD.blnExisteDatoenTabla(GSTR_TBL_PLAZA_FLD_CVEBANCO, _
                            GSTR_TBL_PLAZA, _
                            pstrDato4, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, _
                            pstrDato2, GSTR_TBL_PLAZA_FLD_ID_PLAZA, pstrDato3,
                             GSTR_TBL_PLAZA_FLD_ACTIVO, 1) Then

        blnCreaDato = False
        Exit Function
    End If

    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_PLAZA_FLD_DESCRIPCION, _
                            GSTR_TBL_PLAZA, _
                            pstrDato1, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, _
                            pstrDato2, GSTR_TBL_PLAZA_FLD_ACTIVO, 1) Then
        lngID = objBD.lngGetNewId(GSTR_TBL_PLAZA, GSTR_TBL_PLAZA_FLD_ID_PLAZA, _
                            GSTR_TBL_PLAZA_FLD_ID_BANCO, pstrDato2)

        strSQL = "INSERT INTO " & GSTR_TBL_PLAZA & vbCr
        strSQL = strSQL & "(" & GSTR_TBL_PLAZA_FLD_ID_BANCO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_ID_PLAZA & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_DESCRIPCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_PLAZA_FLD_CVEBANCO & ", " & vbCr
```

```
            strSQL = strSQL & GSTR_TBL_PLAZA_FLD_CVEBANCO_BMR & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_PLAZA_FLD_ACTIVO & ") " & vbCr
            strSQL = strSQL & " VALUES (" & vbCr
            strSQL = strSQL & Val(pstrDato2) & ", " & vbCr
            strSQL = strSQL & lngID & ", " & vbCr
            strSQL = strSQL & "'" & UCase$(Trim$(pstrDato1)) & "'," & vbCr
            strSQL = strSQL & pstrDato4 & ","
            strSQL = strSQL & pstrDato5 & ",1)"

            mstrConcepto = GSTR_TBL_PLAZA_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1)) & " " & vbCr
            mstrConcepto = mstrConcepto & GSTR_TBL_PLAZA_FLD_CVEBANCO & "= " & pstrDato4

            Call RegistrarBitacora(GSTR_TBL_PLAZA, plngEmpleado, MSTR_MOVIMIENTO_ALTA, mstrConcepto,
                    mlngIdOficina)
        End If



errHandler:
    Select Case Err.Number

        Case 0

        Case Else
            Call mobjError.RegistraError(Err.Number, Err.Description, STR_FUNCION)
            blnCreaDato = False
            Err.Clear
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnEliminaDato(ByVal plngEmpleado As Long, ByVal enmCatalogo As enmCatalogos, ByVal plngID As Long, Optional ByVal plngValor As Long, Optional ByVal plngValor2 As Long, Optional ByVal pstrDato1 As String) As Boolean

Descripción: Función para eliminar un registro en una tabla determinada

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnEliminaDato"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String

On Error GoTo errHandler
mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)



    Select Case enmCatalogo
        Case 1  'Rubro
            strSQL = "UPDATE " & GSTR_TBL_RUBRO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=0"
            strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO_FLD_ID & "=" & plngID

            mstrConcepto = GSTR_TBL_RUBRO_FLD_ID & "=" & plngID

            Call RegistrarBitacora(GSTR_TBL_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                    mlngIdOficina)

        Case 10     'Proveedores

            strSQL = " DELETE " & GSTR_TBL_PRESTADOR_TEL & vbCrLf
            strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID & vbCrLf
```

```
    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)


    strSQL = " DELETE " & GSTR_TBL_PRESTADOR_DOMICILIO & vbCrLf
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & vbCrLf

    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)


    strSQL = " DELETE " & GSTR_TBL_PRESTADOR & vbCrLf
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & vbCrLf


    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_PRESTADOR, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)

Case 11     'Areas
    strSQL = "DELETE " & GSTR_TBL_AREA & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_AREA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_AREA_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                        mlngIdOficina)

Case 23     'Tipo Consulta
    strSQL = "DELETE " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)

Case 24     'Concepto del Rubro

    Set objBD = New SIFIBD.BDAcceso

    If Not objBD.blnBuscaRegistro(plngID, GSTR_TBL_RUBRO, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, True)
      Then


        strSQL = "DELETE " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
        strSQL = strSQL & " WHERE " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & plngID

        mstrConcepto = GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & plngID
        Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                mstrConcepto, mlngIdOficina)
    Else
            blnEliminaDato = False
            Exit Function
    End If

Case 25     'Rol - Rubro
    strSQL = "DELETE ROL_PLANTILLA " & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "=" &
                plngID
    strSQL = strSQL & " AND " & GSTR_TBL_ROL_FLD_ID & "=" & plngValor

    mstrConcepto = GSTR_TBL_GUIA_MONEDA_TIPO_PERSONA_FLD_ID_GUIA_CON & "=" & plngID & " " &
                        vbCr
    mstrConcepto = mstrConcepto & GSTR_TBL_ROL_FLD_ID & "=" & plngValor
```

```
    Call RegistrarBitacora("ROL_PLANTILLA", plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                        mlngIdOficina)

Case 26     'Rol - Oficinas
    strSQL = "DELETE " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_OFICINA_FLD_ID & "=" & plngValor

    mstrConcepto = GSTR_TBL_ROL_FLD_ID & "=" & plngID
    mstrConcepto = mstrConcepto & GSTR_TBL_OFICINA_FLD_ID & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_ROL_OFICINAS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                        mstrConcepto, mlngIdOficina)

Case 28     'Oficina - Impuesto
    strSQL = "DELETE " & GSTR_TBL_OFICINA_IMPUESTO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_OFICINA_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & "=" & plngValor



    Call RegistrarBitacora(GSTR_TBL_OFICINA_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case 30     'Prestador - Domicilio
    strSQL = "DELETE " & GSTR_TBL_PRESTADOR_DOMICILIO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_DOMICILIO_FLD_ID & "=" & plngValor


    mstrConcepto = GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_DOMICILIO_FLD_ID & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_DOMICILIO, plngEmpleado,
                                MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto, mlngIdOficina)

Case 31     'Prestador - Telefono
    strSQL = "DELETE " & GSTR_TBL_PRESTADOR_TEL & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_PRESTADOR_TEL_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_PRESTADOR_TEL, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 34     'Tipo Impuesto
    strSQL = "DELETE " & GSTR_TBL_TIPO_IMPUESTO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_IMPUESTO_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_TIPO_IMPUESTO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case 36     'Tipo Pago
    strSQL = "DELETE " & GSTR_TBL_TIPO_PAGO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_PAGO_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_TIPO_PAGO_FLD_ID & "=" & plngID
    Call RegistrarBitacora(GSTR_TBL_TIPO_PAGO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 51     'Rubro - Tipo Rubro
    strSQL = "DELETE " & GSTR_TBL_RUBRO_TIPO_RUBRO & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO_FLD_ID & "=" & plngID & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_RUBRO_FLD_ID & "=" & plngValor & vbCr


    mstrConcepto = GSTR_TBL_RUBRO_FLD_ID & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_RUBRO_FLD_ID & "=" & plngValor & vbCr
```

```
    Call RegistrarBitacora(GSTR_TBL_RUBRO_TIPO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                           mstrConcepto, mlngIdOficina)

Case 52     'Bancos_TE

    strSQL = "UPDATE " & GSTR_TBL_BANCOS & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_BANCOS_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_BANCOS_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_BANCOS_FLD_ID & "=" & plngID

    Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                           mlngIdOficina)

Case 53     'Plazas
    strSQL = "UPDATE " & GSTR_TBL_PLAZA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_PLAZA_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_PLAZA_FLD_ID_PLAZA & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_PLAZA_FLD_ID_BANCO & "=" & plngValor


     mstrConcepto = GSTR_TBL_PLAZA_FLD_ID_PLAZA & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_PLAZA_FLD_ID_BANCO & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_BANCOS, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR, mstrConcepto,
                           mlngIdOficina)

Case 54     'Sucursal
    strSQL = "UPDATE " & GSTR_TBL_SUCURSAL & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_SUCURSAL_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_SUCURSAL_FLD_ID_BANCO & "=" & plngValor
    strSQL = strSQL & " AND " & GSTR_TBL_SUCURSAL_FLD_ID_PLAZA & "=" & plngValor2

    mstrConcepto = GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & "=" & plngID & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_SUCURSAL_FLD_ID_BANCO & "=" & plngValor & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_SUCURSAL_FLD_ID_PLAZA & "=" & plngValor2

    Call RegistrarBitacora(GSTR_TBL_SUCURSAL, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                           mstrConcepto, mlngIdOficina)


Case 55     'Operación Bancaria
    strSQL = "UPDATE " & GSTR_TBL_OP_BANCA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_OP_BANCA_FLD_ACTIVO & "=0"
    strSQL = strSQL & " WHERE " & GSTR_TBL_OP_BANCA_FLD_ID & "=" & plngID
    strSQL = strSQL & " AND " & GSTR_TBL_OP_BANCA_FLD_ID_BANCO & "=" & plngValor

    mstrConcepto = GSTR_TBL_OP_BANCA_FLD_ID & "=" & plngID & " "
    mstrConcepto = GSTR_TBL_OP_BANCA_FLD_ID_BANCO & "=" & plngValor

    Call RegistrarBitacora(GSTR_TBL_OP_BANCA, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR,
                                mstrConcepto, mlngIdOficina)

Case 56  'Cuentas Bancarias


Case 57  'Filtros de búsquedas
    strSQL = "DELETE " & GSTR_TBL_FIL_OP & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_FIL_OP_FLD_IDR_FILTRO & "=" & plngID


    Call RegistrarBitacora(GSTR_TBL_FIL_OP, plngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

Case Else
    mobjError.RegistraError 203, GSTR_ERROR_203, STR_FUNCION
```

```
    End Select


    Set objBD = New SIFIBD.BDAcceso
    blnEliminaDato = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnEliminaDato = False
            Err.Clear
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnModificaDato(ByVal plngEmpleado As Long, ByVal catCatalogo As enmCatalogos, ByVal plngID As Long, ByVal pstrDato1 As String, Optional ByVal pstrDato2 As String, Optional ByVal pstrDato3 As String, Optional ByVal pstrDato4 As String, Optional ByVal pstrDato5 As String, Optional ByVal pstrDato6 As String, Optional ByVal pstrDato7 As String, Optional ByVal pstrDato8 As String, Optional ByVal pstrDato9 As String) As Boolean

Descripción: función general para eliminar un registro de la Base de datos

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnModificaDato"

    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler
mlngIdOficina = strObtieneCve(GSTR_TBL_OFICINA_FLD_ID, plngEmpleado, GSTR_TBL_EMPLEADO,
GSTR_TBL_EMPLEADO_FLD_ID, True)



    Select Case catCatalogo

        Case 11     'Areas
            strSQL = "UPDATE " & GSTR_TBL_AREA & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_AREA_FLD_NOMBRE & "='" & UCase$(Trim$(pstrDato1)) & "'" &
                     vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_AREA_FLD_ID & "=" & plngID

            mstrConcepto = strSQL = strSQL & " SET " & GSTR_TBL_AREA_FLD_NOMBRE & "= " &
                           UCase$(Trim$(pstrDato1))
            Call RegistrarBitacora(GSTR_TBL_AREA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR, mstrConcepto,
                                   mlngIdOficina)

        Case 13 'Tipo Objeto
            strSQL = "UPDATE " & GSTR_TBL_TIPO_OBJETO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION & "='" &
                     UCase$(Trim$(pstrDato1)) & "'" & vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_OBJETO_FLD_ID & "=" & plngID

            mstrConcepto = GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1))
            Call RegistrarBitacora(GSTR_TBL_TIPO_OBJETO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                           mstrConcepto, mlngIdOficina)

        Case 14 'Rol
```

```
    strSQL = "UPDATE " & GSTR_TBL_ROL & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_ROL_FLD_DESCRIPCION & "='" & UCase$(Trim$(pstrDato1)) & "'" &
                        vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_CANTIDAD & "=" & CDbl(pstrDato2) & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_ROL_FLD_DESCRIPCION & "= " & UCase$(Trim$(pstrDato1)) & " "
    mstrConcepto = GSTR_TBL_ROL_FLD_CANTIDAD & "= " & CDbl(pstrDato2)

    Call RegistrarBitacora(GSTR_TBL_ROL, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR, mstrConcepto,
                        mlngIdOficina)

Case 15 'Pantalla
    strSQL = "UPDATE " & GSTR_TBL_PANTALLA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_PANTALLA_FLD_NOMBRE & "='" & UCase$(Trim$(pstrDato1)) & "'"
                & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_PANTALLA_FLD_ID & "=" & plngID

    mstrConcepto = GSTR_TBL_PANTALLA_FLD_NOMBRE & "= " & UCase$(Trim$(pstrDato1)) & " "
    mstrConcepto = mstrConcepto & GSTR_TBL_PANTALLA_FLD_ID & "= " & plngID

    Call RegistrarBitacora(GSTR_TBL_PANTALLA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                                    mstrConcepto, mlngIdOficina)

Case 17 'Objeto
    strSQL = "UPDATE " & GSTR_TBL_OBJETO
    strSQL = strSQL & " SET " & GSTR_TBL_OBJETO_FLD_NOMBRE & " = '" & UCase$(Trim$(pstrDato1)) & "' " &
                        vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_OBJETO_FLD_ID & " = " & plngID & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_PANTALLA_FLD_ID & " = " & CLng(pstrDato3) & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_TIPO_OBJETO_FLD_ID & " = " & CLng(pstrDato2)


    mstrConcepto = GSTR_TBL_OBJETO_FLD_NOMBRE & " = " & UCase$(Trim$(pstrDato1)) & " " & vbCr

    Call RegistrarBitacora(GSTR_TBL_OBJETO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                        mstrConcepto, mlngIdOficina)

Case 23     'Modifica Tipo Consulta
    strSQL = "UPDATE " & GSTR_TBL_TIPO_CONSULTA & vbCr
    strSQL = strSQL & " SET " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "='" &
                UCase$(Trim$(pstrDato1)) & "'" & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & "=" & plngID

    mstrConcepto = mstrConcepto & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & "= " &
                        UCase$(Trim$(pstrDato1))

    Call RegistrarBitacora(GSTR_TBL_TIPO_CONSULTA, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
                                    mstrConcepto, mlngIdOficina)

Case 24     'Concepto del Rubro

    Set objBD = New SIFIBD.BDAcceso

     If Not objBD.blnExisteDatoenTabla(GSTR_TBL_CONCEPTO_RUBRO_FLD_ID,
                GSTR_TBL_CONCEPTO_RUBRO, pstrDato1, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE,
     plngID) Then

     If objBD.blnBuscaRegistro(pstrDato1, GSTR_TBL_CONCEPTO_RUBRO,
     GSTR_TBL_CONCEPTO_RUBRO_FLD_ID, False) Then
         blnModificaDato = False
         Exit Function
       End If
    End If
```

```
            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO,
            GSTR_TBL_CONCEPTO_RUBRO, pstrDato2, GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE, plngID)
            Then

            If objBD.blnBuscaRegistro(pstrDato2, GSTR_TBL_CONCEPTO_RUBRO,
            GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO, False) Then
                blnModificaDato = False
                Exit Function
              End If
          End If

          strSQL = "UPDATE " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
            strSQL = strSQL & " SET " & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "='" &
            UCase$(Trim$(pstrDato2)) & "'" & vbCr
          strSQL = strSQL & ", " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "=" & pstrDato1 & vbCr
          strSQL = strSQL & " WHERE " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & "=" & plngID

            mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & "= " &
            UCase$(Trim$(pstrDato2)) & " " & vbCr
          mstrConcepto = mstrConcepto & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "= " & pstrDato1 & vbCr

            Call RegistrarBitacora(GSTR_TBL_CONCEPTO_RUBRO, plngEmpleado, MSTR_MOVIMIENTO_MODIFICAR,
            mstrConcepto, mlngIdOficina)



        Case Else
            mobjError.RegistraError 202, GSTR_ERROR_201, STR_FUNCION
            blnModificaDato = False
            Exit Function
    End Select

    If objBD Is Nothing Then
        Set objBD = New SIFIBD.BDAcceso
    End If

    If Trim$(strSQL) <> vbNullString Then
        blnModificaDato = objBD.blnEjecutaQuery(strSQL)
    End If

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnModificaDato = False
            Err.Clear
    End Select

End Function
```

Public Function LoadCatalogo(ByVal catCatalogo As enmCatalogos, _
Optional ByVal plngIdDato1 As Long = 0, _
Optional ByVal plngIdDato2 As Long = 0, _
Optional ByVal plngIdDato3 As Long = 0, _
Optional ByVal plngIdDato4 As Long = 0, _
Optional ByVal plngIdDato5 As Long = 0, _
Optional ByVal plngIdDato6 As Long = 0, _
Optional ByVal plngIdDato7 As Long = 0, _
Optional ByVal plngIDDato8 As Long) As ADODB.Recordset

Descripción: Función General para cargar la información de un catálogo específico en un recordset y

```
                poder deplegarlo en un objeto

    Const STR_FUNCION As String = MSTR_MODULO & ".LoadCatalogo"

    Const LNG_ACTIVO As Long = 1

    Dim strSQL As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler

    Select Case catCatalogo

        Case 0 'Tipo Telefono
            strSQL = "SELECT " & GSTR_TBL_TIPO_TELEFONO_FLD_ID & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_TIPO_TELEFONO_FLD_TIPO & vbCr
            strSQL = strSQL & " FROM " & GSTR_TBL_TIPO_TELEFONO & vbCr
            strSQL = strSQL & " ORDER BY " & GSTR_TBL_TIPO_TELEFONO_FLD_TIPO

        Case 1 'Rubro
            strSQL = "SELECT " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_MONTO_MAX & ", " & vbCr
              strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ",
              " & vbCr
            strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." &
            GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr


            strSQL = strSQL & " CONVERT(bigint," & GSTR_TBL_RUBRO & "." &
            GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ") AS " &
            GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
              strSQL = strSQL & GSTR_TBL_NATURALEZA & "." & GSTR_TBL_NATURALEZA_DESCRIPCION & " AS " &
              GSTR_TBL_NATURALEZA & ", " & vbCr

            '15-Mayo-2004
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_TIPOCUENTA & "," & vbCr

            strSQL = strSQL & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr

            strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO & ", " & vbCr

              strSQL = strSQL & GSTR_TBL_BANCOS & "." & GSTR_TBL_BANCOS_FLD_DESCRIPCION & " AS " &
              GSTR_TBL_RUBRO_FLD_ID_NOMBRE_BANCO & vbCr


            strSQL = strSQL & " FROM " & GSTR_TBL_RUBRO & vbCr

            'Concepto del rubro
            strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
              strSQL = strSQL & "=" & GSTR_TBL_CONCEPTO_RUBRO & "." &
              GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & vbCr

            'Naturaleza del Rubro
            strSQL = strSQL & " INNER JOIN " & GSTR_TBL_NATURALEZA & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID
            strSQL = strSQL & "=" & GSTR_TBL_NATURALEZA & "." & GSTR_TBL_NATURALEZA_FLD_ID & vbCr

            'Bancos
            strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_BANCOS & vbCr
            strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
            strSQL = strSQL & " = " & GSTR_TBL_BANCOS & "." & GSTR_TBL_BANCOS_FLD_ID & vbCr
```

```
    'Moneda
    strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_MONEDA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
    strSQL = strSQL & "=" & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr



    If plngIdDato1 > 0 Then
        strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ROL_RUBRO & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID

        strSQL = strSQL & " WHERE " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_ROL_FLD_ID & "=" &
      plngIdDato1 & vbCr
        strSQL = strSQL & " AND " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & "="
      & plngIdDato2 & vbCr
        '15-Mayo-2004 Solo muestra los activos (no eliminados lógicamente)
        strSQL = strSQL & " AND (" & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=1 OR
      " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & " IS NULL )" & vbCr
    Else
        '15-Mayo-2004 Solo muestra los activos (no eliminados lógicamente)
        strSQL = strSQL & " WHERE " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & "=1
      OR " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_ACTIVO & " IS NULL " & vbCr
    End If




    strSQL = strSQL & " ORDER BY " & GSTR_TBL_CONCEPTO_RUBRO & "." &
      GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION



Case 2
    strSQL = "SELECT " & GSTR_TBL_NATURALEZA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_NATURALEZA_DESCRIPCION & vbCr
    strSQL = strSQL & " FROM " & GSTR_TBL_NATURALEZA & vbCr
    strSQL = strSQL & "ORDER BY " & GSTR_TBL_NATURALEZA_DESCRIPCION

Case 3 'Empleados
    strSQL = "select " & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO_FLD_NOMBRE & vbCr
    strSQL = strSQL & " + ' ' + " & GSTR_TBL_EMPLEADO_FLD_PATERNO & vbCr
    strSQL = strSQL & " + ' ' + " & GSTR_TBL_EMPLEADO_FLD_MATERNO & " as " &
      GSTR_TBL_EMPLEADO_FLD_NOMBRE & vbCr
    strSQL = strSQL & " from " & GSTR_TBL_EMPLEADO & vbCr
    strSQL = strSQL & " where " & GSTR_TBL_EMPLEADO_FLD_ACTIVO & " = " & LNG_ACTIVO & vbCr
    strSQL = strSQL & " order by " & GSTR_TBL_EMPLEADO_FLD_NOMBRE

Case 4 'Centro de Costos

    If plngIdDato1 = 0 Then
        strSQL = "SELECT " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
      & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_TIPO_DISTRIBUCION & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_EMPLEADOS & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
      GSTR_TBL_CENTRO_COSTO_FLD_METROS_CUADRADOS & ", " & vbCr
```

```
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
    GSTR_TBL_AREA & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & vbCr
      strSQL = strSQL & " FROM " & GSTR_TBL_CENTRO_COSTO & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
      strSQL = strSQL & " WHERE " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " = 1 " & vbCr
      strSQL = strSQL & " ORDER BY " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & ",
    " & vbCr
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO
  Else
      strSQL = "SELECT " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
    & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_TIPO_DISTRIBUCION & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_EMPLEADOS & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_METROS_CUADRADOS & ", " & vbCr
            strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
    GSTR_TBL_AREA & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & vbCr
      strSQL = strSQL & " FROM " & GSTR_TBL_CENTRO_COSTO & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
      strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
      strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
      strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
      strSQL = strSQL & " WHERE " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " <> 0 " & vbCr
      strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " <> 3 " & vbCr
      strSQL = strSQL & " ORDER BY " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & ",
    " & vbCr
      strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & ", " & vbCr
      strSQL = strSQL & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO
  End If

Case 5 'Departamentos
  strSQL = "select " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & ", " & vbCr
  strSQL = strSQL & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr
  strSQL = strSQL & " from " & GSTR_TBL_CENTRO_COSTO & vbCr
  strSQL = strSQL & " Where " & GSTR_TBL_CENTRO_COSTO_FLD_ACTIVO & " = " & LNG_ACTIVO & vbCr
  strSQL = strSQL & " ORDER BY " & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO

Case 6 'Oficinas
  If plngIdDato1 > 0 Then
    strSQL = "select " & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & " from " & GSTR_TBL_OFICINA
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO
    strSQL = strSQL & " where " & GSTR_TBL_OFICINA_FLD_ID & " = " & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & "and " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIdDato1
  Else
```

```
        strSQL = "select " & GSTR_TBL_OFICINA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_OFICINA
        strSQL = strSQL & " Order by " & GSTR_TBL_OFICINA_FLD_NOMBRE
    End If



Case 8 'Monedas
    If plngIdDato1 > 0 Then
        strSQL = "select " & GSTR_TBL_MONEDA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_SIGNO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ABREV & vbCr
        strSQL = strSQL & " from " & GSTR_TBL_MONEDA & vbCr
        strSQL = strSQL & " where " & GSTR_TBL_MONEDA_FLD_ID & " = " & plngIdDato1 & vbCr
        strSQL = strSQL & " order by " & GSTR_TBL_MONEDA_FLD_NOMBRE
    Else
        strSQL = "select " & GSTR_TBL_MONEDA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_SIGNO & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_MONEDA_FLD_ABREV & vbCr
        strSQL = strSQL & " from " & GSTR_TBL_MONEDA & vbCr
        strSQL = strSQL & " order by " & GSTR_TBL_MONEDA_FLD_NOMBRE
    End If


Case 9 'Tipo de Cambio
    If plngIdDato1 >= 0 And plngIdDato2 > 0 And plngIdDato3 > 0 Then
        strSQL = "select " & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_MES
        strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_CAMBIO
        strSQL = strSQL & " where " & GSTR_TBL_MONEDA_FLD_ID & " = " & plngIdDato1
        strSQL = strSQL & " AND " & GSTR_TBL_TIPO_CAMBIO_FLD_MES & " = " & plngIdDato2
        strSQL = strSQL & " AND " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO & " = " & plngIdDato3
    Else
        strSQL = "select " & GSTR_TBL_TIPO_CAMBIO_FLD_VALOR
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_ANIO
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CAMBIO_FLD_MES
        strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_CAMBIO

    End If

Case 10 'Prestadores
    strSQL = "SELECT " & GSTR_TBL_PRESTADOR_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_MATERNO & ", " & vbCr

    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & " + ' ' + " & vbCr
    strSQL = strSQL & " ISNULL(" & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", ' ' ) + ' ' + " & vbCr
    strSQL = strSQL & " ISNULL(" & GSTR_TBL_PRESTADOR_FLD_MATERNO & ", ' ') as  " &
      GSTR_TBL_PRESTADOR & ", " & vbCr

    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_FISICA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_RFC & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CURP & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_SEXO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_FECHA_NAC & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_SERVICIO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EDO_CIVIL_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDBANCO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDPLAZA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDSUCURSAL & ", " & vbCr
```

```
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CTA_BANCARIA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_CTA_CLABE & "," & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & "," & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & vbCr



    strSQL = strSQL & " FROM " & GSTR_TBL_PRESTADOR & vbCr

    If plngIdDato1 > 0 Then
       strSQL = strSQL & " WHERE " & GSTR_TBL_PRESTADOR_FLD_ID & "=" & plngIdDato1 & vbCr
       If plngIdDato2 > 0 Then _
          strSQL = strSQL & " AND NOT " & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & " Is Null" &
     vbCr
    Else
       If plngIdDato2 > 0 Then _
          strSQL = strSQL & " WHERE NOT " & GSTR_TBL_PRESTADOR_FLD_IDPRESTADOR_SISE & " Is Null"
     & vbCr
    End If

    strSQL = strSQL & " ORDER BY " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_NOMBRE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_PATERNO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

Case 11 'Areas
    strSQL = "select " & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & " from " & GSTR_TBL_AREA
    strSQL = strSQL & " Order by " & GSTR_TBL_AREA_FLD_NOMBRE

Case 12 'Oficinas-Areas
        strSQL = "SELECT " & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & vbCr
        strSQL = strSQL & " FROM " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & vbCr
        strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
        strSQL = strSQL & " = " & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
        strSQL = strSQL & " ORDER BY" & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OFICINA_AREA & "." & GSTR_TBL_AREA_FLD_ID

Case 13 'Tipo Objeto
        strSQL = "select " & GSTR_TBL_TIPO_OBJETO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION
        strSQL = strSQL & " from " & GSTR_TBL_TIPO_OBJETO
        strSQL = strSQL & " Order by " & GSTR_TBL_TIPO_OBJETO_FLD_DESCRIPCION

Case 14 'Rol
        strSQL = "select " & GSTR_TBL_ROL_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_DESCRIPCION
        strSQL = strSQL & ", " & GSTR_TBL_ROL_FLD_CANTIDAD
        strSQL = strSQL & " from " & GSTR_TBL_ROL
        strSQL = strSQL & " Order by " & GSTR_TBL_ROL_FLD_DESCRIPCION

Case 15 'Pantalla
        strSQL = "select " & GSTR_TBL_PANTALLA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_PANTALLA_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_PANTALLA
        strSQL = strSQL & " Order by " & GSTR_TBL_PANTALLA_FLD_NOMBRE

Case 16 'Rol-Objeto
        strSQL = "SELECT " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_OBJETO_FLD_ID_PWD & ", " &
       vbCr
```

```
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE & ", " & vbCr
        strSQL = strSQL & GSTR_TBL_OBJETO & "." & GSTR_TBL_PANTALLA_FLD_ID & vbCr
        strSQL = strSQL & " FROM " & GSTR_TBL_ROL_OBJETO & vbCr
        strSQL = strSQL & " RIGHT JOIN " & GSTR_TBL_OBJETO & vbCr
        strSQL = strSQL & " ON " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & vbCr
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE

    Case 17 'Objeto
        strSQL = "select " & GSTR_TBL_OBJETO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_OBJETO_FLD_NOMBRE
        strSQL = strSQL & ", " & GSTR_TBL_PANTALLA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_OBJETO_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_OBJETO
        strSQL = strSQL & " Order by " & GSTR_TBL_OBJETO_FLD_NOMBRE

    Case 18 'Banco
        strSQL = "select " & GSTR_TBL_BANCO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_BANCO_FLD_NOMBRE
        strSQL = strSQL & " from " & GSTR_TBL_BANCO
        strSQL = strSQL & " Order by " & GSTR_TBL_BANCO_FLD_NOMBRE

    Case 19 'Status_OP
        strSQL = "SELECT " & GSTR_TBL_STATUS_OP_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_DESCRIPCION
        strSQL = strSQL & " FROM " & GSTR_TBL_STATUS_OP
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_STATUS_OP_FLD_DESCRIPCION


    Case 20 'Orden_Status
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_STATUS_FLD_FECHA
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_STATUS
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 21 'Orden_Pago
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
        strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_RUBRO_FLD_ID
'       strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_DESCRIPCION
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
        strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_ID
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_PAGO
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 22 'Orden_Pago_Cuenta
        strSQL = "select " & GSTR_TBL_ORDEN_PAGO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
        'strSQL = strSQL & ", " & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_GUIAS_AUT_CC_FLD_ID_GUI_CON
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE
        strSQL = strSQL & " from " & GSTR_TBL_ORDEN_PAGO_CUENTA
        strSQL = strSQL & " Order by " & GSTR_TBL_ORDEN_PAGO_FLD_ID

    Case 23 'TipoConsulta
        strSQL = "SELECT " & GSTR_TBL_TIPO_CONSULTA_FLD_ID & vbCr
        strSQL = strSQL & ", " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE & vbCr
        strSQL = strSQL & " FROM " & GSTR_TBL_TIPO_CONSULTA & vbCr
        strSQL = strSQL & " ORDER BY " & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
```

```
        Case 24 'Concepto de rubro
           strSQL = "SELECT " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID_UNIQUE & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_CONCEPTO & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO
           strSQL = strSQL & " FROM " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
           strSQL = strSQL & " ORDER BY " & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID

        Case 25     'Rol - Rubro
           strSQL = "SELECT " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & ", " & vbCr
           strSQL = strSQL & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID & ", " & vbCr
                    strSQL = strSQL & GSTR_TBL_CONCEPTO_RUBRO & "." &
             GSTR_TBL_CONCEPTO_RUBRO_FLD_ID & vbCr
           strSQL = strSQL & " FROM " & GSTR_TBL_ROL_RUBRO & vbCr
           strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
           strSQL = strSQL & " ON " & GSTR_TBL_ROL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
           strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
           strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
             strSQL = strSQL & " ON " & GSTR_TBL_CONCEPTO_RUBRO & "." &
             GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
           strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID



        Case Else
           mobjError.RegistraError 202, GSTR_ERROR_202, STR_FUNCION

     End Select

     'Cargo el recordset
     Set objBD = New SIFIBD.BDAcceso

     Set LoadCatalogo = objBD.rsdDevuelveRecordset(strSQL)

errHandler:
     Select Case Err.Number
        Case 0

        Case Else
           mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

     End Select

     'Libero la memoria
     Set objBD = Nothing

End Function
```

Public Sub RegistrarBitacora(ByVal pstrTabla As String, ByVal plngEmpleado As Long, ByVal plngMovimiento As Long, Optional ByVal pstrConcepto As String, Optional ByVal plngOficina As Long)

Descripción: Función para registrar los movimientos de la operación del sistema de un usuario determinado.

```
     Const STR_FUNCION As String = MSTR_MODULO & ".RegistrarBitacora"

     Dim strSQL As String
     Dim lngID As Long
     Dim strFecha As String
     Dim strHora As String
     Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler

     'strFecha = GetDateFromServer
     strFecha = Format(GetDateFromServer, "yyyy-mm-dd")
```

```
    strHora = Format(GetDateFromServer, "Hh:Mm")

    Set objBD = New SIFIBD.BDAcceso

    lngID = objBD.lngGetNewId(GSTR_TBL_BITACORA, GSTR_TBL_BITACORA_FLD_ID)


    strSQL = "INSERT INTO " & GSTR_TBL_BITACORA & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_BITACORA_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_TABLA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_FECHA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_MOVIMIENTO_FLD_ID & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_HORA & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_CONCEPTO & "," & vbCr
    strSQL = strSQL & GSTR_TBL_BITACORA_FLD_IDOFICINA & ") " & vbCr

    strSQL = strSQL & " VALUES (" & vbCr
    strSQL = strSQL & lngID & ", " & vbCr
    strSQL = strSQL & "'" & UCase$(Trim$(pstrTabla)) & "', " & vbCr
    strSQL = strSQL & "'" & strFecha & "'," & vbCr
    strSQL = strSQL & plngEmpleado & ", " & vbCr
    strSQL = strSQL & plngMovimiento & ", " & vbCr
    strSQL = strSQL & "'" & strHora & "', " & vbCr
    strSQL = strSQL & "'" & pstrConcepto & "'," & vbCr
    strSQL = strSQL & plngOficina & ")" & vbCr


    Call objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select
'</CSCustomCode> 1

    'Libero la memoria
    Set objBD = Nothing

End Sub
```

CLASE EMPLEADO

Public Function blnNuevoEmpleado(ByVal pstrNombre As String, _
ByVal pstrPaterno As String, _
ByVal pstrMaterno As String, _
ByVal pblnActivo As Boolean, _
ByVal pstrLogin As String, _
ByVal pstrPassword As String, _
ByVal plngOficina As Long, _
ByVal plngArea As Long, _
ByVal plngDepto As Long, _
ByVal pblnUpdate As Boolean, _
ByVal pstrContabilidad As String, _
ByVal pstrLogSISE As String, _
ByVal pstrLogSABE As String, _
ByVal pstrPwdFunEsp As String, _
ByVal plngIDRol As Long, _
ByVal plngIDConsulta As Long, _
ByVal plngIDNivelConsulta As Long, _
Optional ByVal plngID As Long) As Boolean

Descripción: Función para registro de Usuarios al sistema

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnNuevoEmpleado"

    Dim strSql As String
    Dim lngIDEmpleado As Long
    Dim objBD As SIFIBD.BDAcceso
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strContabilidad As String
    Dim strPwdFunEsp As String
    Dim objRolEmpleado As SIFISeguridad.Seguridad
    Dim strSql2 As String
    Dim blnCreaRolOficina As Boolean


On Error GoTo errHandler

    'Mando encriptar el password
    Set objEncripta = New xmsEncriptador.Encriptador

    pstrPassword = objEncripta.EncriptaString(pstrPassword)

    Set objBD = New SIFIBD.BDAcceso

    strSql2 = "INSERT INTO " & GSTR_TBL_ROL_OFICINAS & vbCr
    strSql2 = strSql2 & "(" & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql2 = strSql2 & GSTR_TBL_OFICINA_FLD_ID & ")" & vbCr
    strSql2 = strSql2 & " VALUES (" & plngIDRol & ", " & vbCr
    strSql2 = strSql2 & plngOficina & ")"




    If pblnUpdate Then
        strSql = "UPDATE " & GSTR_TBL_EMPLEADO & vbCr
        strSql = strSql & " SET " & vbCr
        strSql = strSql & GSTR_TBL_OFICINA_FLD_ID & "= " & plngOficina & ", " & vbCr
        strSql = strSql & GSTR_TBL_AREA_FLD_ID & "=" & plngArea & ", " & vbCr
```

```
    strSql = strSql & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & "=" & plngDepto & ", " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_NOMBRE & "='" & UCase$(Trim$(pstrNombre)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PATERNO & "='" & UCase$(Trim$(pstrPaterno)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_MATERNO & "='" & UCase$(Trim$(pstrMaterno)) &
             "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOGIN & "='" & UCase$(Trim$(pstrLogin)) & "', " &
             vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" & pstrPassword & "', " & vbCr



    If pstrContabilidad <> vbNullString Then
       strContabilidad = objEncripta.EncriptaString(pstrContabilidad)
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & "='" & strContabilidad & "', " &
             vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & "=NULL," & vbCr
    End If

    'Modificacion 05/05/2003 Agregar usuario de SISE y SABE y Pwd de Funciones Especiales

    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SISE & "='" & UCase$(Trim$(pstrLogSISE))
             & "', " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SABE & "='" & UCase$(Trim$(pstrLogSABE))
             & "', " & vbCr

    If pstrPwdFunEsp <> vbNullString Then
       strPwdFunEsp = objEncripta.EncriptaString(pstrPwdFunEsp)
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & "='" & strPwdFunEsp & "', " &
             vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & "=NULL," & vbCr
    End If


    If pblnActivo Then
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & "=1, " & vbCr
    Else
       strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & "=0, " & vbCr
    End If

    strSql = strSql & GSTR_TBL_EMPLEADO_FLD_FECHA_ACTUALIZACION & "='" & Format(Now,
             "YYYY-MM-DD") & "'"
    strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngID

If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOGIN, GSTR_TBL_EMPLEADO,
     Trim$(pstrLogin), GSTR_TBL_EMPLEADO_FLD_ID, plngID) Then
    If objBD.blnBuscaRegistro(Trim$(pstrLogin), GSTR_TBL_EMPLEADO,
             GSTR_TBL_EMPLEADO_FLD_LOGIN, False) Then
       blnNuevoEmpleado = False
    Else
       If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SISE,
             GSTR_TBL_EMPLEADO, Trim$(pstrLogSISE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
     Then

          If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
             GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
            blnNuevoEmpleado = False
```

```
        Else

            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                    GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID,
                    plngID) Then

                If Trim$(pstrLogSABE) <> vbNullString And
                    objBD.blnBuscaRegistro(Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO,
                    GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
                  blnNuevoEmpleado = False
                Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                   blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                End If
            Else
               blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
               blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)


            End If


        End If

    Else
       If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
             GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
     Then

          If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
              GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
            blnNuevoEmpleado = False

          Else
             blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
             blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
          End If
       Else
             blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
             blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
       End If
    End If
    End If
Else
    If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SISE,
              GSTR_TBL_EMPLEADO, Trim$(pstrLogSISE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
              Then

          If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
              GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
             blnNuevoEmpleado = False
          Else

             If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                     GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID,
                     plngID) Then

                If Trim$(pstrLogSABE) <> vbNullString And
                     objBD.blnBuscaRegistro(Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO,
                     GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
```

```
                    blnNuevoEmpleado = False
                  Else
                    blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                     blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                  End If
               Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
               End If


            End If

         Else
            If Not objBD.blnExisteDatoenTabla(GSTR_TBL_EMPLEADO_FLD_LOG_SABE,
                  GSTR_TBL_EMPLEADO, Trim$(pstrLogSABE), GSTR_TBL_EMPLEADO_FLD_ID, plngID)
          Then

               If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
                  GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
                 blnNuevoEmpleado = False

               Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
               End If
            Else
                  blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                  blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)

            End If
         End If
      End If

      'Actualiza el Perfil del Usuario
         If blnNuevoEmpleado Then
            Set objRolEmpleado = New SIFISeguridad.Seguridad

            objRolEmpleado.Empleado = lngIDEmpleado

            If objRolEmpleado.blnAsignaRolUsuario(plngIDRol, _
                        plngIDConsulta, _
                        plngIDNivelConsulta, _
                        plngID) Then

            End If
            Set objRolEmpleado = Nothing

         End If


   Else

      If objBD.blnBuscaRegistro(Trim$(pstrLogin), GSTR_TBL_EMPLEADO,
          GSTR_TBL_EMPLEADO_FLD_LOGIN, False) Then
         blnNuevoEmpleado = False
      Else
         If Trim$(pstrLogSISE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSISE),
                  GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SISE, False) Then
            blnNuevoEmpleado = False
```

```
Else
   If Trim$(pstrLogSABE) <> vbNullString And objBD.blnBuscaRegistro(Trim$(pstrLogSABE),
         GSTR_TBL_EMPLEADO, GSTR_TBL_EMPLEADO_FLD_LOG_SABE, False) Then
      blnNuevoEmpleado = False
   Else


      lngIDEmpleado = objBD.lngGetNewId(GSTR_TBL_EMPLEADO,
         GSTR_TBL_EMPLEADO_FLD_ID)

      strSql = "INSERT INTO " & GSTR_TBL_EMPLEADO & vbCr
      strSql = strSql & "(" & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_OFICINA_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_AREA_FLD_ID & ", " & vbCr
      strSql = strSql & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_NOMBRE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PATERNO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_MATERNO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOGIN & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PASSWORD & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_ACTIVO & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_FECHA_ACTUALIZACION & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SISE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_LOG_SABE & ", " & vbCr
      strSql = strSql & GSTR_TBL_EMPLEADO_FLD_PWD_FUN_ESP & ") " & vbCr

      strSql = strSql & " VALUES (" & vbCr

      strSql = strSql & lngIDEmpleado & ", " & vbCr
      strSql = strSql & plngOficina & ", " & vbCr
      strSql = strSql & plngArea & ", " & vbCr
      strSql = strSql & plngDepto & ", " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrNombre)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrPaterno)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrMaterno)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrLogin)) & "', " & vbCr
      strSql = strSql & "'" & pstrPassword & "', " & vbCr

      If pblnActivo Then
         strSql = strSql & " 1, " & vbCr
      Else
         strSql = strSql & " 0, " & vbCr
      End If

      strSql = strSql & "'" & Format(Now, "YYYY-MM-DD") & "'," & vbCr

      If pstrContabilidad <> vbNullString Then
         strContabilidad = objEncripta.EncriptaString(pstrContabilidad)
         strSql = strSql & "'" & strContabilidad & "', " & vbCr
      Else
         strSql = strSql & "NULL, " & vbCr
      End If

      strSql = strSql & "'" & UCase$(Trim$(pstrLogSISE)) & "', " & vbCr
      strSql = strSql & "'" & UCase$(Trim$(pstrLogSABE)) & "', " & vbCr

      If pstrPwdFunEsp <> vbNullString Then
         strPwdFunEsp = objEncripta.EncriptaString(pstrPwdFunEsp)
         strSql = strSql & "'" & strPwdFunEsp & "')" & vbCr
```

```
                Else
                    strSql = strSql & "NULL)" & vbCr
                End If


                blnNuevoEmpleado = objBD.blnEjecutaQuery(strSql)
                blnCreaRolOficina = objBD.blnEjecutaQuery(strSql2)
                    If blnNuevoEmpleado Then
                        Set objRolEmpleado = New SIFISeguridad.Seguridad

                        objRolEmpleado.Empleado = lngIDEmpleado

                        If objRolEmpleado.blnAsignaRolUsuario(plngIDRol, _
                                plngIDConsulta, _
                                plngIDNivelConsulta, _
                                lngIDEmpleado) Then

                        End If
                        Set objRolEmpleado = Nothing

                    End If


            End If
        End If
    End If

    End If

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnNuevoEmpleado = False

    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set objEncripta = Nothing

End Function
```

Public Function rsdPerfilEmpleado() As ADODB.Recordset

Descripción: Función para cargar el perfil del Usuario dado su número de clave

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdPerfilEmpleado"

    Dim strSql As String
    Dim objBD As SIFIBD.BDAcceso

On Error GoTo errHandler


    strSql = "SELECT " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & ", " & vbCr
```

```
    strSql = strSql & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_EMPLEADO_ULTIMO_ROL &
                ", " & vbCr
    strSql = strSql & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_CONTABILIDAD & ", "
                & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " &
                vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
                & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_DESCRIPCION & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_CANTIDAD & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_TIPO_CONSULTA
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID


    Set objBD = New SIFIBD.BDAcceso

    Set rsdPerfilEmpleado = objBD.rsdDevuelveRecordset(strSql)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
            Set rsdPerfilEmpleado = Nothing

    End Select

    'Libera la memoria
    Set objBD = Nothing

End Function
```

Public Function rsdListaEmpleados(Optional ByVal plngNumEmpleado As Long) As ADODB.Recordset

Descripción: Función para desplegar una lista de Usuarios

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdListaEmpleados"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo errHandler

    strSql = "SELECT " & GSTR_TBL_EMPLEADO & ".*," & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID & ", " & vbCr
    strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID & ", " &
                vbCr
```

```
    ' strSql = strSql & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_NOMBRE
                    & ", " & vbCr
    strSql = strSql & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_NIVEL_CONSULTA & ", " & vbCr
    strSql = strSql & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_DESCRIPCION_NIVEL & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_DESCRIPCION & ", " & vbCr
    strSql = strSql & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_CANTIDAD & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_TIPO_CONSULTA
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_TIPO_CONSULTA & "." & GSTR_TBL_TIPO_CONSULTA_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_ROL & "." & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " INNER JOIN " & GSTR_TBL_NIVEL_CONSULTA & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." &
            GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA
    strSql = strSql & " = " & GSTR_TBL_NIVEL_CONSULTA & "." &
            GSTR_TBL_NIVEL_CONSULTA_FLD_ID_NIVEL_CONSULTA


    If Not IsMissing(plngNumEmpleado) Then

        If plngNumEmpleado > 0 Then
            strSql = strSql & vbCr & " WHERE " & GSTR_TBL_EMPLEADO & "." &
            GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngNumEmpleado

        End If

    End If

    strSql = strSql & " Order by " & GSTR_TBL_EMPLEADO & "." &
            GSTR_TBL_EMPLEADO_FLD_NOMBRE & "," & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_PATERNO & "," & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_MATERNO

    Set objBD = New SIFIBD.BDAcceso

    Set rsdListaEmpleados = objBD.rsdDevuelveRecordset(strSql)

errHandler:
    Select Case Err.Number
        Case 0

        Case Else
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function ObtieneNombreUsuario(ByVal plngIDUser As Long, ByRef plngIDOficina As Long) As String

Descripción: Función para consultar y obtener los datos personales del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneNombreUsuario"

Dim strSql      As String
Dim objBD       As SIFIBD.BDAcceso
Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneNombreUsuario = vbNullString
    plngIDOficina = -1

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_EMPLEADO_FLD_NOMBRE & " + ' ' + " & _
                     GSTR_TBL_EMPLEADO_FLD_PATERNO & " + ' ' + " & _
                     GSTR_TBL_EMPLEADO_FLD_MATERNO & ", " & _
                     GSTR_TBL_OFICINA_FLD_ID & " from " & _
                     GSTR_TBL_EMPLEADO & " where  " & _
                     GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser


    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)
    If Not rsEmpleado Is Nothing Then
       If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
          rsEmpleado.MoveFirst
          ObtieneNombreUsuario = rsEmpleado(0)
          plngIDOficina = rsEmpleado(1)
          rsEmpleado.Close
       End If
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          ObtieneNombreUsuario = vbNullString
          plngIDOficina = -1
          objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function
```

```
Public Function ObtieneRolUsuario(ByVal plngIDUser As Long) As Long
'Descripion:    Funcion para obtener el Rol de un usuario

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneRolUsuario"

Dim strSql      As String
Dim objBD       As SIFIBD.BDAcceso
Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneRolUsuario = -1

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_ROL_FLD_ID
    strSql = strSql & " from " & GSTR_TBL_ROL_EMPLEADO
    strSql = strSql & " where " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser

    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
        If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
            rsEmpleado.MoveFirst
            ObtieneRolUsuario = rsEmpleado.Fields(GSTR_TBL_ROL_FLD_ID)
            rsEmpleado.Close
        End If
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ObtieneRolUsuario = -1
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function


Public Function ObtieneNivelConsulta(ByVal plngIDUser As Long) As Long
'Descripion:    Funcion para obtener el Tipo de consulta que puede realizar

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneNivelConsulta"

            Dim strSql      As String
            Dim objBD       As SIFIBD.BDAcceso
            Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneNivelConsulta = -1

    Set objBD = New SIFIBD.BDAcceso
```

```
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA
    strSql = strSql & " from " & GSTR_TBL_ROL_EMPLEADO
    strSql = strSql & " where " & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser

    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
       If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
          rsEmpleado.MoveFirst
          ObtieneNivelConsulta = rsEmpleado.Fields(GSTR_TBL_ROL_EMPLEADO_IDNIVELCONSULTA)
          rsEmpleado.Close
       End If
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          ObtieneNivelConsulta = -1
          objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function


Public Function ObtieneUsuarioDepartamento(ByVal plngIDUser As Long) As String
'Descripción:    Funcion para obtener el departamento de un usuario

    Const STR_FUNCION As String = MSTR_MODULO & ".ObtieneUsuarioDepartamento"

          Dim strSql      As String
          Dim objBD       As SIFIBD.BDAcceso
          Dim rsEmpleado  As ADODB.Recordset

    On Error GoTo ErrorHandler

    ObtieneUsuarioDepartamento = vbNullString

    Set objBD = New SIFIBD.BDAcceso
    Set rsEmpleado = New ADODB.Recordset

    strSql = "Select " & GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & " from " & _
                    GSTR_TBL_CENTRO_COSTO & ", " & _
                    GSTR_TBL_EMPLEADO & " where " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_OFICINA_FLD_ID & " = " & _
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID & " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_AREA_FLD_ID & " = " & _
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID & " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & " = " &
_
                    GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO &
                    " and " & _
                    GSTR_TBL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID & " = " & plngIDUser
```

```
    Set rsEmpleado = objBD.rsdDevuelveRecordset(strSql)

    If Not rsEmpleado Is Nothing Then
        If Not rsEmpleado.EOF And Not rsEmpleado.BOF Then
            rsEmpleado.MoveFirst
            ObtieneUsuarioDepartamento =
             rsEmpleado.Fields(GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO)
            rsEmpleado.Close
        End If
    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ObtieneUsuarioDepartamento = vbNullString
            objError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
    Set rsEmpleado = Nothing

End Function
```

CLASE MANEJO DE ERRORES

```
Public Sub RegistraError(ByVal lngErrNum As Long, _
                    ByVal strDescripcion As String, _
                    ByVal pstrModulo As String, _
                    Optional ByVal pstrUserName As String = vbNullString, _
                    Optional ByVal pstrLogin As String = vbNullString, _
                    Optional ByVal plngAbreLog As Long = 0)
'Descripción:       Funcion para obtener la cadena de conexion
'                   de un archivo de texto y guardarla en una
'                   propiedad que es global y accesible de cualquier
'                   lugar del proyecto

   Dim intIDFile       As Long

   intIDFile = FreeFile
   Open App.Path + "\SIF_Errores.log" For Append As #intIDFile

   Print #intIDFile, vbCrLf & vbCrLf
   Print #intIDFile, String(50, "$")
   Print #intIDFile, "NÚMERO DE ERROR  : " & lngErrNum
   Print #intIDFile, "DESCRIPCIÓN      : " & strDescripcion & vbCrLf
   Print #intIDFile, "ORIGEN           : " & pstrModulo
   Print #intIDFile, "FECHA            : " & Now
   Print #intIDFile, String(50, "$")

   Close intIDFile

End Sub

Private Sub CreaLog(pintIdFile As Integer)
'Descripción:       Funcion para obtener abrir el archivo Log y escribir la fecha
'                   y hora del acceso

          Dim intFreeFile As Integer
          Dim strConn As String


   Open App.Path + "\SIF_errores.log" For Append As #pintIdFile

   Print #pintIdFile, vbNullString
   Print #pintIdFile, vbNullString
   Print #pintIdFile,
"++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++"

   Print #pintIdFile, "Se inicio el sistema SIF el dia ", Format(Date, "dddd dd/mm/yyyy"), Time
   Print #pintIdFile, vbNullString

   Close pintIdFile

End Sub
```

CLASE ORDEN DE PAGO

Public Function blnActualizaLoteTransf(ByVal plngOP As Long, ByVal pstrNumero As String, ByVal plngNoLote As Long, ByVal pstrIDCuentaBancaria As String, ByVal pintIDOperacion As String, ByVal plngSucursal As Long) As Boolean

'**Descripción:** Se actualiza en cada una de las OP, transferencias, del archivo (lote)
' en el que se está enviando al banco (liberacion).

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaLoteTransf"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo errHandler


    strSQL = "UPDATE " & GSTR_TBL_REF_TRAN_AUT
    strSQL = strSQL & " SET " & GSTR_TBL_REF_TRAN_AUT_FLD_LOTE & " = " & plngNoLote & ", "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA & " = " &
        pstrIDCuentaBancaria & ", "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_OPERA & " = " & pintIDOperacion & ", " &
        strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL & " = " & plngSucursal
    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & " = " & plngOP
    strSQL = strSQL & " AND " & GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON & " = " & pstrNumero &
        vbCr

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaLoteTransf = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

       Case 0
          'Do nothing
       Case Else
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
          blnActualizaLoteTransf = False
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnDesActualizaLoteTransf(ByVal plngOP As Long, ByVal pstrNumero As String) As Boolean

'Descripción: Se inicializa los valores de las transferencias a Nulos para que puedan ser
'liberadas nuevamente

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnDesActualizaLoteTransf"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso
```

```
    On Error GoTo errHandler


    strSQL = "UPDATE " & GSTR_TBL_REF_TRAN_AUT
    strSQL = strSQL & " SET " & GSTR_TBL_REF_TRAN_AUT_FLD_LOTE & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_OPERA & " = NULL, "
    strSQL = strSQL & GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL & " = NULL" & vbCr
    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & " = " & plngOP
   ' strSQL = strSQL & " AND " & GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON & " = " & pstrNumero &
            vbCr

    Set objBD = New SIFIBD.BDAcceso

    blnDesActualizaLoteTransf = objBD.blnEjecutaQuery(strSQL)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnDesActualizaLoteTransf = False
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnActualizaOrdenPago(ByVal pstrNumero As String, ByVal pdblImporte As Double, ByVal plngTipoPago As Long, ByVal plngCuentaBancaria As Long, ByVal pstrDescripcion As String, ByVal pstrNumeroAnterior As String) As Boolean

Descripción: Función para actualizar la información de una Orden de Pago.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaOrdenPago"

    Dim strSQL As String
    Dim strSQLImputacion As String
    Dim objBD As SIFIBD.BDAcceso


On Error GoTo errHandler

    strSQL = "INSERT INTO " & GSTR_TBL_REFERENCIAS & " (" & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPRESO & "," & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_ID & ")" & vbCr

    strSQL = strSQL & " Values (" & vbCr
    strSQL = strSQL & "'" & Trim$(pstrNumero) & "', " & vbCr
```

```
    strSQL = strSQL & plngCuentaBancaria & ", " & vbCr
    strSQL = strSQL & plngTipoPago & ", " & vbCr
    strSQL = strSQL & pdblImporte & "," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrDescripcion)) & "', " & vbCr
    strSQL = strSQL & "0," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrNumeroAnterior)) & "')"

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaOrdenPago = objBD.blnEjecutaQuery(strSQL)



errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            'mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function
```

Public Function blnActualizaOrdenSISE(ByVal plngOrden As Long, ByVal plngEjercicio As Long, ByVal plngCaratula As Long, ByVal plngAsiento As Long, ByVal plngMovimiento As Long) As Boolean

Descripción: Actualizar la información de una orden importada de los sistemas centrales.

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaOrdenSISE"
    Const LNG_VALOR_SISE As Long = 1

    Dim strSQL          As String
    Dim objBD           As SIFIBD.BDAcceso
    Dim strTransaccion  As String
    Dim strSQLOP        As String
  '  Dim strSQLMov       As String

On Error GoTo errHandler

    strSQL = strSQL & " BEGIN TRAN OPERA " & vbCr
    strSQL = strSQL & " INSERT INTO " & GSTR_TBL_ORDEN_SISE
    strSQL = strSQL & " (" & GSTR_TBL_ORDEN_SISE_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_SISE_FLD_CARATULA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_SISE_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_MOVIMIENTOS_SISE_FLD_ID & ")" & vbCr
    strSQL = strSQL & " VALUES ( " & plngAsiento
    strSQL = strSQL & ", " & plngCaratula
    strSQL = strSQL & ", '" & Format(GetDateFromServer, "YYYY-MM-DD") & "'"
    strSQL = strSQL & ", " & plngOrden
    strSQL = strSQL & ", " & plngEjercicio
    strSQL = strSQL & ", " & plngMovimiento & ");" & vbCr
    strSQL = strSQL & "UPDATE " & GSTR_TBL_ORDEN_PAGO
    strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" & plngAsiento & ", "
```

```
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula
    strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" & plngOrden
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & "=" & plngEjercicio & ";" &
          vbCr
    strSQL = strSQL & " IF @@ERROR = 0 COMMIT TRAN OPERA ELSE ROLLBACK TRAN OPERA " &
          vbCr

    Set objBD = New SIFIBD.BDAcceso
    blnActualizaOrdenSISE = objBD.blnEjecutaQuery(strSQL)
    Set objBD = Nothing

errHandler:
    Select Case Err.Number

       Case 0
          'Do nothing
       Case Else
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
          blnActualizaOrdenSISE = False
          Call objBD.blnRollBack(strTransaccion)
    End Select
'</CSCustomCode> 1

    'Libero la memoria
    Set objBD = Nothing
End Function
```

Public Function blnActualizaStatus(ByVal plngOrden As Long, ByVal plngEjercicio As Long, ByVal plngStatus As Long, ByVal plngIDEmpleado As Long, Optional ByVal blnEliminar As Boolean) As Boolean

Descripción: Actualiza el estatus de una orden cuando se autoriza o cancela

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaStatus"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String
    Dim strSQLOPStatus As String

On Error GoTo errHandler

    'agrego un registro a la tabla de status de la OP
    strSQL = "INSERT INTO " & GSTR_TBL_ORDEN_STATUS & vbCr
    strSQL = strSQL & "(" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", "
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_STATUS_OP_FLD_ID & ", "
    strSQL = strSQL & GSTR_TBL_ORDEN_STATUS_FLD_FECHA & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_EMPLEADO_FLD_ID & ") " & vbCr
    strSQL = strSQL & " VALUES " & vbCr
    strSQL = strSQL & "(" & plngEjercicio & ", "
    strSQL = strSQL & plngOrden & ", "
    strSQL = strSQL & plngStatus & ", "
    strSQL = strSQL & "'" & Format(Now, "YYYY-MM-DD h:m:s") & "', " & vbCr
    strSQL = strSQL & plngIDEmpleado & ") " & vbCr


    Set objBD = New SIFIBD.BDAcceso

    If Not blnEliminar Then
```

```
        'Actualizo el último status de la OP
        strSQLOPStatus = "UPDATE " & GSTR_TBL_ORDEN_PAGO & vbCr
        strSQLOPStatus = strSQLOPStatus & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
                         & "=" & plngStatus & vbCr
        strSQLOPStatus = strSQLOPStatus & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" &
                         plngOrden & vbCr
        strSQLOPStatus = strSQLOPStatus & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & "="
                         & plngEjercicio


        If objBD.blnEjecutaQuery(strSQL) Then
           blnActualizaStatus = objBD.blnEjecutaQuery(strSQLOPStatus)
        Else
           blnActualizaStatus = False
        End If

     Else
        blnActualizaStatus = objBD.blnEjecutaQuery(strSQL)

     End If

errHandler:
     Select Case Err.Number

        Case 0
           'Do nothing
        Case Else
           mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
           blnActualizaStatus = False

     End Select
'</CSCustomCode> 1

     'Libero la memoria
     Set objBD = Nothing

End Function
```

Public Function blnGuardarOrdenPagoLIQ(ByVal plngOP As Long, ByVal plngEjercicio As Long, ByVal plngLiq As Long)

Descripción: Se guarda el Recibo asignado a la OP (Devolución de Primas)

```
     Const STR_FUNCION As String = MSTR_MODULO & ".blnGuardarOrdenPagoLIQ"
     Dim strSQL        As String
     Dim objBD         As SIFIBD.BDAcceso

     On Error GoTo ErrorHandler

     Set objBD = New SIFIBD.BDAcceso

     strSQL = "Insert Into " & GSTR_TBL_ORDEN_PAGO_LIQ & " (" & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_IDOP & ", " & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_EJERCICIO & ", " & _
                      GSTR_TBL_ORDEN_PAGO_LIQ_FLD_IDLIQ & ")"

     strSQL = strSQL & " Values (" & plngOP & ", " & _
                       plngEjercicio & ", " & _
                       plngLiq & ")"
```

```
    blnGuardarOrdenPagoLIQ = objBD.blnEjecutaQuery(strSQL)


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnGuardarOrdenPagoLIQ = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing
End Function
```

Private Function lngObtenIDOP()

Descripción: Función utilizada para obtner un nuevo número de Orden de Pago

```
    Const STR_FUNCION As String = MSTR_MODULO & ".lngObtenIDOP"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSQL As String
    Dim rsdOP As ADODB.Recordset

On Error GoTo errHandler

    strSQL = "SELECT MAX(" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ") AS " &
GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", " & vbCr
    strSQL = strSQL & "getdate()" & vbCr
    strSQL = strSQL & " FROM " & GSTR_TBL_ORDEN_PAGO & vbCr


    Set objBD = New SIFIBD.BDAcceso

    Set rsdOP = objBD.rsdDevuelveRecordset(strSQL)

    If Not (rsdOP.EOF And rsdOP.BOF) Then

        If rsdOP(GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO) <> Year(rsdOP(1)) Then
            lngObtenIDOP = 1
        End If

    End If

    If lngObtenIDOP <> 1 Then

        lngObtenIDOP = objBD.lngGetNewId(GSTR_TBL_ORDEN_STATUS,
                                GSTR_TBL_ORDEN_PAGO_FLD_ID)
    End If

errHandler:
    Select Case Err.Number

        Case 0
```

```
        'Do nothing
      Case Else
        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select
'</CSCustomCode> 1

   'Libero la memoria
   Set objBD = Nothing
   Set rsdOP = Nothing

End Function


Public Function NuevaOrden(ByVal plngEjercicio As Long, ByVal plngOficina As Long, ByVal plngArea As
        Long, ByVal plngDepto As Long, ByVal plngBeneficiario As Long, ByVal plngRubro As Long, ByVal
        pstrSolicita As String, ByVal datFechaSolicitud As Date, ByRef plngIDOP As Long, _
            ByVal pstrObservaciones As String, ByVal pstrLeyendaCheque As String, ByVal plngIdMoneda
        As Long, ByVal pdblTipoCambio As Double, ByVal pblnOP As Boolean, Optional ByVal
        plngTraspaso As Long, Optional ByVal plngEjercicioTraspaso As Long, _
            Optional ByVal plngIDBanco As Long, Optional ByVal plngIDPlaza As Long, Optional ByVal
        plngIDSucursal As Long, Optional ByVal pstrHora As String, Optional ByVal pintFormaPago As
        Integer, Optional ByVal pdblImporteTotal As Double = 0) As Boolean


'Descripción:       Funcion Crear una Nueva OP en la BD

   Const STR_FUNCION As String = MSTR_MODULO & ".NuevaOrden"
   Const INT_ID_STATUS_PENDIENTE As Integer = 1

Dim strSQL         As String
Dim objBD          As SIFIBD.BDAcceso
Dim rdsTransOP     As ADODB.Recordset
Dim intIdOp        As Integer
Dim strTansaccion  As String
Dim lngErrorSQL    As Long

On Error GoTo ErrorHandler

   Set objBD = New SIFIBD.BDAcceso

   NuevaOrden = False

   'intIdOp = lngObtenIDOP()
   'plngIDOP = intIdOp

   strSQL = "DECLARE @IDOP       INT " & vbCr
   strSQL = strSQL & "DECLARE @ERRORES    INT" & vbCr
   strSQL = strSQL & "SET NOCOUNT ON " & vbCr
   strSQL = strSQL & "SET @ERRORES = 0 " & vbCr & vbCr
   strSQL = strSQL & "BEGIN TRAN NuevaOrdenPago" & vbCr
   strSQL = strSQL & "SELECT @IDOP = ISNULL(MAX(IDOP),0) + 1 FROM ORDEN_STATUS" & vbCr

   'Orden_Pago
   strSQL = strSQL & "INSERT INTO " & GSTR_TBL_ORDEN_PAGO
   strSQL = strSQL & " ( " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
   strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr
   strSQL = strSQL & ", " & GSTR_TBL_OFICINA_FLD_ID
   strSQL = strSQL & ", " & GSTR_TBL_AREA_FLD_ID
   strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
   strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR_FLD_ID
```

```
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_LEYENDACHEQUE
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
strSQL = strSQL & ", " & GSTR_TBL_MONEDA_FLD_ID & vbCr

strSQL = strSQL & ", " & GSTR_TBL_TRASPASOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_TRASPASOS_FLD_EJERCICIO
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP
'Nuevos campos
strSQL = strSQL & ", " & GSTR_TBL_BANCOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_PLAZA_FLD_ID_PLAZA
strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
strSQL = strSQL & ", " & "IdTipo_Pago"
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_REG_HOR
strSQL = strSQL & ", " & "Importe)" & vbCr

strSQL = strSQL & " Values (" & plngEjercicio
strSQL = strSQL & ", @IDOP "
strSQL = strSQL & ", " & plngOficina
strSQL = strSQL & ", " & plngArea
strSQL = strSQL & ", " & plngDepto & vbCr
strSQL = strSQL & ", " & plngBeneficiario
strSQL = strSQL & ", " & pstrSolicita
strSQL = strSQL & ", '" & Format(datFechaSolicitud, "yyyy-mm-dd Hh:Nn:Ss")
strSQL = strSQL & "', '" & pstrObservaciones
strSQL = strSQL & "', '" & pstrLeyendaCheque
strSQL = strSQL & "', " & pdblTipoCambio
strSQL = strSQL & ", " & INT_ID_STATUS_PENDIENTE
strSQL = strSQL & ", " & plngIdMoneda

'Verifico si se está mandando los datos del traspaso
If plngTraspaso > 0 Then
   strSQL = strSQL & ", " & plngTraspaso
   strSQL = strSQL & ", " & plngEjercicioTraspaso
Else
   strSQL = strSQL & ", NULL"
   strSQL = strSQL & ", NULL"

End If

If pblnOP Then
   strSQL = strSQL & ", 0" & vbCr
Else
   strSQL = strSQL & ", 1" & vbCr
End If

'Nuevos campos
strSQL = strSQL & ", " & plngIDBanco
strSQL = strSQL & ", " & plngIDPlaza
strSQL = strSQL & ", " & plngIDSucursal
strSQL = strSQL & ", " & pintFormaPago
strSQL = strSQL & ", '" & Format(pstrHora, "yyyy-mm-dd h:m:s") & "'"
strSQL = strSQL & ", " & pdblImporteTotal & ")" & vbCr

strSQL = strSQL & "IF @ERRORES = 0 " & vbCr
strSQL = strSQL & "BEGIN" & vbCr
'Orden_Status
```

```
    strSQL = strSQL & "Insert Into " & GSTR_TBL_ORDEN_STATUS
    strSQL = strSQL & " (" & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_STATUS_OP_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_STATUS_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_EMPLEADO_FLD_ID & ")" & vbCr

    strSQL = strSQL & " Values (" & plngEjercicio
    strSQL = strSQL & ", @IDOP"
    strSQL = strSQL & ", 1, '"
    strSQL = strSQL & Format(datFechaSolicitud, "yyyy-mm-dd Hh:Nn:Ss") & "'"
    strSQL = strSQL & ", " & pstrSolicita & ")" & vbCr
    strSQL = strSQL & "END" & vbCr & vbCr


    strSQL = strSQL & "SELECT @IDOP 'IDOP', @ERRORES 'ERROR'" & vbCr
    strSQL = strSQL & "SET NOCOUNT OFF" & vbCr

    'objBD.blnEjecutaQuery (strSQL)
    'strTansaccion = objBD.strBeginTrans

    Set rdsTransOP = objBD.rsdDevuelveRecordset(strSQL)
    If Not rdsTransOP Is Nothing Then
       If Not rdsTransOP.EOF And Not rdsTransOP.BOF Then
          plngIDOP = rdsTransOP.Fields("IDOP")
          lngErrorSQL = rdsTransOP.Fields("ERROR")
       End If
    End If

    If lngErrorSQL = 0 Then
       NuevaOrden = True
       strSQL = "COMMIT TRAN NuevaOrdenPago"
       objBD.blnEjecutaQuery (strSQL)
    Else   'Hubo un error
       NuevaOrden = False
       strSQL = "ROLLBACK TRAN NuevaOrdenPago"
       objBD.blnEjecutaQuery (strSQL)
    End If


ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          strSQL = "ROLLBACK TRAN NuevaOrdenPago"
          objBD.blnEjecutaQuery (strSQL)

          NuevaOrden = False
          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function

Public Function ImputacionOrden(ByVal plngEjercicio As Long, _
                                ByRef plngIDOP() As Long, _
                                ByVal plngCuentaBancaria As Long, _
                                ByVal plngTipoPago As Long, _
```

```
                    ByVal pstrNumero As String, _
                    ByVal pdblImporte As Double, _
                    ByVal plngEmpleado As Long, _
                    ByVal pstrDescripcion As String, _
                    ByVal plngIDTraspaso As Long, _
                    ByVal plngEjercicioTraspaso As Long, _
                    ByVal pblnOP As Boolean) As Boolean


'Descripción:      Funcion para realizar la contabilización de una Orden o mas órdenes

    Const STR_FUNCION As String = MSTR_MODULO & ".ImputacionOrden"

    Const LNG_STATUS_IMPUTADO As Long = 4
    Const LNG_TRASPASO_PERIODO As Long = 1
    Const LNG_STATUS_TRASPASO_PENDIENTE As Long = 1

    Dim strSQL              As String
    Dim objBD               As SIFIBD.BDAcceso
    Dim intIdConsolida      As Integer
    Dim strFecha As String
    Dim strTransaccion As String
    Dim objTraspaso As Traspasos
    Dim blnActualizaOP As Boolean
    Dim strOP As String
    Dim blnOP As Boolean
    Dim lngCont As Long
    Dim blnEjecutaStatus As Boolean


On Error GoTo ErrorHandler

    strFecha = GetDateFromServer

    strSQL = "INSERT INTO " & GSTR_TBL_REFERENCIAS & " (" & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_TIPO_PAGO_FLD_ID & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION & ", " & vbCr
    strSQL = strSQL & GSTR_TBL_REFERENCIAS_FLD_IMPRESO & ")" & vbCr

    strSQL = strSQL & " Values (" & vbCr
    strSQL = strSQL & "'" & Trim$(pstrNumero) & "', " & vbCr
    strSQL = strSQL & plngCuentaBancaria & ", " & vbCr
    strSQL = strSQL & plngTipoPago & ", " & vbCr
    strSQL = strSQL & pdblImporte & "," & vbCr
    strSQL = strSQL & "'" & Trim$(UCase$(pstrDescripción)) & "', " & vbCr
    strSQL = strSQL & "0)"


    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans

    'si se pudo ejecutar la insercion
    If objBD.blnEjecutaQuery(strSQL) Then

        For lngCont = 0 To UBound(plngIDOP)
```

```
    strOP = "INSERT INTO " & GSTR_TBL_ORDEN_PAGO_REFERENCIAS & vbCr
    strOP = strOP & "(" & GSTR_TBL_REFERENCIAS_FLD_NUMERO & ", " & vbCr
    strOP = strOP & GSTR_TBL_ORDEN_PAGO_FLD_ID & ", " & vbCr
    strOP = strOP & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & ", " & vbCr
    strOP = strOP & GSTR_TBL_CUENTAS_BANCARIAS_FLD_ID & ", " & vbCr
    strOP = strOP & GSTR_TBL_TIPO_PAGO_FLD_ID & ") " & vbCr

    'Valores
    strOP = strOP & " VALUES (" & vbCr
    strOP = strOP & "'" & Trim$(pstrNumero) & "', " & vbCr
    strOP = strOP & plngIDOP(lngCont) & ", " & vbCr
    strOP = strOP & plngEjercicio & ", " & vbCr
    strOP = strOP & plngCuentaBancaria & ", " & vbCr
    strOP = strOP & plngTipoPago & ") " & vbCr

    If objBD.blnEjecutaQuery(strOP) Then
       blnOP = True
    Else
       blnOP = False
       Exit For

    End If


Next

If blnOP Then

    If pblnOP Then
       Set objTraspaso = New Traspasos

       If plngIDTraspaso > 0 Then
          blnActualizaOP = objTraspaso.blnModificarStatusTraspaso(plngIDTraspaso,
              plngEjercicioTraspaso, LNG_STATUS_TRASPASO_PENDIENTE, plngEmpleado,
              LNG_TRASPASO_PERIODO, True, False)
       Else
          blnActualizaOP = True
       End If
    Else
       blnActualizaOP = True
    End If

    If blnActualizaOP Then

       For lngCont = 0 To UBound(plngIDOP)
          If blnActualizaStatus(plngIDOP(lngCont), plngEjercicio, LNG_STATUS_IMPUTADO,
              plngEmpleado) Then
             blnEjecutaStatus = True

          Else
             blnEjecutaStatus = False
             Exit For
          End If

       Next

    End If

End If
```

```
        If blnEjecutaStatus Then
            ImputacionOrden = objBD.blnCommitTrans(strTransaccion)

        Else
            ImputacionOrden = False
            Call objBD.blnRollBack(strTransaccion)
        End If


    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            ImputacionOrden = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
            Call objBD.blnRollBack(strTransaccion)

    End Select

    Set objBD = Nothing
    Set objTraspaso = Nothing

End Function


Public Function AutorizacionOrden(ByVal plngIDOP As Long, _
                                  ByVal plngEjercicio As Long, _
                                  ByVal pstrComentarios As String, _
                                  ByVal pblnAutorizado As Boolean, _
                                  ByVal plngIDUser As Long) As Boolean
'Descripción:       Funcion para autorizar una OP


    Const STR_FUNCION As String = MSTR_MODULO & ".AutorizacionOrden"

    Const LNG_STATUS_AUTORIZADO As Long = 2
    Const LNG_STATUS_RECHAZADO As Long = 3

Dim strSQL          As String
Dim objBD           As SIFIBD.BDAcceso
Dim lngIDStatus     As Long
Dim strTransaccion As String
Dim blnResultado As Boolean

On Error GoTo ErrorHandler

    If pblnAutorizado Then
        lngIDStatus = LNG_STATUS_AUTORIZADO
    Else
        lngIDStatus = LNG_STATUS_RECHAZADO
    End If


    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans
```

```
    If Not pblnAutorizado Then

        strSQL = "UPDATE " & GSTR_TBL_ORDEN_PAGO & " SET " & _
                        GSTR_TBL_ORDEN_PAGO_FLD_RECHAZADO_POR & " = '" & pstrComentarios & "' "
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & "=" &
            LNG_STATUS_RECHAZADO
        strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " &
            plngEjercicio & " AND " & _
                                GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngIDOP & vbCr




        If objBD.blnEjecutaQuery(strSQL) Then
            blnResultado = True

        Else
            blnResultado = False
        End If

    Else
        blnResultado = True
    End If

    If blnResultado Then

        If pblnAutorizado Then

            If blnActualizaStatus(plngIDOP, plngEjercicio, lngIDStatus, plngIDUser) Then
                AutorizacionOrden = objBD.blnCommitTrans(strTransaccion)


            Else
                objBD.blnRollBack strTransaccion
            End If

        Else
            If blnActualizaStatus(plngIDOP, plngEjercicio, lngIDStatus, plngIDUser, True) Then
                AutorizacionOrden = objBD.blnCommitTrans(strTransaccion)


            Else
                objBD.blnRollBack strTransaccion
            End If

        End If

    Else
        objBD.blnRollBack strTransaccion

    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
```

```
        objBD.blnRollBack strTransaccion

        AutorizacionOrden = False

        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION

    End Select

    Set objBD = Nothing

End Function


Public Function ConsultaOrden(ByVal plngOrden As Long, _
                              ByVal plngEjercicio As Long) As ADODB.Recordset
'Descripción:       Funcion para realizar una consulta de una OP especifica

    Const STR_FUNCION As String = MSTR_MODULO & ".ConsultaOrden"

        Dim strSQL      As String
        Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT Distinct " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ID_PRESTADOR & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
                GSTR_TBL_OFICINA
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
        GSTR_TBL_AREA
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID & vbCr
    'prestador
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE
    strSQL = strSQL & " + ' ' + ISNULL(" & GSTR_TBL_PRESTADOR & "." &
        GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ' )"
    strSQL = strSQL & " + ' ' + ISNULL(" & GSTR_TBL_PRESTADOR & "." &
        GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ' )" & " AS " & GSTR_TBL_PRESTADOR & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TRASPASOS_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_TRASPASOS_FLD_EJERCICIO

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA & " as " & GSTR_TBL_EMPLEADO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES & vbCr
```

```
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TIPO_PAGO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & " AS MONEDA_INGRESO"
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID
    'strSQL = strSQL & ", SUM( " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE & ")  AS " &
            GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA & ".ID_Gui_Con" & vbCr

    'FROM
    strSQL = strSQL & " FROM " & GSTR_TBL_ORDEN_PAGO & vbCr
    'Orden Pago - Cuenta
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ORDEN_PAGO_CUENTA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID &
            vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & vbCr
    'Movimientos Contables
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MOVTOS_CONTABLES
    strSQL = strSQL & " ON " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO_CUENTA + "." +
            GSTR_TBL_ORDEN_PAGO_FLD_ID
    'Guia Rubros
    strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_GUIA_RUBROS
    strSQL = strSQL & " ON " & GSTR_TBL_GUIA_RUBROS & "." &
            GSTR_TBL_GUIA_RUBROS_FLD_ID_GUI_CON
    strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO_CUENTA & "." &
            GSTR_TBL_GUIA_RUBROS_FLD_ID_GUI_CON
    'Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." & GSTR_TBL_RUBRO_FLD_ID &
            vbCr
    'Conceptos del Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CONCEPTO_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_CONCEPTO_RUBRO & "." &
GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
```

```
strSQL = strSQL & " = " & GSTR_TBL_RUBRO & "." & GSTR_TBL_CONCEPTO_RUBRO_FLD_ID
'Proveedor
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_PRESTADOR & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_ID &
    vbCr
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID &
    vbCr
'Moneda
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MONEDA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
'Centros de Costos
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_CENTRO_COSTO & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CENTRO_COSTO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
strSQL = strSQL & " = " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
'Oficinas
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr
'Areas
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_CENTRO_COSTO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & vbCr
'WHERE
strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOrden & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio & vbCr
strSQL = strSQL & " AND " & GSTR_TBL_CONCEPTO_RUBRO & "." &
    GSTR_TBL_CONCEPTO_RUBRO_FLD_IND_IMPUESTO & " = '" &
    GSTR_CONCEPTO_RUBRO_NO_INDICA_IMPUESTO & "'"

'Cláusla Group
strSQL = strSQL & " GROUP BY " & vbCr
strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_RUBRO_FLD_ID_MONEDA
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_ID_PRESTADOR & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PRESTADOR_FLD_ID & vbCr
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_MATERNO
    & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TRASPASOS_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_TRASPASOS_FLD_EJERCICIO
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_SOLICITA & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
```

```
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_FECHA_SOLICITUD
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_TIPO_PAGO_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_CUENTA_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_NATURALEZA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_CUENTA & ".ID_Gui_Con" & vbCr

    'ORDER BY
    strSQL = strSQL & " ORDER BY " & GSTR_TBL_MOVTOS_CONTABLES & "." &
          GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO

    Set ConsultaOrden = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          Set ConsultaOrden = Nothing

          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function


Public Function ConsultaOPImprimir(ByVal plngOrden As Long, _
                        ByVal plngEjercicio As Long) As ADODB.Recordset
'Descripción:    Funcion para realizar una consulta de una OP especifica
'                y obtener los datos necesarios para la impresión
    Const STR_FUNCION As String = MSTR_MODULO & ".ConsultaOPImprimir"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler
```

```
    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_REFERENCIAS_FLD_IMPORTE & " AS IMPORTEOP" & vbCr

    strSQL = strSQL & " FROM " & GSTR_TBL_MOVTOS_CONTABLES

    'Rubro
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_RUBRO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO & vbCr

    'Orden Pago
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_ORDEN_PAGO & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & vbCr

    'Moneda
    strSQL = strSQL & " INNER JOIN " & GSTR_TBL_MONEDA & vbCr
    strSQL = strSQL & " ON " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr

    'Referencias
'   strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_REFERENCIAS & vbCr
'   strSQL = strSQL & " ON " & GSTR_TBL_REFERENCIAS & "." &
            GSTR_TBL_REFERENCIAS_FLD_IDOP
'   strSQL = strSQL & " = " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & vbCr

    strSQL = strSQL & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOrden & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio & vbCr

    strSQL = strSQL & " GROUP BY " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." & G
            STR_TBL_MOVTOS_CONTABLES_FLD_MONEDA & vbCr
```

```
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_CONTABLE
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_DESCRIPCION & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ID_TIPO_CAMBIO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_REFERENCIAS_FLD_IMPORTE & vbCr

    strSQL = strSQL & "Order by " & GSTR_TBL_MOVTOS_CONTABLES & "." &
            GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO

    Set ConsultaOPImprimir = objBD.rsdDevuelveRecordset(strSQL)



ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ConsultaOPImprimir = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function ListadoChequesLiberar(ByVal pstrBanco As String) As ADODB.Recordset

Descripción: Funcion para obtener una lista de los cheques
por liberar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoChequesLiberar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso
    Dim strCveBanco As String
    Dim objCatalogo As Catalogos

    On Error GoTo ErrorHandler

    'Obtenemos la Clave del Banco
    Set objCatalogo = New Catalogos
    strCveBanco = objCatalogo.strObtieneCve(GSTR_TBL_BANCOS_FLD_ID, pstrBanco,
            GSTR_TBL_BANCOS, _
```

```
        GSTR_TBL_BANCOS_FLD_DESCRIPCION, False)

Set objBD = New SIFIBD.BDAcceso


strSQL = "SELECT " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_NUMERO
strSQL = strSQL & ", Sum(" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPORTE & ") As " &
        GSTR_TBL_REFERENCIAS_FLD_IMPORTE & vbCr
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_REF_CON
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE &
        vbCr
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO
strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & ".IDConsolida"

strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr

strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA
        & vbCr

strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
        "NOMBRE_OFICINA"
strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
        "NOMBRE_AREA"
strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

strSQL = strSQL + " FROM "
strSQL = strSQL + GSTR_TBL_REFERENCIAS
         'Orden_Pago
strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
strSQL = strSQL + " ON " + GSTR_TBL_REFERENCIAS + "." + GSTR_TBL_REFERENCIAS_FLD_IDOP
        & " = "
strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr

'Rubro
strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_RUBRO & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
strSQL = strSQL & " = " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & vbCr

'Oficina
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_OFICINA & vbCr
strSQL = strSQL & " ON " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_ID & " = "
strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID & vbCr

'Area
strSQL = strSQL & " INNER JOIN " & GSTR_TBL_AREA & vbCr
```

```
    strSQL = strSQL & " ON " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_ID & " = "
    strSQL = strSQL & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID & vbCr

    'Centro Costos
    strSQL = strSQL + " INNER JOIN " + GSTR_TBL_CENTRO_COSTO & vbCr
    strSQL = strSQL + " ON " + GSTR_TBL_CENTRO_COSTO + "." + GSTR_TBL_OFICINA_FLD_ID + " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_OFICINA_FLD_ID + " And " & vbCr
    strSQL = strSQL + GSTR_TBL_CENTRO_COSTO + "." + GSTR_TBL_AREA_FLD_ID + " = " +
        GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_AREA_FLD_ID + " AND "
    strSQL = strSQL + GSTR_TBL_CENTRO_COSTO + "." +
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO + " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO
        & vbCr


    strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & " = 1"   'Cheque
    strSQL = strSQL & " AND (" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_LOTE & " IS NULL "
    strSQL = strSQL & " OR " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_LOTE & " = 0 )" & vbCr
    strSQL = strSQL & " AND (" & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & " = 1"
    strSQL = strSQL & " OR " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & "= 0)" & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_RUBRO_FLD_ID & " IN (

Select IDRubro From Rubro " & vbCr
    strSQL = strSQL & " where IDConcepto_Rubro = 1502 and IDBanco = " & Val(strCveBanco) & ")" & vbCr
    strSQL = strSQL & " AND " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & " = 0
        " & vbCr
    strSQL = strSQL & " AND (" & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & " = 10 "  'Autorizados, no rechazadas, y
Reimpresas
    strSQL = strSQL & " OR " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS & "= 7)" & vbCr



    strSQL = strSQL + " Group By "
    strSQL = strSQL & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_NUMERO
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_REF_CON
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IMPRESO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & ".IDConsolida" & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
```

```
    strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA
          & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    Set ListadoChequesLiberar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else
          Set ListadoChequesLiberar = Nothing

          mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing


End Function
```

Public Function ListadoTransferenciasEnviar() As ADODB.Recordset

Descripción: Funcion para obtener una lista de los transferencias por liberar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoTransferenciasEnviar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso


    strSQL = "SELECT " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON
    strSQL = strSQL & ", Sum(" & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & ") As " &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCOS_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & ".CTA_BCO_DES"

          strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_NOMBRE & " AS NOMBRE_PRESTADOR " & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
```

```
            strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
            GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

    strSQL = strSQL & ", IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_NOMBRE & ",' ')" & vbCr
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ')"
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ') AS " & GSTR_TBL_PRESTADOR & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_CUENTAS_BANCARIAS_FLD_CUENTA_BANCARIA
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
"NOMBRE_OFICINA"
    strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
"NOMBRE_AREA"
    strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

    strSQL = strSQL & "," & GSTR_TBL_BANCOS & ".IDBMR" & vbCr

    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_CVEBANCO
    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_DESCRIPCION
    strSQL = strSQL & ", " & GSTR_TBL_PLAZA & ".CveBancoPlazaBMR"
    strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL & "." & GSTR_TBL_SUCURSAL_FLD_CVEBANCO &
vbCr

    strSQL = strSQL & " FROM " & GSTR_TBL_REF_TRAN_AUT & vbCr

    'Orden_Pago
    strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
    strSQL = strSQL + " ON " + GSTR_TBL_REF_TRAN_AUT + "." +
GSTR_TBL_REF_TRAN_AUT_FLD_IDOP & " = "
    strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr


    Set ListadoTransferenciasEnviar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ListadoTransferenciasEnviar = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

```
Public Function ListadoConciliacion(ByVal pdtmFechaIni As Date, ByVal pdtmFechaFin As Date, ByVal
        plngIDRubro As Long) As ADODB.Recordset

'Descripción: Funcion para obtener una lista de las OP a conciliar

   Const STR_FUNCION As String = MSTR_MODULO & ".ListadoConciliacion"

   Dim strSQL     As String
   Dim objBD      As SIFIBD.BDAcceso

   On Error GoTo ErrorHandler

   Set objBD = New SIFIBD.BDAcceso


   strSQL = "SELECT " & GSTR_TBL_REFERENCIAS_FLD_ID
        strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT & ", " &
        GSTR_TBL_REFERENCIAS_FLD_DESCRIPCION
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_IDOP
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_NUMERO
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_IMPORTE
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_ESTATUS
   strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS_FLD_FECHA_CON


   strSQL = strSQL & " FROM " & GSTR_TBL_REFERENCIAS & vbCr

   strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_ESTATUS & " <> 'C'"
        strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & " = " &
        plngIDRubro & vbCr
   strSQL = strSQL & " AND " & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT & vbCr
   strSQL = strSQL & " BETWEEN '" & Format(pdtmFechaIni, "yyyy-mm-dd") & " 00:00:00'"
   strSQL = strSQL & " AND '" & Format(pdtmFechaFin, "yyyy-mm-dd") & " 23:59:59'" & vbCr

   strSQL = strSQL & "Order by " & GSTR_TBL_REFERENCIAS_FLD_FECHA_AUT
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_NUMERO
   strSQL = strSQL & "," & GSTR_TBL_REFERENCIAS_FLD_IDOP

   Set ListadoConciliacion = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set ListadoConciliacion = Nothing

         mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select

   Set objBD = Nothing

End Function


Public Function ListadoConciliacionBancos(ByVal pstrCtaBanca As String) As ADODB.Recordset
'Public Function ListadoConciliacionBancos(ByVal pdtmFechaIni As Date, ByVal pdtmFechaFin As Date,
ByVal pstrCtaBanca As String) As ADODB.Recordset
```

'Descripción: Funcion para obtener una lista de las OP a conciliar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoConciliacionBancos"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso


    strSQL = "SELECT * FROM Registro_Conciliacion_Bancos"

    strSQL = strSQL & " WHERE Estatus_Conciliacion <> 'C' OR Estatus_Conciliacion IS NULL AND

    Set ListadoConciliacionBancos = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            Set ListadoConciliacionBancos = Nothing

            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function ListadoLoteRegenerar(ByVal plngLote As Long) As ADODB.Recordset

Descripción: Funcion para obtener una lista de los Transferencias que pertenencen
' al Lote por Regenerar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoLoteRegenerar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    strSQL = "SELECT " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_ORDEN_PAGO_FLD_ID
            strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
            strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
            GSTR_TBL_ORDEN_PAGO_FLD_ULTIMO_STATUS
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_OBSERVACIONES
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_REFERENCIAS_FLD_IMPORTE
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
    GSTR_TBL_ORDEN_PAGO_FLD_TIPO_OP & vbCr
```

```
  strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
  strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
        GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
 strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
 GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr

 strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_NUMERO
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_TIPO_PAGO & vbCr
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_IMPORTE
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." &
 GSTR_TBL_REFERENCIAS_FLD_REF_CON
  strSQL = strSQL & ", " & GSTR_TBL_REFERENCIAS & "." & GSTR_TBL_REFERENCIAS_FLD_LOTE &
 vbCr

  strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID_BANCO
        strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." &
        GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & vbCr

        strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & "
        AS " & "NOMBRE_OFICINA"
  strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
  "NOMBRE_AREA"
  strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
  GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

  strSQL = strSQL + " FROM "
  strSQL = strSQL + GSTR_TBL_REFERENCIAS
  'Orden_Pago
  strSQL = strSQL + " INNER JOIN " + GSTR_TBL_ORDEN_PAGO & vbCr
        strSQL = strSQL + " ON " + GSTR_TBL_REFERENCIAS + "." +
        GSTR_TBL_REFERENCIAS_FLD_IDOP & " = "
  strSQL = strSQL + GSTR_TBL_ORDEN_PAGO + "." + GSTR_TBL_ORDEN_PAGO_FLD_ID & vbCr

  'Rubro
  strSQL = strSQL & " LEFT JOIN " & GSTR_TBL_RUBRO & vbCr
  strSQL = strSQL & " ON " & GSTR_TBL_RUBRO & "." & GSTR_TBL_RUBRO_FLD_ID
        strSQL = strSQL & " = " & GSTR_TBL_REFERENCIAS & "." &
        GSTR_TBL_REFERENCIAS_FLD_IDCUENTABANC & vbCr


  Set ListadoLoteRegenerar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
  Select Case Err.Number
     Case 0
        'do nothing
     Case Else
        Set ListadoLoteRegenerar = Nothing

        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
  End Select

  Set objBD = Nothing
```

End Function

Public Function ListadoLoteTransfRegenerar(ByVal plngLote As Long) As ADODB.Recordset

Descripción: Funcion para obtener una lista de las Transferencias que pertenencen
' al Lote por Regenerar

```
    Const STR_FUNCION As String = MSTR_MODULO & ".ListadoLoteTransfRegenerar"

    Dim strSQL      As String
    Dim objBD       As SIFIBD.BDAcceso

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

          strSQL = "SELECT " & GSTR_TBL_REF_TRAN_AUT & "." &
          GSTR_TBL_REF_TRAN_AUT_FLD_REF_CON
    strSQL = strSQL & ", Sum(" & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & ") As " &
    GSTR_TBL_REF_TRAN_AUT_FLD_IMPORTE & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_IDCUENTABANCARIA 'Cta Origen
    strSQL = strSQL & ", " & GSTR_TBL_REF_TRAN_AUT & "." &
    GSTR_TBL_REF_TRAN_AUT_FLD_ID_SUCURSAL      'Sucursal Origen

    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_MONEDA_FLD_ID & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_OFICINA_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_AREA_FLD_ID
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_CENTRO_COSTO_FLD_ID_DEPTO & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_BANCOS_FLD_ID
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." & GSTR_TBL_PLAZA_FLD_ID_PLAZA
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_SUCURSAL_FLD_ID_SUCURSAL & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & ".CTA_BCO_DES"
          strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO & "." &
          GSTR_TBL_ORDEN_PAGO_FLD_OTRO_BENEFICIARIO

          strSQL = strSQL & ", " & GSTR_TBL_RUBRO & "." &
          GSTR_TBL_RUBRO_FLD_CUENTA_BANCARIA & " AS CUENTA_ORIGEN "

    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_NOMBRE &
" AS NOMBRE_PRESTADOR " & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." & GSTR_TBL_PRESTADOR_FLD_PATERNO
          strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_MATERNO & vbCr

          strSQL = strSQL & ", IsNull(" & GSTR_TBL_PRESTADOR & "." &
          GSTR_TBL_PRESTADOR_FLD_NOMBRE & ",' ')" & vbCr
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_PATERNO & ",' ')"
    strSQL = strSQL & " + ' ' + IsNull(" & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_MATERNO & ",' ') AS " & GSTR_TBL_PRESTADOR & vbCr
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_CUENTAS_BANCARIAS_FLD_CUENTA_BANCARIA
    strSQL = strSQL & ", " & GSTR_TBL_PRESTADOR & "." &
    GSTR_TBL_PRESTADOR_FLD_TIPO_CUENTA & vbCr
```

```
   strSQL = strSQL & ", " & GSTR_TBL_MONEDA & "." & GSTR_TBL_MONEDA_FLD_NOMBRE & vbCr

   strSQL = strSQL & ", " & GSTR_TBL_OFICINA & "." & GSTR_TBL_OFICINA_FLD_NOMBRE & " AS " &
"NOMBRE_OFICINA"
   strSQL = strSQL & ", " & GSTR_TBL_AREA & "." & GSTR_TBL_AREA_FLD_NOMBRE & " AS " &
   "NOMBRE_AREA"
   strSQL = strSQL & ", " & GSTR_TBL_CENTRO_COSTO & "." &
   GSTR_TBL_CENTRO_COSTO_FLD_NOMBRE_DEPARTAMENTO & vbCr

   strSQL = strSQL & "," & GSTR_TBL_BANCOS & ".IDBMR" & vbCr

   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_CVEBANCO
   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & "." & GSTR_TBL_PLAZA_FLD_DESCRIPCION
   strSQL = strSQL & ", " & GSTR_TBL_PLAZA & ".CveBancoPlazaBMR"
         strSQL = strSQL & ", " & GSTR_TBL_SUCURSAL & "." &
         GSTR_TBL_SUCURSAL_FLD_CVEBANCO & vbCr

   strSQL = strSQL & " FROM " & GSTR_TBL_REF_TRAN_AUT & vbCr



   Set ListadoLoteTransfRegenerar = objBD.rsdDevuelveRecordset(strSQL)

ErrorHandler:
   Select Case Err.Number
      Case 0
         'do nothing
      Case Else
         Set ListadoLoteTransfRegenerar = Nothing

         mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
   End Select

   Set objBD = Nothing

End Function
```

Public Function blnEliminaOrden(ByVal plngOP As Long, ByVal plngEjercicio As Long, ByVal plngEmpleado As Long) As Boolean
Descripción: Funcion para ejecutar Borrar una cuenta asociado a una OP

```
   Const STR_FUNCION As String = MSTR_MODULO & ".blnEliminaOrden"

   Const LNG_STATUS_ELIMINAR As Long = 9

   Dim strSQL      As String
   Dim objBD       As SIFIBD.BDAcceso
   Dim rsDatos     As ADODB.Recordset
   Dim strTransaccion As String
   Dim blnResultado As Boolean
   Dim strCondicion As String

On Error GoTo ErrorHandler

         strCondicion = " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_EJERCICIO & " = " & plngEjercicio
         & vbCr
   strCondicion = strCondicion & " AND " & GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOP
```

```
    Set objBD = New SIFIBD.BDAcceso

    strTransaccion = objBD.strBeginTrans

    If Trim$(strTransaccion) <> vbNullString Then

        'Elimino los impuestos
        strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO_CUENTA_IMPUESTO & vbCr
        strSQL = strSQL & strCondicion

        blnResultado = objBD.blnEjecutaQuery(strSQL)

        'Elimino los movimientos contables
        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_MOVTOS_CONTABLES & vbCr
            strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & " = " & plngOP

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If


        'Elimino las cuentas
        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO_CUENTA & vbCr
            strSQL = strSQL & strCondicion

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If



        If blnResultado Then
            strSQL = "DELETE " & GSTR_TBL_ORDEN_PAGO & vbCr
            strSQL = strSQL & strCondicion

            blnResultado = objBD.blnEjecutaQuery(strSQL)

        End If

        If blnResultado Then

            If blnActualizaStatus(plngOP, plngEjercicio, LNG_STATUS_ELIMINAR, plngEmpleado, True) Then
                objBD.blnCommitTrans strTransaccion
                blnEliminaOrden = True
            Else
                objBD.blnRollBack strTransaccion
                blnEliminaOrden = False
            End If

        End If

    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
```

```
        Case Else
            objBD.blnRollBack strTransaccion

            blnEliminaOrden = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION


    End Select

    Set objBD = Nothing

End Function
```

Public Function blnNuevaOrdenConceptoRubro(ByVal plngOP As Long, ByVal plngConsecutivo As Long, ByVal plngRubro As Long, ByVal pdatFechaSolicitud As Date, ByVal plngMvto As Long, ByVal plngMoneda As Long, ByVal pdblImporte As Double) As Boolean

Descripción: Funcion para Guardar los rubros de cada una de las partidas
de la OP

```
Const STR_FUNCION As String = MSTR_MODULO & ".blnNuevaOrdenConceptoRubro"

Dim strSQL          As String
Dim objBD           As SIFIBD.BDAcceso
Dim lngIDMovimiento As Long

On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    blnNuevaOrdenConceptoRubro = False

    'Valor del IDMovimiento
        lngIDMovimiento = objBD.lngGetNewId(GSTR_TBL_MOVTOS_CONTABLES,
        GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO)


    strSQL = "Insert Into " & GSTR_TBL_MOVTOS_CONTABLES
    strSQL = strSQL & " (" & GSTR_TBL_MOVTOS_CONTABLES_FLD_ID_MOVIMIENTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_CONSECUTIVO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_FECHA
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA
    strSQL = strSQL & ", " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE & ")"

    strSQL = strSQL & " Values (" & lngIDMovimiento
    strSQL = strSQL & ", " & plngOP
    strSQL = strSQL & ", " & plngConsecutivo
    strSQL = strSQL & ", " & plngRubro
    strSQL = strSQL & ", '" & Format(pdatFechaSolicitud, "yyyy-mm-dd") & "'"
    strSQL = strSQL & ", " & plngMvto
    strSQL = strSQL & ", " & plngMoneda
    strSQL = strSQL & ", " & pdblImporte & ")"

    blnNuevaOrdenConceptoRubro = objBD.blnEjecutaQuery(strSQL)
```

```
ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnNuevaOrdenConceptoRubro = False
            mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

Public Function blnActualizaCaratulaMov(ByVal plngOrden As Long, ByVal plngCaratula As Long, ByVal plngAsiento As Long, ByVal pblnCancelacion As Boolean) As Boolean

```
' Esta función actualiza el Asiento y la Carátula en las tablas de Movimientos y Referencias
' Si se está cancelando entonces no se actualiza Referencia y se ingresan nuevos registros en
' Movimientos

    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaCaratulaMov"
    Const LNG_VALOR_SISE As Long = 1

    Dim strSQL          As String
    Dim objBD           As SIFIBD.BDAcceso
    Dim strTransaccion  As String
    Dim strSQLMov       As String
    Dim rdsMovs         As ADODB.Recordset
    Dim intNaturaleza   As Integer

On Error GoTo errHandler

    Set objBD = New SIFIBD.BDAcceso

    If Not pblnCancelacion Then
        strSQLMov = "UPDATE " & GSTR_TBL_MOVTOS_CONTABLES
            strSQLMov = strSQLMov & " SET " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO & "=" &
            plngAsiento & ", " & vbCr
        strSQLMov = strSQLMov & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & "=" &
        plngCaratula & vbCr
        strSQLMov = strSQLMov & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & "=" &
        plngOrden

        strSQL = "UPDATE " & GSTR_TBL_REFERENCIAS & vbCr
        strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" & plngAsiento & vbCr
        strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula & vbCr
        strSQL = strSQL & " WHERE " & GSTR_TBL_REFERENCIAS_FLD_IDOP & "=" & plngOrden

    Else
        strSQL = "SELECT * FROM " & GSTR_TBL_MOVTOS_CONTABLES
        strSQL = strSQL & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP & "=" & plngOrden

        Set rdsMovs = objBD.rsdDevuelveRecordset(strSQL)
        If Not rdsMovs Is Nothing Then
            If Not rdsMovs.BOF And Not rdsMovs.EOF Then
                With rdsMovs
                    .MoveFirst
                    Do While Not .EOF
```

```
                    intNaturaleza = .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_MOVTO)
                    If intNaturaleza = 1 Then
                        intNaturaleza = 2
                    Else
                        intNaturaleza = 1
                    End If
            blnNuevaOrdenConceptoRubro plngOrden,
            .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_CONSECUTIVO), _
                        .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_IDRUBRO),
            Format(GetDateFromServer, "YYYY-MM-DD"), _
                        intNaturaleza, .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_MONEDA), _
                        .Fields(GSTR_TBL_MOVTOS_CONTABLES_FLD_IMPORTE)
                    .MoveNext
                Loop
                'Actualizamos a los nuevos movimientos la Caratula y el Asiento
                strSQLMov = "UPDATE " & GSTR_TBL_MOVTOS_CONTABLES
            strSQLMov = strSQLMov & " SET " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO & "=" &
            plngAsiento & ", " & vbCr
                strSQLMov = strSQLMov & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA & "=" &
                plngCaratula & vbCr
                strSQLMov = strSQLMov & " WHERE " & GSTR_TBL_MOVTOS_CONTABLES_FLD_IDOP &
                "=" & plngOrden & vbCr
                strSQLMov = strSQLMov & " AND " & GSTR_TBL_MOVTOS_CONTABLES_FLD_ASIENTO &
                " Is Null "
                strSQLMov = strSQLMov & " AND " & GSTR_TBL_MOVTOS_CONTABLES_FLD_CARATULA
                & " Is Null "

                strSQL = "UPDATE " & GSTR_TBL_ORDEN_PAGO & vbCr
                    strSQL = strSQL & " SET " & GSTR_TBL_ORDEN_PAGO_FLD_ASIENTO & "=" &
                    plngAsiento & vbCr
                strSQL = strSQL & ", " & GSTR_TBL_ORDEN_PAGO_FLD_CARATULA & "=" & plngCaratula
                & vbCr
                strSQL = strSQL & " WHERE " & GSTR_TBL_ORDEN_PAGO_FLD_ID & "=" & plngOrden

                'blnActualizaCaratulaMov = objBD.blnEjecutaQuery(strSQLMov)
            End With
        End If
    End If
End If

strTransaccion = objBD.strBeginTrans

If objBD.blnEjecutaQuery(strSQL) Then
    If objBD.blnEjecutaQuery(strSQLMov) Then
        blnActualizaCaratulaMov = objBD.blnCommitTrans(strTransaccion)
    Else
        Call objBD.blnRollBack(strTransaccion)
    End If
Else
    Call objBD.blnRollBack(strTransaccion)
End If

errHandler:
    Select Case Err.Number

    Case 0
        'Do nothing
    Case Else
        mobjError.RegistraError Err.Number, Err.Description, STR_FUNCION
        blnActualizaCaratulaMov = False
```

```
            Call objBD.blnRollBack(strTransaccion)
        End Select

        'Libero la memoria
        Set objBD = Nothing

    End Function
```

CLASE SEGURIDAD

Descripción: Clase utilizada para la validación de Claves de acceso de Usuarios

Public Function blnCambiaPassword(ByVal plngID As Long, ByVal pstrPassword As String, ByVal pstrViejo As String) As Boolean
Descripción: Función utilizada para modificar la clave de acceso del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".blnCambiaPassword"

    Dim strSql As String
    Dim objBD As SIFIBD.BDAcceso
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strPassword As String

On Error GoTo errHandler

        Set objEncripta = New xmsEncriptador.Encriptador
        'encripto el nuevo password
        strPassword = objEncripta.EncriptaString(pstrPassword)

        'Formo el query
        strSql = "UPDATE " & GSTR_TBL_EMPLEADO & vbCr
        strSql = strSql & " SET " & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" & Trim$(strPassword) &
"'" & vbCr
        strSql = strSql & " ," & GSTR_TBL_EMPLEADO_FLD_STATUS_ACCESO & "=1" & vbCr
        strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_ID & "=" & plngID

        Set objBD = New SIFIBD.BDAcceso

        'Ejecuto el query
        blnCambiaPassword = objBD.blnEjecutaQuery(strSql)

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            blnCambiaPassword = False
    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set objEncripta = Nothing

End Function
```

Public Property Let Empleado(ByVal plngEmpleado As Long)

```
    mlngEmpleado = plngEmpleado
End Property
```

Public Function lngNuevoRol(ByVal pstrNombre As String) As Long
Descripción: Función para agregar una nueva clave de Rol al usuario

```
Dim objBD          As SIFIBD.BDAcceso
```

```
Dim strSql          As String
Dim lngIdRol        As Long

On Error GoTo ErrorHandler

    If Trim$(pstrNombre) <> vbNullString Then

        lngIdRol = objBD.lngGetNewId("Rol", "IdRol")

        If lngIdRol > 0 Then

            Set objBD = New SIFIBD.BDAcceso

            strSql = "Insert Into Rol (Nombre, IdRol) "
            strSql = strSql & "Values ('" & pstrNombre & "', " & _
                                    lngIdRol & ")"


            Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ALTA)


            objBD.blnEjecutaQuery strSql

            Set objBD = Nothing

            lngNuevoRol = lngIdRol
        Else
'            mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"
        End If
    Else
'       mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"
    End If

ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

            Set objBD = Nothing

            lngNuevoRol = -1

'            mobjErr.RegistraError Err.Number, vbCrLf & Err.Description, "SIFISeguridad.lngNuevoRol"

    End Select
End Function
```

Public Function blnAsignaRolObjeto(ByVal plngObjeto As Long, ByVal plngRol As Long) As Boolean

Descripción: Función utilizada para dar permisos a un objeto de la pantalla a un rol del usuario

```
    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo ErrorHandler

    blnAsignaRolObjeto = False
```

```
    Set objBD = New SIFIBD.BDAcceso


'    strSql = "UPDATE " &  Objeto set IdRol=" & plngRol
'    strSql = strSql & "Where idObjeto =" & plngObjeto

    objBD.blnEjecutaQuery strSql

    Set objBD = Nothing

    blnAsignaRolObjeto = True


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else

            Set objBD = Nothing

            blnAsignaRolObjeto = False

            mobjErr.RegistraError Err.Number, Err.Description, "SIFISeguridad.blnAsignaRolObjeto"

    End Select

End Function
```

Public Function lngValidaExistenciaEmpleado(ByVal pstrLogin As String, ByVal pstrPassword As String) As Long

Descripción: Función para validar los datos del Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".lngValidaExistenciaEmpleado"
    Const LNG_NO_EXISTE_EMPLEADO As Long = -1
    Const LNG_EMPLEADO_INACTIVO As Long = -2
    Const LNG_PASSWORD_INCORRECTO As Long = -3

    Dim strSql As String

    Dim objBD As SIFIBD.BDAcceso
    Dim rsdEmpleado As ADODB.Recordset
    Dim objEncripta As xmsEncriptador.Encriptador
    Dim strPassword As String

On Error GoTo errHandler

    strSql = "SELECT * FROM " & GSTR_TBL_EMPLEADO & vbCr
    strSql = strSql & " WHERE " & GSTR_TBL_EMPLEADO_FLD_LOGIN & "='" & LCase$(pstrLogin) & "'" &
vbCr
    'strSQL = strSQL & " AND " & GSTR_TBL_EMPLEADO_FLD_PASSWORD & "='" &
LCase$(pstrPassword) & "'" & vbCr

    Set objBD = New SIFIBD.BDAcceso


    Set rsdEmpleado = objBD.rsdDevuelveRecordset(strSql)
```

```
    If Not (rsdEmpleado.BOF And rsdEmpleado.EOF) Then
        Set objEncripta = New xmsEncriptador.Encriptador
        strPassword =
objEncripta.DesencriptaString(rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_PASSWORD))

        'Verifico que el password sea correcto
        If UCase$(strPassword) = UCase$(pstrPassword) Then
            lngValidaExistenciaEmpleado = rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_ID)
        Else
            lngValidaExistenciaEmpleado = LNG_PASSWORD_INCORRECTO
        End If

        'Verifico si el empelado está activo o no para el sistema
        If Not rsdEmpleado(GSTR_TBL_EMPLEADO_FLD_ACTIVO) Then

            lngValidaExistenciaEmpleado = LNG_EMPLEADO_INACTIVO
        End If


    Else
        'El empleado no existe
        lngValidaExistenciaEmpleado = LNG_NO_EXISTE_EMPLEADO
    End If

errHandler:
    Select Case Err.Number

        Case 0
            'Do nothing
        Case Else
            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
            lngValidaExistenciaEmpleado = LNG_NO_EXISTE_EMPLEADO
    End Select

    'Libero la memoria
    Set objBD = Nothing
    Set rsdEmpleado = Nothing
    Set objEncripta = Nothing

End Function
```

Public Function rsdObtenPermisosUsuario(plngUsuario As Long) As ADODB.Recordset

Descripción: Función para obtener los permisos de un Usuario

```
    Const STR_FUNCION As String = MSTR_MODULO & ".rsdObtenPermisosUsuario"

    Dim objBD As SIFIBD.BDAcceso
    Dim strSql As String

On Error GoTo ErrorHandler

    strSql = "SELECT " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_NOMBRE & " AS " &
            GSTR_TBL_OBJETO & ", " & vbCr
    strSql = strSql & GSTR_TBL_PANTALLA & "." & GSTR_TBL_PANTALLA_FLD_NOMBRE & " AS " &
            GSTR_TBL_PANTALLA & vbCr
    strSql = strSql & " FROM " & GSTR_TBL_ROL_EMPLEADO & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_ROL_OBJETO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_ROL_FLD_ID
```

```
    strSql = strSql & " = " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_ROL_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_OBJETO & vbCr
    strSql = strSql & " ON " & GSTR_TBL_ROL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_OBJETO & "." & GSTR_TBL_OBJETO_FLD_ID & vbCr
    strSql = strSql & " INNER JOIN " & GSTR_TBL_PANTALLA & vbCr
    strSql = strSql & " ON " & GSTR_TBL_OBJETO & "." & GSTR_TBL_PANTALLA_FLD_ID
    strSql = strSql & " = " & GSTR_TBL_PANTALLA & "." & GSTR_TBL_PANTALLA_FLD_ID
    strSql = strSql & " WHERE " & GSTR_TBL_ROL_EMPLEADO & "." & GSTR_TBL_EMPLEADO_FLD_ID
          & "=" & plngUsuario
    strSql = strSql & " AND " & GSTR_TBL_ROL_EMPLEADO & "." &
          GSTR_TBL_ROL_EMPLEADO_ULTIMO_ROL & "=1"

    Set objBD = New SIFIBD.BDAcceso
    Set rsdObtenPermisosUsuario = objBD.rsdDevuelveRecordset(strSql)

ErrorHandler:
    Select Case Err.Number
       Case 0
          'do nothing
       Case Else



          Set rsdObtenPermisosUsuario = Nothing

          mobjErr.RegistraError Err.Number, Err.Description, "SIFISeguridad.blnAsignaRolUsuario"

    End Select

    'Libero la memoria
    Set objBD = Nothing

End Function


'
Public Function blnActualizaRol(ByVal pstrDescripcion As String, ByVal pdblMontoMaximo As Double,
          ByVal pblnInserta As Boolean, Optional ByVal plngIDRol As Long) As Boolean
'Descripcion:       Funcion para actualizar un rol

    Const STR_FUNCION As String = MSTR_MODULO & ".blnActualizaRol"

          Dim strSql            As String
          Dim objBD             As SIFIBD.BDAcceso
          Dim lngIdRol     As Long

    On Error GoTo ErrorHandler

    Set objBD = New SIFIBD.BDAcceso

    blnActualizaRol = False

    If pblnInserta = True Then
       'verifica que no exista el nombre del Rol

       If objBD.blnExisteDatoenTabla(GSTR_TBL_ROL_FLD_DESCRIPCION, GSTR_TBL_ROL,
                pstrDescripcion) = False Then

          lngIdRol = objBD.lngGetNewId(GSTR_TBL_ROL, GSTR_TBL_ROL_FLD_ID)
```

```
            strSql = "Insert into " & GSTR_TBL_ROL & " (" & GSTR_TBL_ROL_FLD_ID & ", " & _
                                                GSTR_TBL_ROL_FLD_CANTIDAD & ", " & _
                                                GSTR_TBL_ROL_FLD_DESCRIPCION & ") values(" & _
                                                lngIdRol & ", " & _
                                                pdblMontoMaximo & ", '" & _
                                                pstrDescripcion & "')"

            Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ALTA)

        End If
    Else

        strSql = "delete from " & GSTR_TBL_ROL & " where " & GSTR_TBL_ROL_FLD_ID & "=" & plngIDRol

        Call RegistrarBitacora(GSTR_TBL_ROL, mlngEmpleado, MSTR_MOVIMIENTO_ELIMINAR)

    End If

        'verifica que exista una sentencia a ejecutar
    If Len(strSql) > 0 Then
        strSql = UCase(strSql)

        blnActualizaRol = objBD.blnEjecutaQuery(strSql)

    End If


ErrorHandler:
    Select Case Err.Number
        Case 0
            'do nothing
        Case Else
            blnActualizaRol = False

            mobjErr.RegistraError Err.Number, Err.Description, STR_FUNCION
    End Select

    Set objBD = Nothing

End Function
```

REFERENCIAS

1. Object Management Group
http://www.uml.org/what_is_uml.htm
página consultada en Octubre del 2004.

2. Booch, G., Rumbaugh, J., Jacobson, I.
The Unified Modeling Process, Addison Wesley.
1998.