UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA

CONCILIADOR BANCARIO AUTOMATIZADO
"UNA ALTERNATIVA PARA EL DESARROLLO EMPRESARIAL"

TESIS

PARA OBTENER EL TÍTULO DE INGENIERO EN COMPUTACIÓN

PRESENTAN:

DÁVILA ISLAS SAHAD YAMIL
MARTÍNEZ GÁLVEZ BERNARDO
TERÁN PAREDES VÍCTOR ÁNGEL

DIRECTORA DE TESIS:

M. EN I. NORMA ELVA CHÁVEZ RODRÍGUEZ

LUGAR DE PUBLICACIÓN: CIUDAD UNIVERSITARIA

FECHA DE PUBLICACIÓN: 05 DE NOVIEMBRE DE 2014

AGRADECIMIENTOS

En primer lugar le agradezco a dios por acompañarme siempre en mi camino y darme la salud suficiente para realizar mis actividades día a día.

Agradezco a la Universidad Nacional Autónoma de México por haberme formado como persona y como profesionista lo cual para mi es una gran satisfacción, ya que a lo largo de mi carrera el panorama no fue fácil, hubo obstáculos pero nunca perdí la esperanza de que este día llegaría.

Agradezco a la Facultad de Ingeniería por haberme dado las herramientas necesarias como persona y profesionista para hacer de nuestro México un mejor país, transformando nuestro entorno y haciendo que las personas tengan una mejor conciencia del impacto que la ingeniería tiene sobre el desarrollo del país.

También quiero agradecer infinitamente a mis padres y hermanos, los cuales han contribuido mucho a mi formación, a mi madre Lourdes que siempre me ha inculcado que todo se puede lograr a base de esfuerzo y dedicación, a mi padre Bernardino por sus buenos consejos que me han ayudado a ser una persona humilde, a mi hermano Hugo por apoyarme en cada momento brindándome consejos y apoyándome incondicionalmente, a mi hermana Diana que gracias a su compañía y consejos he logrado mucho.

Otro agradecimiento muy amplio se lo debo a mis tíos y primos los cuales me apoyaron bastante en este proceso, gracias tío Alejandro por brindarme ese apoyo incondicional, por orientarme para tomar mejores decisiones, a mi tía Elvira por ayudarme en el día a día, por tomar un papel de madre, del cual le estoy muy agradecido y por apoyarme desde que supo que había sido aceptado por la universidad. A ustedes dos los considero como mis segundos padres, aunque nunca se los exprese les agradezco infinitamente el apoyo y sé que ni con toda la riqueza del mundo terminaría de pagarles por toda su ayuda. A mi primo Ale por acompañarme en este tiempo, por brindarme ese espacio y compartir juntos buenas y malas experiencias, a mi prima Pauli por hacer más ligero el día con sus chistes y por brindarme ese apoyo al convivir con ella en un mismo hogar.

Agradezco a mis abuelitos, abuelitas, tíos, tías, primos, primas, sobrinos y sobrinas por creer en mí. A mis compañeros y amigos Yamil y Victor por compartir experiencias durante la carrera y colaborar para que este proyecto fuera posible, gracias por todo su apoyo.

A mi directora de tesis Norma Elva por todo el apoyo recibido desde que le presentamos la propuesta, al igual que el jurado compuesto por el profesor Adán Zepeda, Jorge Valeriano, Juan Jose Carreón y la profesora Luciralía Hernandez, muchas gracias por haber revisado mi tesis y aceptar ser parte de mi jurado.

Y por último agradezco a todos los profesores de México que día a día hacen un esfuerzo por educarnos y nos hacen ver que un mejor país es posible, gracias a todos por elegir esa vocación.

Bernardo Martínez Gálvez

Esta tesis se la dedico a mi familia por quienes he logrado ser lo que soy.

Para mis padres Sandra y Nabor por su apoyo, consejos, compresión y Amor (si, Amor con mayuscula). Me han dado todo lo que soy como persona, mis valores, mis principios, mi carácter, mi empeño, mi perseverancia, mi coraje y decisión para conseguir mis objetivos. Gracias por inculcarme una excelente educación en el transcurso de mi Vida, pero sobre todo por ser un excelente ejemplo de Vida a seguir.

A mis hermanos Giovanni y Edwin por estar siempre presentes. Por ser parte importante en mi Vida y representar un verdadero lazo de unidad familiar.

A mi abuela Irene y mi tía Silvia que aunque ya no se encuentren físicamente con nosotros, siempre estarán presentes en mi Corazón , por haber creído en mí hasta el último momento. Hoy mirando al cielo puedo decirles con alegría ¡Ya soy Ingeniera!

A Ricardo Rigel mi Sensei, por el gran apoyo que me brindó durante esta etapa de mi Vida. Gracias por estar siempre presente en la adversidad y en la alegría, por tenderme la mano cuando más lo necesité. Gracias por tu Amor.

A Bernardo y a Víctor por haber sido excelentes compañeros de tesis y amigos. Por haberme tenido la paciencia necesaria y por motivarme a seguir adelante en los momentos de desesperación. He aquí el resultado de nuestro gran esfuerzo.

A mis amigos Diana, Emanuel, Jesús, Angie, Merilin por confiar y creer en mí y haber hecho de mi etapa universitaria un trayecto de vivencias que nunca olvidaré.

A mis profesores Norma y Jaramillo, agradezco el apoyo brindado a lo largo de la carrera, por su tiempo, amistad y por los conocimientos que sabiamente supieron transmitirnos.

Sahad Yamil Dávila Islas

Agradezco a mi padre Ángel por los consejos que me ofrece y que me han ayudado a afrontar una gran cantidad de retos, estoy muy agradecido por el gran esfuerzo que ha realizado al proporcionarme todo lo que yo necesitaba para que pudiera estudiar y así tener una formación profesional.

Agradezco a mi madre Elfega por me ha brindado todo su amor, su comprensión, por enseñarme a no rendirme, a realizar las cosas con determinación, a ser una persona honesta y responsable pero sobre todo por estar siempre dispuesta a ayudarme.

A mi hermana Rubi por formar parte de mi vida, aunque casi no nos vemos, siempre he disfrutado la bonita forma en que nos tratamos y llevamos, haciendo que mi vida sea más elegre.

Agradezco a mis compañeros de tesis Bernardo y Yamil, porque ellos creyeron que yo era una persona en quien se podía confiar, también por el entusiasmo y el esfuerzo que pusieron a lo largo de todo el proyecto y que en estos momentos los tres empezamos a ver el resultado de nuestro trabajo.

Agradezco a la profesora Norma Elva por la paciencia que tuvo con nosotros, también por creer que el proyecto que presentamos valia la pena y por el tiempo que invirtio para asesorarnos en el desarrollo de la tesis.

Por ultimo quiero dedicar esta tesis a toda mi familia, primero que nada por haberme dado la oportunidad de cumplir con mi meta, pero por sobre todas las cosas por que estuvieron conmigo brindándome su apoyo incondicional en los momentos más difíciles de mi vida. Gracias al cariño que me brindaron he podido librar muchos obstáculos, permitiéndome seguir adelante y llegar hasta donde estoy en estos momentos.

Víctor Ángel Terán Paredes

Temario

Indice de Imagenes

1.0 Introducción

1.1 Antecedentes

En la actualidad existen varias aplicaciones para realizar conciliaciones bancarias, algunas de las cuales son básicas debido a que no cuentan con interfaz gráfica de usuario para tener una interacción más amena, únicamente hacen uso de funciones o herramientas ya existentes como, por ejemplo, libros de Excel que realizan todo el procesamiento de los datos, manipulándolos únicamente por un solo método (conciliaciones por monto), algunas concilian estos datos sólo si no existen cantidades repetidas, obligando a los usuarios a realizar el resto de las conciliaciones manualmente.

Otras aplicaciones sí incluyen manejo de datos por los métodos concepto-monto y monto e incluso con características adicionales como gráficas, procesamiento de conciliaciones de periodos anteriores con actuales, control de manejo durante la ejecución del proceso, etcétera. Sin embargo, para poder hacer las conciliaciones con estos programas se requiere realizar una serie de configuraciones para poder obtener los resultados esperados. Motivo por el cual, se vuelve difícil y confuso su uso, exigiendo a los usuarios llevar una capacitación o solicitar soporte técnico para poder usar la aplicación.

Muchas de estas aplicaciones requieren que la información que va a ser conciliada sea enviada por internet para poder ser procesada desde un servidor, lo cual puede generar desconfianza por parte de los usuarios, debido a que dicha información es confidencial; por ello, la seguridad que se requiere se vuelve un problema más a resolver. Además de representar un costo extra debido a que implica tener acceso a internet en las computadoras donde sea utilizada, incluyendo la renta o tarifa que se cobra por la aplicación que realiza las conciliaciones y que es proporcionado por un proveedor.

Durante el proceso de investigación se analizaron algunas versiones de prueba de aplicaciones que también realizan conciliaciones bancarias y que son desarrolladas por empresas especializadas. Uno de los aspectos relevantes de dichas aplicaciones es que para poder realizar las pruebas de su funcionamiento, en la gran mayoría de ellas, se necesita algún complemento para poder hacer una carga masiva de datos interactuando con archivos de Excel, el cual se adquiere por separado o sólo está disponible comprando la licencia.

Algunas de ellas no cuentan con suficiente información en la página principal referente a aspectos de procesamiento, como métodos que utiliza para procesar, interacción con Excel o algún otro complemento para hacer la carga de datos más rápida; otras versiones de prueba no pueden hacer las conciliaciones de manera automática, ya que en estas se realiza la carga de datos de manera manual, añadiendo dato por dato, lo cual demanda una mayor carga de trabajo para los usuarios. La manera en que funcionan es indicando a la aplicación qué datos son los que se quieren conciliar relacionando cada registro con otro registro y descartando los

que no se pueden vincular, para que al final sean presentados como el resultado de la conciliación; esto aumenta la posibilidad de realizar la conciliación con errores, debido a que el usuario podría equivocarse.

La mayoría de las versiones de evaluación, aparte de contar con el módulo de conciliaciones bancarias, traen implementadas otras funciones que permiten realizar inventarios de productos, tener registros del personal, llevar un mejor control de caja, control de clientes, proveedores, pedidos, presupuestos, órdenes de pago, facturación de ventas y de compras, entre otras.

Por lo general, las aplicaciones de evaluación no realizan el procesamiento de datos esperado puesto que son versiones incompletas. Así mismo hay empresas que no permiten descargar su versión de demostración completa solo si se proporciona una justificación y datos personales por lo que no suelen ser confiables o bien, dificultan su evaluación para conocer sus características.

1.2 Objetivos

El conciliador bancario automatizado procesará el libro auxiliar de bancos y el de estado de cuenta, considerando que los datos deben de apegarse a un formato específico para que su procesamiento sea el correcto, además de que los resultados serán mostrados en un formato predefinido con la intención de que sea fácilmente analizado por los usuarios.

La aplicación no necesitará conexión a internet para el procesamiento de los datos, lo que reduce su costo a largo plazo debido a que, una vez instalada la aplicación en la PC, el usuario podrá realizar las conciliaciones que requiera cuantas veces lo necesite, sin la necesidad de exponer su información por la red a un proveedor de servicios, asegurando así que el manejo de la información sólo esté disponible para los usuarios que realicen dicha actividad.

Ésta misma realizará el procesamiento de todos los datos incluidos en el libro auxiliar y el de estado de cuenta, junto con datos que no fueron conciliados en procesos anteriores, siempre y cuando el usuario junte los datos no conciliados con los actuales que va a procesar.

La aplicación procesará los datos por medio de tres métodos:

El **primer método** será aquel que concilie ambos libros por concepto (claves que permitirán identificar cheques) y monto al mismo tiempo, lo que implicará que los datos coincidentes en ambos campos quedarán descartados para el siguiente método.

El **segundo método** será por monto y fecha, lo cual representará que estos campos deben de coincidir en los dos libros, los datos que no coincidan serán procesados por el último método.

El **tercer método** procesará los datos únicamente por monto, en el cual deben de coincidir tanto en auxiliar como en el estado de cuenta.

Al final los datos que no sean procesados por cualquiera de estos tres métodos serán reportados en un libro de Excel como ***"no conciliados"*** y dependiendo del signo del monto (**positivo o negativo**) y del libro de donde proceden originalmente, estos datos serán presentados en las siguientes cuatro categorías:

- Cargos de la empresa no reconocidos por el banco
- Abonos de la empresa no reconocidos por el banco
- Cargos del banco no reconocidos por la empresa
- Abonos del banco no reconocidos por la empresa

Cargos de la empresa no reconocidos por el banco: Son los montos positivos que aparecen en el libro auxiliar de bancos pero no aparecen en el estado de cuenta.

Abonos de la empresa no reconocidos por el banco: Son los montos negativos que aparecen en el libro auxiliar de bancos pero no aparecen en el estado de cuenta.

Cargos del banco no reconocidos por la empresa: Son los montos negativos que aparecen en el estado de cuenta pero no aparecen en el libro auxiliar de bancos.

Abonos del banco no reconocidos por la empresa: Son los montos positivos que aparecen en el estado de cuenta pero no aparecen en el libro auxiliar de bancos.

El objetivo de procesar los datos por los tres métodos mencionados anteriormente será el de tener una mayor precisión y obtener los datos que no serán posibles de conciliar, lo cual brindará un mayor porcentaje de certeza en los datos presentados en el reporte final.

El procesamiento será realizado con interacción a una base de datos, libros de Excel, archivos scripts, código en lenguaje Java, con la finalidad de presentar al usuario final una interfaz interactiva que le permitirá realizar el procesamiento de manera intuitiva y en un menor tiempo en comparación al realizarlo manualmente.

El ahorro de tiempo será fundamental, debido a que los procesos internos se realizarán de manera automatizada para obtener una ejecución rápida pero sin pérdida de datos o modificación en los mismos. Por tal motivo la codificación será analizada parte por parte para hacer funciones que permitirán un procesamiento más veloz.

La interfaz del conciliador bancario automatizado será capaz de brindar facilidad de uso a los usuarios finales ya que no tendrán que realizar muchos pasos para poder obtener los resultados de la conciliación; lo que se pretende es que dichos resultados sean interpretados sin ambigüedad para que el usuario no gaste mucho tiempo en analizarlos.

Se pretende dar un seguimiento a la evaluación del producto por parte de los desarrolladores como del vendedor para ofrecer a los clientes un mejor servicio y si así lo fuera, poder implementar otros procesos importantes en la empresa en los que se invierte mucho tiempo el realizarlos manualmente, con la finalidad de poder ofrecer una suite más completa a largo plazo y poder beneficiar a las empresas que adquieran estas aplicaciones, ofreciéndoles, posteriormente, un buen soporte para retroalimentar los sistemas con base a las opiniones y a la experiencia adquirida por los usuarios que llegarán a utilizarla.

2.0 Marco teórico

En la actualidad es necesario procesar una gran cantidad de información, lo cual ha hecho que muchas empresas deban de implementar mecanismos para poder llevar a cabo sus actividades. Parte importante de dichas actividades son realizadas manualmente lo cual implica una mayor inversión tanto de tiempo como de recursos financieros para realizarlas; tal es el caso particular de las conciliaciones bancarias.

Es por ello que las empresas deciden hallar la manera de automatizar dichas actividades con la finalidad de reducir la inversión de recursos, con lo que posteriormente logran ser independientes desarrollando sus propias aplicaciones y satisfaciendo a la medida sus requerimientos. Algunas más, con el fin de lograr tales objetivos, optan por buscar algún proveedor de servicios o empresa desarrolladora de software quienes se encargarán de cumplir con los resultados esperados. En ambos casos, realizando las actividades que antes hacían, en un menor tiempo, contando además con el soporte necesario para que las operaciones se realicen correctamente.

Lamentablemente la mayor parte de las aplicaciones existentes en el mercado son caras y muchas de ellas no brindan información suficiente para que los usuarios finales queden convencidos de adquirirla, otras a pesar de ser gratuitas, simplemente no cumplen con el objetivo.

Las ventajas de realizar conciliaciones bancarias por medio de una aplicación es la rapidez con que se realizan, aunque también deben de ofrecer una sencilla interacción entre la interfaz gráfica y el usuario quien será capaz de interpretar fácilmente los resultados, así como una manipulación simple y práctica de los datos.

2.1 Conciliaciones bancarias

Conciliar consiste en comparar dos o más conjuntos de elementos.

De ahí que se derive el concepto de **“Conciliación Bancaria”** que consiste en el proceso mediante el cual existe una confrontación, a través de la cual se deben encontrar discrepancias que existan entre dos o más cuentas relacionadas entre sí y que al parecer son contrarias o arrojan saldos diferentes. Esta operación se realiza entre el estado de cuenta corriente que envía el banco (extracto bancario) y el libro auxiliar de bancos para:

- Establecer responsabilidades
- Corregir errores u omisiones

Cabe señalar que una conciliación bancaria forma parte de un grupo de procedimientos empleados por una organización para mantener el control sobre sus activos, los cuales también en conjunto se les conoce como **"sistema de control interno".**

Entre las posibles partidas que se consideran en una conciliación bancaria figuran:

- Cheques pendientes de pago.
- Depósitos en tránsito.
- Cargos por servicios bancarios.
- Cobros efectuados por el banco a favor del depositante (por ejemplo, el cobro de una letra por cobrar).
- Pagos efectuados por el banco en nombre del depositante (por ejemplo, la cancelación de una letra por pagar).
- Errores cometidos ya sea por el banco o por el depositante (por ejemplo, errores aritméticos o la omisión de un asiento).

Cheques pendientes de pago: son aquellos que ha girado el depositante pero que no han sido presentados al banco para su cancelación con la suficiente antelación para que aparezcan en el estado de cuenta. Los cheques pendientes de pago deben *deducirse* del saldo presentado por el banco.

Depósitos en tránsito: son aquellos que se han registrado en los libros del depositante pero que no fueron entregados al banco con la suficiente anticipación para que aparecieran en el estado de cuenta bancario.

Cargos por servicios bancarios: son aquellos que el banco carga en la cuenta del depositante por el "servicio" de la cuenta. Por ejemplo, si el saldo de una cuenta baja de cierto nivel mínimo establecido, muchos bancos cobran una remuneración por cada cheque girado. Si los cargos por servicios bancarios no han sido registrados en los libros del depositante, deben *deducirse* del saldo en libros.

2.1.1 Estado de cuenta

El estado de cuenta es aquel documento que el banco envía mensualmente al depositante; consiste en un informe que muestra los registros efectuados en su cuenta y el cual informa lo siguiente:

- El saldo que presenta la cuenta corriente al iniciarse el mes y al finalizar éste.
- Cargos y abonos durante el mismo.
- La causa por la cual se hicieron estos registros.

2.1.2 Libro auxiliar de bancos

El libro auxiliar de bancos es el libro contable en donde se registra de forma analítica y detallada, los valores e información registradas en los libros principales de la empresa. Existe la obligación de llevarlos ya que estos deben servir de soporte para conocer los movimientos (ingresos y egresos) de la empresa. Su número es ilimitado de acuerdo a las necesidades de cada ente económico, de acuerdo con su tamaño y el trabajo que se tenga que realizar, de manera que permitan el completo entendimiento de los libros obligatorios de contabilidad.

2.1.3 Pasos para elaborar una conciliación bancaria

El procedimiento general para elaborar la conciliación bancaria manualmente se realiza de la siguiente manera:

Primer paso
Se compara el libro auxiliar de bancos con el estado de cuenta para determinar si los movimientos que el depositante hizo durante el mes aparecen registrados en ambos. Las diferencias se anotan en una hoja de borrador.

Segundo paso
Se compara el libro auxiliar de bancos con el estado de cuenta para determinar si los cheques que el depositante ordenó pagar durante el mes fueron pagados por el banco, si se encontró alguna diferencia se anotan en una hoja de borrador (concepto, fecha y monto).

Tercer paso
Se comparan libro auxiliar de bancos y el estado de cuenta para determinar si los débitos y créditos que aparecen en el estado de cuenta están registrados en el libro auxiliar de bancos del depositante. Si no están registradas se anotan en la hoja de borrador.

Las partidas más comunes que generan los ajustes (tras realizar la conciliación) son:

- Cheques expedidos y no registrados.
- Comisiones bancarias e impuestos cargados en cuenta.
- Intereses a favor abonados por el banco.
- Depósitos en cuenta bancaria y no registrados.

Las diferencias encontradas y detalladas en la hoja de borrador se analizan para determinar:

- Si obedecen a errores cometidos por el banco. En este caso debe informarse al banco oportunamente.
- Si obedece a errores u omisiones al registrar las transacciones en el libro auxiliar de bancos. En este caso se procede a corregir los errores.

2.1.4 Beneficios de la conciliación bancaria

Las conciliaciones pueden detectar y prevenir el fraude intencional, junto con los errores de cajeros, contadores, empleados y la gerencia. A pesar de que la conciliación bancaria es típicamente un procedimiento que se realiza cada mes, las empresas con movimientos de efectivo más pequeños podrían llevarla a cabo todos los días.

Detectar fraudes
Las conciliaciones bancarias ayudan a revelar las actividades fraudulentas debido a que éstas llevan una revisión cuidadosa basada en controles y procedimientos adecuados, tanto en los cheques desembolsados de una empresa, como los cheques compensados en el estado de cuenta de la empresa.

Identifica los errores bancarios
Los empleados del banco pueden cometer errores bancarios con regularidad, tal como transponer números, registrar cantidades erróneas, registrar la cantidad correcta en otra cuenta bancaria, entre otras. La conciliación de la cuenta bancaria da a la empresa tiempo para notificar al banco de su error, lo que permite al banco investigar y corregir la discrepancia de la cuenta.

2.2 Base de datos

Una base de datos es una colección de datos organizados o clasificados que se relacionan entre sí y que se desarrollan bajo un mismo contexto a partir de la información requerida por una organización.

El hecho de utilizar una base de datos provee de ciertos beneficios, tales como:

- Una interfaz estándar para acceder a los datos
- La capacidad de manipular grandes volúmenes de datos
- Seguridad
- Flexibilidad y rapidez al obtener datos

Normalmente los datos que las aplicaciones procesan se almacenan en bases de datos. Por ello durante el desarrollo de una aplicación, se debe realizar el diseño de la base de datos (definir su contenido y estructura, tales como qué tablas va a contener, qué campos tendrá cada tabla, qué claves habrá para cada tabla, etc.).

2.2.1 Modelo entidad-relación

El modelo entidad-relación (E-R) es uno de los varios modelos conceptuales existentes para el diseño de bases de datos. El propósito de este modelo es simplificar el diseño de bases de datos a partir de descripciones textuales de los requerimientos.

El modelo E-R es un modelo de datos basado en una percepción del mundo real que consiste en un conjunto de objetos básicos llamados entidades y de relaciones entre estos objetos, implementándose en forma gráfica a través del diagrama E-R.

Una entidad es un objeto del mundo real sobre el que queremos almacenar información, que se distingue de los demás objetos porque posee características que la hacen única y las relaciones representan asociaciones del mundo real entre una o más entidades.

2.2.2 Modelo relacional

El modelo relacional es un modelo lógico que establece una estructura sobre los datos. Así mismo es un modelo de datos donde estos se representan en forma de tablas. Este modelo se puede obtener a partir del modelo entidad-relación.

Cada entidad (tabla) almacena información sobre un tema, ésta dispone de columnas (campos) que contienen los diferentes tipos de información sobre ese tema y de filas (registros o tuplas) que describen todos los atributos de una única instancia del tema.

Por ejemplo, llámese la entidad autores, que contiene la información del nombre y las novelas, cuentos, etc. de un determinado autor. En este caso, los campos serían el nombre y el título del libro; mientras que un registro sería una estancia específica como: Gabriel García Márquez con su obra cien años de soledad.

Para la mayoría de las aplicaciones de gestión que utilizan bases de datos, el modelo más empleado es el modelo relacional por su gran versatilidad, potencia y por los formalismos sobre los que se basa.

2.2.3 Sistema manejador de base de datos

El sistema manejador de base de datos, también conocido como **DBMS** por sus siglas en inglés: Database Management System, es uno de los elementos fundamentales de una base de datos. Consiste en un conjunto de módulos preprogramados, cada uno de los cuales es responsable de una tarea específica, tal como crear y organizar la base de datos, establecer la trayectoria de acceso a la misma, manejar o manipular los datos de acuerdo a la petición de los usuarios, etc.

Su importancia radica en que el DBMS será la interfaz entre los datos de bajo nivel almacenados en la base de datos y los programas de aplicaciones, así como las consultas hechas al sistema.

2.2.4 SQL

El acrónimo **SQL** es la abreviatura de Structured Query Languaje (Lenguaje estructurado de consultas). Es una herramienta para la organización, gestión y recuperación de los datos almacenados en bases de datos informáticas.

Es un lenguaje que se utiliza para interactuar con bases de datos relacionales y controlar las funciones que el DBMS ofrece a los usuarios, entre las que se hallan:

*** Definición de los datos:** permite que el usuario defina la estructura y la organización de los datos almacenados y las relaciones entre los elementos de datos almacenados.

*** Recuperación de los datos:** permite que el usuario o un programa de aplicación recupere de la base de datos los datos almacenados y los emplee.

*** Manipulación de los datos:** permite que el usuario o un programa de aplicación actualice la base de datos añadiendo datos nuevos, eliminando datos antiguos y modificando los datos almacenados previamente.

*** Control de acceso:** puede utilizarse para restringir la capacidad del usuario para recuperar, añadir y modificar datos, protegiendo así los datos almacenados contra los accesos no autorizados.

*** Integridad de los datos:** define restricciones de integridad en la base de datos, protegiéndola así del deterioro debido a las actualizaciones inconsistentes o a los fallos del sistema.

2.3 Lenguajes de programación

Un lenguaje de programación es una notación o conjunto de símbolos y caracteres combinados entre sí de acuerdo con una sintaxis ya definida que posibilita la transmisión de instrucciones a la CPU.

Un lenguaje de programación en el ámbito de la informática, es un lenguaje artificial, compuesto por un vocabulario fijo y un conjunto de reglas (llamadas sintaxis), que se usan para crear instrucciones que la computadora debe seguir. Casi todos los programas se escriben con un editor de texto o un programa de procesamiento de texto para crear un código fuente, que se interpreta o compila después a un lenguaje de máquina que la computadora puede ejecutar.

En la actualidad, y en una sociedad industrializada como la nuestra, donde los avances tecnológicos van a un paso desenfrenado, el hombre busca cada vez más la reducción o eliminación de tareas rutinarias y repetitivas, sobre todo, a la hora de gestionar o procesar información. Con el fin de alcanzar este objetivo, surge la *informática*, siendo definida como la ciencia del tratamiento automático y racional de la información, así como soporte del conocimiento y las comunicaciones.

A pesar de que los adelantos conseguidos han sido enormes después de largos años de estudio, el ordenador sigue siendo incapaz de realizar las tareas por sí mismo sin la ayuda del razonamiento humano.

De la misma forma que el comportamiento y el pensamiento humano se rigen por métodos de razonamiento lógicos que nos permiten la realización de acciones o tareas concretas, el comportamiento o actuación de un ordenador se rige por lo que denominamos *programación*, entendido por tal como el desarrollo y puesta en marcha de soluciones a problemas concretos, mediante una secuencia de instrucciones o conjunto de acciones lógicas que debe ejecutar el ordenador y que son transmitidas a éste por la figura del *programador* en forma de *programa*.

El programa es una secuencia de acciones o actividades determinadas por el programador. A cada una de estas actividades se le llama instrucción.

Un programa es una entidad dinámica que sólo existe al momento de ejecutarse, es aquí donde adquiere su verdadero valor. Antes o después de esto sólo es un conjunto de instrucciones que ocupan espacio en algún dispositivo de almacenamiento.

Al ser una entidad dinámica, además de no poseer una característica cuantificable hace de un programa, una entidad sumamente abstracta.

El uso de la abstracción en el diseño de software (que se define como un conjunto de programas) ha permitido el uso de lenguajes de programación de alto nivel, donde el programador se preocupa sólo de la solución de los problemas y no de la arquitectura de la máquina.

Lo que hace que un programa se pueda considerar como bueno se refiere definitivamente al plano económico y en este punto se deben de considerar varios factores de calidad, que son:

1. **Utilidad.** Un programa debe satisfacer las necesidades del usuario.

2. **Confiabilidad.** Es la capacidad de un programa para desempeñar una función requerida bajo ciertas condiciones durante un tiempo específico, determinado por la relación directa con el costo de una falla.

3. **Legibilidad.** Se refiere a la facilidad de lectura que tiene el programa.

4. **Estilo de programación.** Consiste en planificar el programa empezando con el análisis del problema (especificaciones de entrada y salida).

5. **Eficiencia.** Consiste en optimizar los recursos de la computadora y está en función del tipo de aplicación que se esté desarrollando.

Un programa es un modelo de la solución de un problema el cual debe reunir un requisito importante: ser computable (calculable). Esto significa que la solución a un problema se puede representar a través de un conjunto de proposiciones (a esto se le conoce como diseño de un algoritmo), determinando su grado de exactitud por los límites establecidos por razones económicas, de tiempo, de recursos tecnológicos y humanos.

Diseñar un algoritmo requiere de la abstracción de las partes más importantes y características del problema: el programa y los datos. Estos interactúan entre sí, pues mientras el programa provee los procedimientos necesarios para procesar los datos y así obtener información, los datos son el insumo de estos procedimientos.

La programación estructurada nos ayuda a realizar una abstracción de estas dos características, ya que no solo requiere estructurar los procedimientos sino también los datos simultáneamente. Esta división en unidades estructuradas permite fragmentar la solución de un problema en módulos reduciendo esfuerzo y principalmente costos en el desarrollo de software. Para lograr esta fragmentación se deben considerar dos premisas:

1 La resolución de un problema debe hacer uso de algún método de abstracción.
2 La partición del sistema en módulos redundará en un menor esfuerzo y un menor costo final del producto.

La primera premisa habla del método de abstracción; dentro de la programación estructurada existen dos métodos: el diagrama de flujo y el pseudocódigo. Sus funciones son las mismas lo que cambia es la forma de representarlos ya que mientras los diagramas de flujo son una representación gráfica de un algoritmo, el pseudocódigo es un conjunto de órdenes expresadas en frases cortas en español en cada una de las cuales existe uno y sólo un verbo. Ambas herramientas, principalmente el pseudocódigo permiten no tener ninguna relación con ningún lenguaje en particular, es decir, es independiente de la implementación.

La segunda premisa se sustenta en el principio de diseño modular:

a. Las partes altamente relacionadas del problema deben pertenecer a la misma pieza del sistema.
b. Las partes no relacionadas del problema deben residir con piezas no relacionadas del sistema.

Dentro de la programación, un módulo es un segmento de programa que posee un conjunto finito de entradas, salidas y ejecuta una sola función.

Para asegurarse que un programa sea construido correctamente se deben seguir estos pasos:

***** Analizar el problema. Antes de pensar en la solución de un problema es necesario conocerlo a fondo, saber cuáles son sus causas y cuáles sus componentes principales. Al finalizar esta fase se debe tener planteado el problema.

* Diseñar la solución. Para realizar un programa es necesario crear, antes que nada, un esquema que indique cómo se construirá dicho programa.

El diseño del programa es el equivalente al plano de construcción de una casa o edificio realizado por un arquitecto. Este diseño debe contener una explicación de todos los componentes del programa y se va complicando conforme va creciendo el problema. Para estos casos es necesario dividir el problema de manera que sea más sencillo resolver cada uno de los problemas que surgen de la subdivisión.

Por lo general, un programa cuenta con tres componentes: entrada, procesamiento y salida de información.

Uno de los componentes del diseño es el algoritmo que permite mostrar, por medio de un diagrama o semilenguaje estructurado, la forma en que se va a realizar el programa.

Las propiedades básicas que debe satisfacer todo algoritmo son las siguientes:

* En primer lugar el algoritmo debe ser correcto, es decir, debe solucionar el problema en todos los casos posibles, (si el algoritmo no puede resolver el problema por obvias razones éste no sirve).

* Además, debe estar compuesto por una serie dada de pasos y para cado uno de ellos no debe de existir ambigüedad en la definición de cuál es el siguiente.

* Los algoritmos deben de ser precisos, los resultados de los cálculos deben de ser exactos, de manera rigurosa, no es válido un algoritmo que sólo se aproxime a la solución.

*El número de pasos debe ser finito y por supuesto, el algoritmo debe terminar.

En relación con este último aspecto debe tenerse en cuenta que existen algoritmos que terminan y otros que no. Así por ejemplo el algoritmo de Euclides para calcular el máximo común divisor de dos números naturales presenta el resultado un número finito de pasos, y sin embargo el algoritmo para la expansión de la raíz cuadrada de un número natural en fracciones decimales no garantiza que termine completamente en un número finito de pasos ya que el

resultado puede ser irracional. Sin embargo en estos casos se entiende que el algoritmo termina introduciendo el concepto de precisión requerida o solución suficientemente terminada.

En general, existen diferentes algoritmos para resolver un mismo problema. Evidentemente esto permite elegir el mejor.

Son dos los factores fundamentales a considerar en el costo de un algoritmo: El tiempo de ejecución y el espacio de almacenamiento requerido para implementarlo.

La importancia del primero de ellos es muy evidente, el algoritmo deberá en primer lugar finalizar obteniéndose la solución deseada en un tiempo adecuado según los requerimientos del problema. Y en segundo lugar es útil para comparar distintos algoritmos aplicables para solucionar el mismo problema.

En términos de programación a cada instrucción se le llama declaración.
Para escribir programas es necesario conocer un lenguaje de programación. El lenguaje de programación proporciona un medio, sin ambigüedades, para generar instrucciones a la computadora.

Los componentes más importantes de un sistema computacional son: los programas, el equipo de cómputo, la gente y los datos o información que se procesa.

El objetivo principal de la programación es crear el conjunto de instrucciones necesarias para hacer funcionar las computadoras, las cuales son necesarias en muchas de las cosas que utilizamos actualmente.

Algunas de las actividades en las que se usa actualmente la programación son:

- Recolección, almacenamiento y explotación de la información generada por las transacciones diarias de una empresa.
- Creación de juegos de video.
- Control del funcionamiento de un robot.
- Mantenimiento óptimo de un automóvil.
- Control del vuelo de un jet.
- Control de las actividades de ejecutivos.
- Creación de simuladores virtuales para diferentes ámbitos como ingeniería, medicina, educación, etcétera.

El programador debe estructurar sus procesos correctamente para realizar las tareas de la manera más eficiente y de esta forma sacar el máximo provecho de los recursos computacionales.

Las características que debe tener un buen programador son:

- Creatividad

- Imaginación
- Disciplina
- Dedicación
- Estudio

El método que debe seguir un programador para solucionar adecuadamente un problema es:

- Reconocer el problema: Esta primera etapa involucra el estar atento a los retos y a las oportunidades; el propósito de esta etapa es identificar claramente los objetivos deseados, así como los desafíos implícitos y explorar las oportunidades que se vislumbran.
- Registrar la información: El propósito de esta etapa es obtener la información necesaria para una mejor comprensión de la situación; poco a poco se va separando la información relevante y descartando la que no tiene importancia. La información se registra para formular los problemas y generar nuevas ideas.
- Formular problemas: Esta fase se refiere a definir en términos generales en qué consiste cada problema, determinar si merece nuestra atención, y obtener una buena perspectiva del problema en una situación dada.
- Buscar ideas: Esta fase es de intensa actividad creativa ya que se persigue generar un conjunto de alternativas posibles para satisfacer las necesidades establecidas en las fases anteriores. Hay que realizar algunas preguntas basadas en los principios siguientes: sustituir, combinar, adaptar, modificar, minimizar, maximizar, considerar otros usos, reacomodar, revertir.
- Identificar y evaluar soluciones: Ya que se propusieron las soluciones, es necesario hacer una valoración desde tres puntos básicos (físico, económico y financiero) para elegir las soluciones potenciales más prácticas teniendo como objetivo poner orden, eliminar las peores alternativas y comparar las restantes con otras en base a criterios para evaluar soluciones y transformarlas en unas que se adapten a las circunstancias del momento para así seleccionar la de mayor interés.
- Aceptar e implementar: En esta etapa se eliminan barreras y obstáculos durante la implantación preparando un plan de acción determinando un responsable junto con las siguientes especificaciones: qué deben de hacer, en dónde se debe de realizar, cuándo se deberá de iniciar y terminar, qué recursos y procedimientos se requieren para su buena ejecución.

En el ámbito informático, una filosofía de programación es un conjunto de técnicas y métodos para estructurar en forma uniforme el diseño del conjunto de instrucciones necesarias para ejecutar una tarea específica en la computadora, buscando que el proceso sea confiable, íntegro y eficiente.

Existen tres tipos de filosofías de programación:

- La programación estructurada.
- La programación orientada a objetos.

- La programación orientada a eventos.

La **programación estructurada** es una filosofía de implantación de algoritmos con un conjunto finito de estructuras bien organizadas. Esta filosofía está enfocada a los programas; es decir, a la forma en que se resolverán los problemas. Al implementar los procedimientos y funciones se aplican las estructuras de datos adecuadas a estos y se obtienen los programas.

La **programación orientada a objetos** analiza las entidades implicadas en el programa, buscando características comunes entre entidades y de ahí crea estructuras de datos que capturen estas características permitiendo crear relaciones entre las estructuras de datos también denominados objetos. Las propiedades de la programación orientada a objetos son: abstracción, herencia, encapsulamiento y polimorfismo.

La **programación orientada a eventos** se enfoca a las acciones del usuario como punto principal de la programación. Esta filosofía basa sus características en crear funciones en base a los eventos que produce el usuario. Este tipo de programación es sensiblemente más complicada que la secuencial y la interactiva, ya que las acciones de un usuario son muy variadas.

Clasificación de los lenguajes:

Lenguajes de bajo nivel: son aquellos que por sus características se encuentran más próximos a la arquitectura de la máquina, englobándose en este grupo el lenguaje máquina y el lenguaje ensamblador.

Los primeros lenguajes eran de bajo nivel y estaban directamente ligados al hardware donde iban a ser ejecutados, cada computadora tenía su propio lenguaje, el cual se conocía como lenguaje máquina.

Cualquier computadora puede entender de manera directa sólo su propio lenguaje máquina, y como tal, está definido por el diseño del hardware de dicha computadora. Por lo general los lenguajes máquina consisten en cadenas de números que instruyen a las computadoras para realizar sus operaciones más elementales, una a la vez.

Los lenguajes máquina son dependientes de la máquina, es decir, un lenguaje máquina en particular puede usarse solamente en un tipo de computadora. Dichos lenguajes son difíciles de comprender para los humanos.

La programación en el lenguaje máquina era demasiado tediosa para la mayoría de los programadores, en vez de utilizar las cadenas de números que las computadoras podían entender directamente, los programadores empezaron a utilizar abreviaturas en inglés para representar las operaciones elementales. Estas abreviaturas formaron la base de los lenguajes ensambladores.

Los programas traductores conocidos como ensambladores se desarrollaron para convertir los primeros programas en lenguaje ensamblador a lenguaje máquina. Sin embargo se requería de muchas instrucciones para llevar a cabo incluso hasta las tareas más simples.

Para agilizar el proceso de programación se desarrollaron los lenguajes de alto nivel, en donde podían escribirse instrucciones individuales para realizar tareas importantes.

Lenguajes de alto nivel: son aquellos que por sus características se encuentran más próximos al usuario o programador y se consideran como tales el resto de los lenguajes de programación como por ejemplo BASIC, COBOL, Pascal, C, Java, etc.

Una de las características más importantes de estos lenguajes es que, a diferencia de los lenguajes de bajo nivel, son independientes de la arquitectura del ordenador utilizado como soporte, lo que implica que los programas desarrollados en lenguajes de alto nivel puedan ser ejecutados sobre ordenadores con distinto microprocesador. Por otro lado, cabe destacar una mayor facilidad en el desarrollo, depuración y mantenimiento de los programas frente a los desarrollados con lenguaje de bajo nivel.

Como inconveniente destacable, cabe señalar la necesidad de traducir los programas escritos en un lenguaje de alto nivel a un lenguaje de programación tan primitivo como el lenguaje máquina para que pueda ser interpretado y ejecutado por la unidad central de proceso, lo que significa disponer necesariamente de un traductor (ensamblador, compilador o intérprete) para cada tipo de ordenador utilizado.

Fases de elaboración de un programa informático

El desarrollo de una aplicación o conjunto de programas para obtener una solución informática a un determinado problema se basa en un concepto denominado ciclo de vida, que establece una serie de etapas o fases que hay que seguir secuencialmente y de forma ordenada cuando se desea desarrollar un determinado producto de software.

Se pueden considerar las siguientes fases:

Análisis
A lo largo de esta primera fase se establece cuál es el producto que se va a desarrollar, siendo necesario especificar los procesos y estructuras de datos que se van a emplear, para satisfacer las necesidades del usuario, por lo que debe existir una gran comunicación entre el usuario y el analista para conocer todas las necesidades y restricciones en el desarrollo de la aplicación. En aquellos casos en los que exista ambigüedad o falta de información por parte del usuario, puede ser necesario el desarrollo de prototipos que sirvan para definir con precisión sus requerimientos.

Diseño
En esta fase se alcanza una solución óptima, detallada y con la mayor precisión posible, teniendo en cuenta los recursos físicos del sistema (tipo de ordenador, periféricos, comunicación, etc.) y los recursos lógicos (sistema operativo, programas de utilidad, compiladores, bases de datos, etc.).

Codificación
Consiste en la traducción de la solución obtenida a un determinado lenguaje de programación, (basada en las especificaciones de diseño expresadas) basándose en las especificaciones de diseño expresadas.

Pruebas
En esta fase se realiza la implantación de los programas (aplicación) en el entorno operativo o sistema físico donde van a funcionar habitualmente y su puesta en marcha para obtener un funcionamiento normal de todo el sistema. Se trata de comprobar que el funcionamiento de la aplicación es la adecuada.

Mantenimiento
Esta fase completa el ciclo de vida y en ella se realizan las correcciones necesarias para subsanar errores y deficiencias del producto desarrollado, existiendo la posibilidad de que ciertas acciones de esta fase puedan reiniciar el ciclo de vida.

El tiempo invertido en la actividad de mantenimiento depende en gran medida de la correcta evaluación de las fases anteriores, así como de la documentación desarrollada.

2.3.1 Lenguaje Java

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principio de los años 90´s.

Los creadores de Java son James Gosling (emacs) y Bill Joy (Sun). Java desciende de un lenguaje llamado Oak cuyo propósito era la creación de software para la televisión interactiva. Sus características eran:

- Pequeño.
- Robusto.
- Independiente de la máquina.
- Orientado a objetos.

El proyecto de televisión interactiva fracasó y el interés de los creadores de Oak se dirigió a Internet.
Los criterios de diseño de Java fueron:

- Independiente de la máquina.
- Seguro para trabajar en red.
- Potente para sustituir código nativo.

Un programa Java está formado por un conjunto de clases que interactúan entre sí.
La clase es la unidad básica de programación.

Para realizar una tarea en una aplicación se requiere un método. El método describe los mecanismos que se encargan de realizar sus tareas y oculta al usuario las tareas complejas que realiza. En Java, se empieza por crear una unidad de aplicación llamada clase para alojar a un método. En una clase se proporcionan uno o más métodos, los cuales están diseñados para realizar las tareas de esa clase.

Para poder hacer que un programa realice las tareas que la clase le describe cómo realizar, se debe construir un objeto de una clase. Ésta es una de las razones por las cuales Java se conoce como un lenguaje de programación orientada a objetos.

Para indicar a un método del objeto que realice su tarea, se envían mensajes al objeto, a estos mensajes se les conoce como llamada a un método.

Además de los métodos un objeto tiene atributos que lleva consigo cuando se utiliza en un programa. Las características del objeto se expresan a través de atributos. Estos se especifican como parte de la clase del objeto. Por ejemplo, un objeto tipo cuenta bancaria tiene un atributo llamado saldo, el cual representa la cantidad de dinero en la cuenta. Cada objeto tipo cuenta bancaria conoce el saldo en la cuenta que representa, pero no los saldos de las otras cuentas en el banco. Los atributos se especifican mediante las variables de instancia de la clase.

Como Java es un lenguaje de programación orientado a objetos maneja ciertas características principales las cuales son: abstracción, encapsulamiento, herencia y polimorfismo.

Abstracción, el diseño orientado a objetos modela el software en términos similares a los que utilizan las personas para describir objetos del mundo real. Todos ellos tienen atributos (como tamaño, forma, color y peso) y todos exhiben comportamientos por ejemplo (un automóvil acelera, frena y da vuelta). Distintos objetos pueden tener atributos similares y pueden exhibir comportamientos similares. Así que cada objeto puede distinguirse de los demás.

Encapsulamiento, es un mecanismo que se encarga de aislar el estado (atributos) de un objeto para que éste no sea modificado o leído por otros programas. Los atributos y las operaciones de un objeto se enlazan íntimamente entre sí. Los objetos tienen la propiedad de ocultamiento de información. Esto significa que los objetos pueden saber cómo comunicarse entre sí a través de interfaces bien definidas, pero por lo general no se les permite saber cómo se implementan otros objetos; los detalles de la implementación se ocultan dentro de los mismos objetos.

La herencia, es donde las nuevas clases de objetos se derivan absorbiendo las características de las clases existentes y agregando sus propias características únicas. Es una forma de reutilización de software en la que se crea una nueva clase absorbiendo los miembros de una clase existente, y se mejoran con nuevas capacidades o con modificaciones en las capacidades ya existentes.

Al crear una clase, en vez de declarar miembros completamente nuevos, el programador puede designar que la nueva clase herede los miembros de una clase existente. En Java a esta clase existente se conoce como superclase y la nueva clase se conoce como subclase. Cada subclase puede convertirse en la superclase de futuras subclases.

Una subclase generalmente agrega sus propios campos y métodos. Por lo tanto una subclase es más específica que su superclase y representa a un grupo más especializado de objetos. Generalmente, la subclase exhibe los comportamientos de su superclase juntos con comportamientos adicionales específicos de esta subclase.

El polimorfismo nos permite programar en forma general, en vez de programar en forma específica. Permite escribir programas que procesen objetos que compartan la misma superclase en una jerarquía de clases, como si todos fueran objetos de la superclase. Esto se permite, ya que cada objeto de una subclase es un objeto de su superclase.

La ejecución depende de la clase pública a la que se invoque con el intérprete.
La principal característica de Java es la de ser un lenguaje compilado e interpretado. Todo programa en Java ha de compilarse y el código que se genera bytecodes es interpretado por una máquina virtual. El código compilado se ejecuta en máquinas virtuales que si son dependientes de la plataforma.

Java es un lenguaje orientado a objetos de propósito general. En el diseño de Java se prestó especial atención a la seguridad. Existen varios niveles de seguridad en Java, desde el ámbito del programador, hasta el ámbito de la ejecución en la máquina virtual.

Todas las instancias de una clase se crean con el operador new( ), de manera que un recolector de basura se encarga de liberar la memoria ocupada por los objetos que ya no están referenciados. La máquina virtual de Java gestiona la memoria dinámicamente.

Las herramientas de desarrollo de Java se conocen como Java Development Kit (JDK). Este conjunto de herramientas cuenta entre otros con un compilador de línea de comandos javac; la máquina virtual de Java con la cual se pueden ejecutar aplicaciones Java; una herramienta de documentación javadoc; y una herramienta para empaquetar proyectos jar.

Se debe pensar en Java no solamente como un lenguaje de programación si no como un conjunto de tecnologías basadas en el mismo lenguaje. Este conjunto de tecnologías permite

escribir aplicaciones para gráficos, multimedia, la web, programación distribuida, bases de datos y otras más.

JAR (Java ARchives)
Es el formato de archivos utilizados para empaquetar los componentes, una aplicación en Java está compuesta por varios ficheros .java. Al compilarlos obtenemos varios ficheros .class (uno por fichero .java), y no un único fichero ejecutable como ocurre en otros lenguajes. A menudo la aplicación está formada no sólo por los ficheros .class sino que usa ficheros de sonido (usualmente .au en Java), iconos, etc., lo que multiplica la cantidad de ficheros que forman la aplicación compilada. Esto hace que "llevarse" la aplicación para ejecutarla en un ordenador diferente resulte un poco complicado: olvidar cualquiera de los ficheros que componen la aplicación significaría que ésta no va a funcionar correctamente.

Los ficheros Jar permiten recopilar en un sólo fichero varios ficheros diferentes, almacenándolos en un formato comprimido para que ocupen menos espacio. Es por tanto, algo similar a un fichero .zip (de hecho están basados en ficheros .zip). La particularidad de los ficheros .jar es que no necesitan ser descomprimidos para ser usados, es decir que el intérprete de Java es capaz de ejecutar los archivos comprimidos en un archivo jar directamente.

El JRE (Java Runtime Environment) es una máquina virtual de Java y su función es hacer de intermediario entre una aplicación programada en Java y el sistema operativo que se este usando. De este modo, cualquier aplicación puede funcionar en cualquier sistema operativo que disponga del JRE.

2.3.2 Lenguaje Visual Basic for Applications (VBA)

La programación en Office se emplea mediante el uso del VBA (Visual Basic para Aplicaciones), el cual es un lenguaje de programación que ofrece un conjunto de comandos que manipulan los objetos de cada una de las aplicaciones de Microsoft Office.

Gracias a la programación mediante el VBA, se pueden realizar diversas tareas dentro de Microsoft Word y Microsoft Excel. Algunas de las tareas que se pueden realizar en Excel son:

- Controlar libros de trabajo
- Controlar hojas de cálculo
- Manipular celdas
- Manejar datos dentro de las celdas
- Aplicar formatos a la hoja de cálculo
- Manipular filas y columnas
- Aplicar fórmulas y funciones
- Crear sus propias funciones y cálculos

- Crear y manipular las gráficas
- Manipular bases de datos externas

Con VBA se pueden automatizar las tareas que se ejecutan con cierta frecuencia y configurar de una manera más inteligente. En lugar de ejecutar varias opciones de un menú sucesivamente, se puede crear una macro en VBA que podrá ejecutar todas sus tareas mediante un botón de la barra de herramientas o una combinación de teclas. De igual manera, dentro de la misma macro se puede determinar que ésta tome determinadas decisiones por su cuenta.

Con VBA se puede interactuar con todas las aplicaciones de Office con el objetivo de intercambiar información entre ellas.

También se pueden desarrollar aplicaciones interactivas con una interfaz gráfica, la cual permite, a través de formularios, solicitar información al usuario que posteriormente será procesada por la aplicación.

2.3.2.1 Macros

Una macro es un conjunto de instrucciones que sirven para automatizar procesos que el usuario realiza frecuentemente dentro de una aplicación.

Técnicamente, una macro es una serie de comandos y funciones que se almacenan en un módulo de Visual Basic y que pueden ser ejecutados en el momento que se requiera a través de las siguientes formas:

- Combinación de teclas
- Botón de la barra de herramientas
- Opción de Menú
- Botón de Comando

En Microsoft Word y Microsoft Excel se pueden crear macros grabadas y también macros programadas.

Macros grabadas
Para la creación de una macro grabada, primero se debe planear toda la serie de acciones que se desean grabar, ya que el proceso de grabar una macro consiste en que la aplicación irá registrando todas las acciones o pasos que el usuario realice con el teclado o mouse durante la grabación, por lo que en caso de que se llegue a cometer un error durante este proceso, el error también será registrado en la macro.

Macros programadas

Las macros programadas tienen el objetivo de realizar macros más avanzadas de las que puede ofrecer el método de grabación de macros, que aunque es el más sencillo para realizar macros, está muy limitado y no puede resolver completamente las necesidades del usuario, por lo que al adentrarse en la programación que da origen a las mismas macros, podrá ajustar y explotar todas las herramientas de Office según sus necesidades.

2.3.2.2 Modelo de Objetos de VBA

Microsoft Office ofrece al programador la posibilidad de acceder y manipular la mayoría de sus funciones y herramientas a través de objetos mediante la programación en VBA.

Un objeto es una parte específica de una aplicación, el cual está identificado por un nombre. Además, el objeto puede contener otros *objetos, métodos y propiedades*. El modo más sencillo para familiarizarse con los objetos es mediante la comparación con cualquier objeto del mundo real, como por ejemplo una casa, un automóvil, etcétera.

Un método es una orden, acción o comportamiento que un objeto puede realizar, por ejemplo una acción de un automóvil es acelerar, frenar, abrir una puerta, etcétera.

Las propiedades son las características particulares que pueden tener un objeto, por ejemplo, en el caso de un automóvil:
Color = rojo
Ancho = 2 m
Alto = 2 m
Largo = 4 m
Marca = “Nissan”

3.0 Análisis de los requerimientos de la aplicación

En este capítulo se describirán los requerimientos necesarios para cumplir el objetivo del proyecto el cual consiste en automatizar el proceso de una conciliación bancaria que en la actualidad se realiza de manera manual o bien, a través de aplicaciones que requieren una arquitectura cliente – servidor y cuya desventaja es que no proporcionan seguridad de la información.

Para las empresas es indispensable llevar a cabo un control apropiado de sus movimientos financieros, con la finalidad de invertir una menor cantidad de recursos. Es por ello que se decidió desarrollar una aplicación que ofrezca una disminución de la carga de trabajo y mantenga segura la información.

De acuerdo a lo visto en el punto 2.1.3, del procedimiento general para elaborar una conciliación bancaria, la aplicación requiere que se haga uso del concepto, el monto y la fecha tanto del libro auxiliar de banco y el estado de cuenta.

	A	B	C
1	**BANAMEX, SA.B.**		
2	**CUENTA 5210 (MONEDA NACIONAL)**		
3	**ESTADO DE CUENTA DEL 01 AL 30 DE JULIO 2012**		
4			
5	**FECHA**	**CONCEPTO**	**MONTO**
6	7/1/2012	PAGO DE CHEQUE 101	$10,450.00
7	7/1/2012	PAGO DE NOMINA 309	$159,000.00
8	7/1/2012	DEPOSITO EN EFECTIVO SUC. 23	$3,780.00
9	7/1/2012	PAGO MOLINOS AZTECA 0026	$20,000.00
10	7/1/2012	PAGO CHEQUE 102	$10,500.00
11	7/1/2012	PAGO CHEQUE 103	$11,500.00
12	7/2/2012	PAGO MINERALES EXPANDIDOS 0027	$56,400.00
13	7/2/2012	ENMEX, S.A. DE C.V. 00596	$103,200.00
14	7/2/2012	ALIMENTOS LA CONCORDIA, S.A. DE C.V. 02596	$2,690.00
15	7/2/2012	DEPOSITO EN EFECTIVO SUC. 186	$24,700.00
16	7/2/2012	DEPOSITO EN EFECTIVO SUC. 035	$1,000.00
17	7/3/2012	PAGO DE CHEQUE 105	$50,000.00
18	7/3/2012	PAGO METLER TOLEDO, S.A. DE C.V.	$15,000.00
19	7/4/2012	PAGO POLICIA AUXILIAR DEL ESTADO DE MEXICO	$45,000.00
20	7/4/2012	PAGO DE CHEQUE 106	$2,600.00
21	7/4/2012	PAGO DE CHEQUE 107	$1,390.00
22	7/4/2012	TRASPASO 025634564	$100,000.00
23	7/5/2012	PAGO DE CHEQUE 108	$1,000.00
24	7/5/2012	PAGO DE CHEQUE 109	$1,230.00
25	7/5/2012	PAGO DE CHEQUE 110	$2,450.00
26	7/5/2012	PAGO SERVICIO ADMINISTRACIÓN TRIBUTARIA	$56,000.00
27	7/5/2012	PAGO TRASPORTES CASTORES, S.A. DE C.V	$21,300.00
28	7/6/2012	DEPOSITO EN EFECTIVO SUC. 310	$45,600.00
29	7/7/2012	DEPOSITO EN EFECTIVO SUC. 130	$10,000.00
30	7/7/2012	DEPOSITO EN EFECTIVO SUC. 330	$15,800.00
31	7/7/2012	DEPOSITO EN EFECTIVO SUC. 390	$23,000.00

Imagen 3.0.1 Datos requeridos para una conciliación bancaria

En la Imagen 3.0.1 se muestra un ejemplo correspondiente a los datos de entrada tanto del libro auxiliar de bancos como del estado de cuenta de la empresa.
Para trabajar con estos conceptos, la aplicación debe de tener la capacidad de almacenar dichos datos, procesarlos y entregar los resultados esperados.

A través de la especificación de requerimientos se determinarán las funciones principales con las que la aplicación deberá contar.

3.1 Especificación de requerimientos

La aplicación deberá ser eficiente, óptima y práctica, además de contar con una interfaz gráfica que permita al usuario interactuar sin problemas con los servicios que ésta proporcionará.

Las funciones principales que realizará serán:

1) Carga de datos
2) Procesamiento de la información
    a. Por concepto y monto
    b. Por monto y fecha
    c. Por monto
3) Presentación de un reporte final

3.2 Casos de uso

En el siguiente caso de uso se está representando el proceso que llevará a cabo el usuario al realizar una conciliación bancaria a través de la aplicación.

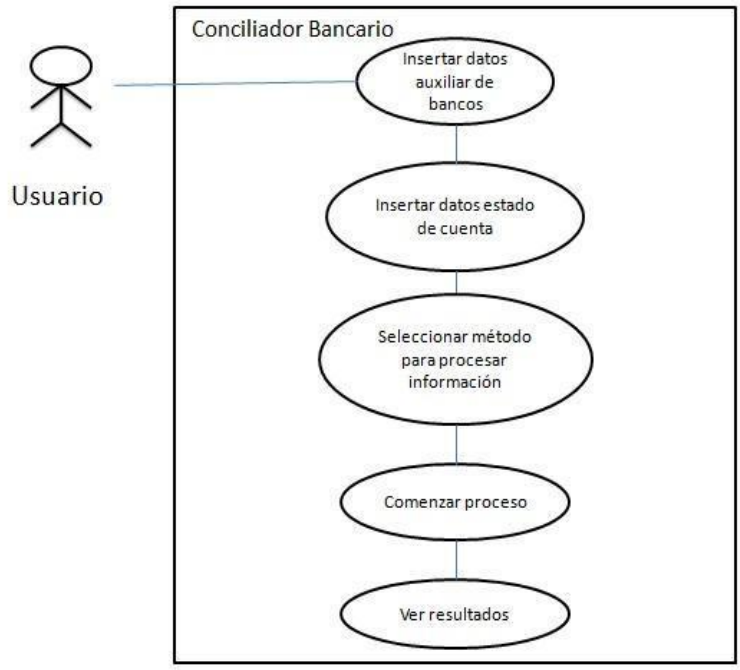


Imagen 3.2.0 Imagen que muestra los pasos de ejecución del conciliador bancario

FORMAS DE CASOS DE USO

Nombre	**Conciliador Bancario**
Actores	**Usuario**
Precondiciones	**El usuario cuenta con la información del libro auxiliar de bancos y el estado de cuenta**
Flujo normal	1. **El usuario ingresa los datos del libro auxiliar de bancos manualmente con ayuda de la interfaz gráfica.** 2. **El usuario ingresa los datos del estado de cuenta manualmente con ayuda de la interfaz gráfica.** 3. **El usuario selecciona el método por el cual desea procesar la información.** 4. **El usuario verifica que sus datos se ingresaron correctamente e inicia el proceso de conciliación.** **5. El usuario obtiene el resultado final.**

CASO DE USO APLICACIÓN ACTUAL

Conforme se desarrolló la aplicación, se realizaron una serie de modificaciones en el caso de uso, ya que se detectaron puntos de mejora. A continuación se presenta la segunda propuesta.

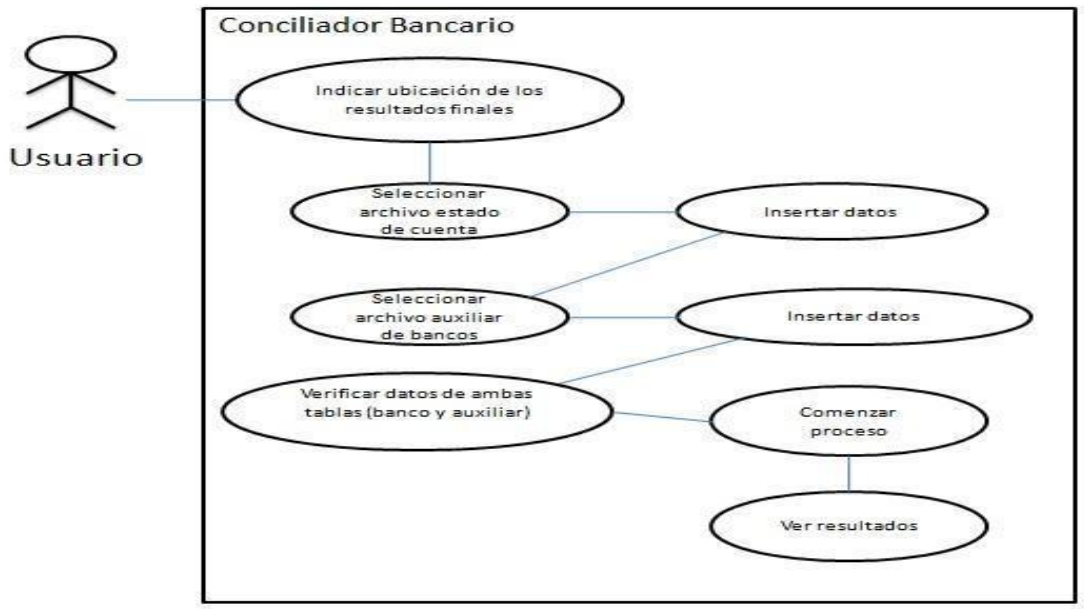


Imagen 3.2.1 Imagen que muestra ajustes al proceso de ejecución de la aplicación

FORMAS DE CASOS DE USO

Nombre	**Conciliador Bancario**
Actores	**Usuario**
Precondiciones	1. **El usuario cuenta con la información del libro auxiliar y el estado de cuenta.** 2. **El usuario valida que la información esté registrada en libros de Excel en el formato adecuado para su posterior uso.**
Flujo de eventos	1. **El usuario indica la ubicación en donde se creará el archivo que contiene los resultados.** 2. **El usuario indica la ubicación en donde se encuentra el archivo del extracto bancario con el formato definido en Excel.** 3. **El usuario almacena los datos en el sistema con ayuda de la interfaz gráfica implementada en Java.** 4. **El usuario indica la ubicación en donde se encuentra el archivo libro auxiliar de bancos en el formato definido en Excel.** 5. **El usuario almacena los datos en el sistema con ayuda de la interfaz gráfica implementada en Java.** **6. El usuario verifica que sus datos se ingresaron correctamente.** **7. El usuario inicia el proceso de conciliación, dicho proceso ejecuta los métodos correspondientes.** **8. El usuario obtiene el resultado final, a través de un reporte generado en Excel el cual es abierto desde la misma interfaz.**
Post-condiciones	**Se abre el archivo con los resultados obtenidos para poder realizar un análisis de los mismos.**

CASO DE USO APLICACIÓN ACTUAL

A continuación se presenta la tercer y última propuesta considerando incluso a todos los actores externos y las funciones que realiza el sistema.

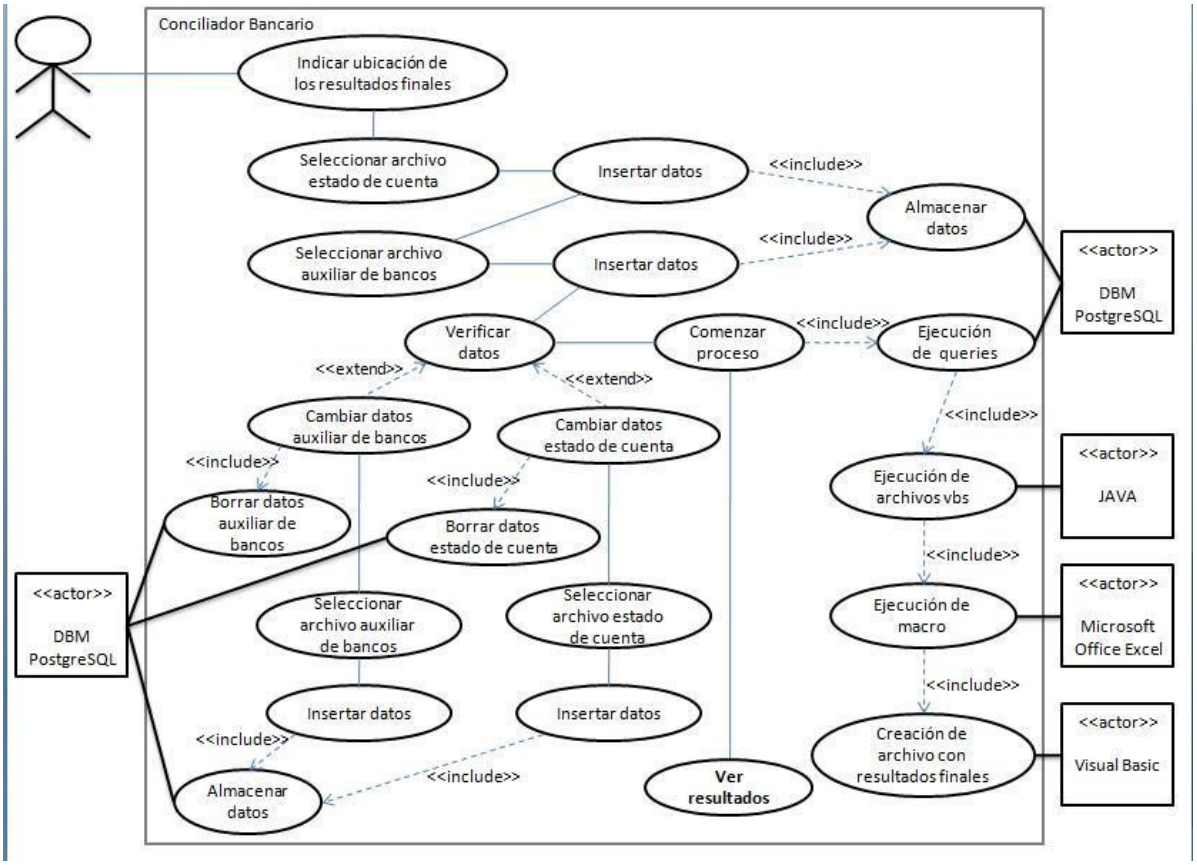


Imagen 3.2.2 Imagen que muestra el proceso de ejecución final de la aplicación

FORMAS DE CASOS DE USO

Nombre	**Conciliador Bancario**
Actores	**Usuario, Java, Microsoft Office Excel, Visual Basic y DBM PostgreSQL**
Precondiciones	1. **El usuario cuenta con la información del libro auxiliar y el estado de cuenta.** 2. **El usuario valida que la información esté registrada en libros de Excel en el formato adecuado para su posterior**

	uso.
Flujo de eventos	1. **El usuario indica la ubicación en donde se creará el archivo que contendrá los resultados.** 2. **El usuario indica la ubicación en donde se encuentra el archivo del extracto bancario con el formato definido en Excel.** 3. **El usuario almacena los datos en el DBM PostgreSQL con ayuda de la interfaz gráfica implementada en Java.** 4. **El usuario indica la ubicación en donde se encuentra el archivo libro auxiliar de bancos en el formato definido en Excel.** 5. **El usuario almacena los datos en el DBM PostgreSQL con ayuda de la interfaz gráfica implementada en Java.** 6. **El usuario verifica que sus datos se ingresaron correctamente.** 7. **El usuario inicia el proceso de conciliación, dicho proceso ejecuta algunos métodos implementados en Java, Visual Basic y SQL.** **8. El usuario obtiene el resultado final, a través de un reporte generado en Excel el cual es abierto desde la misma interfaz.**
Flujos alternos	**Después del paso 6 y antes de empezar el paso 7 si no se cargaron correctamente los datos, el usuario tiene la posibilidad de volver a ingresar los datos, regresando al paso 1.**
Post-condiciones	**Se abre el archivo con los resultados obtenidos para poder realizar un análisis de los mismos.**

3.3 Selección del manejador de base de datos

Al analizar los requerimientos de la aplicación, se determinaron dos posibilidades de uso para el manejo de los datos:

- Arreglos
- Bases de datos relacionales

Los arreglos (array en inglés) son un conjunto ordenado de elementos de datos. Se clasifican de acuerdo a sus dimensiones; así es como existen los denominados arreglos unidimensionales o vectores y los bidimensionales o matrices.

La mayoría de los lenguajes de programación tienen la capacidad de almacenar y manipular arreglos de más de una dimensión, tal es el caso particular de Java. Sin embargo, los arreglos presentan la desventaja de que no permiten almacenar datos de diferente tipo. Se puede tener un arreglo de números enteros, o un arreglo de cadenas, pero no se puede tener un arreglo que tenga enteros y cadenas a la vez; además, dependiendo de la dimensión del arreglo, se necesitará más de un índice para acceder a los datos almacenados en él.

Como ya se ha visto en capítulos anteriores, una base de datos presenta la facilidad de trabajar con conjuntos de datos de distintos tipos cuyas interrelaciones pueden ser ágilmente redefinidas en función de las necesidades cambiantes del desarrollador y/o usuario, proporcionando así mayor eficiencia en la aplicación, es decir, utilizará menos recursos del equipo.

Partiendo de esta evaluación del manejo de los datos, se decide que estos deberán ser almacenados y manipulados a través de una base de datos que sea capaz de establecer una trayectoria de acceso a la misma, manejando la información de acuerdo a las peticiones de los usuarios a través de la aplicación.

Cada usuario podrá acceder a la información de manera independiente sin requerir transacciones múltiples por lo que no habrá necesidad de manejar concurrencia a la base de datos, lo que facilita la disponibilidad y por supuesto la velocidad de navegación a la misma. Lo que implica una base de datos no robusta y con una sencilla administración.

Ante esto, se decide elegir el manejador de base de datos postgreSQL ya que además de cumplir con los requerimientos propios de la aplicación, nos brinda la ventaja de encontrarse distribuido bajo licencia BSD[1]; es decir, permite el uso del código fuente en software no libre.

[1] BSD son las siglas de “Berkeley Software Distribution”. Así se llamó a las distribuciones de código fuente que se hicieron en la Universidad de Berkeley en California y que en origen eran extensiones del sistema operativo UNIX® de AT&T Research.

4.0 Diseño de la Aplicación

En este capítulo describiremos una breve justificación de por qué utilizamos un manejador de base de datos; por qué hicimos modificaciones en el diseño de la aplicación (base de datos, macros, métodos y el ingreso de los datos).

Para normalizar la base de datos se tuvo que agregar una nueva tabla debido a la relación muchos a muchos que existía entre las tablas cheque_banco y cheque_auxiliar quedando de la siguiente forma:

- id_cheque_auxiliar
- id_cheque_banco
- tipo_conciliacion

Para nuestra aplicación se consideraron los siguientes modelos E-R.

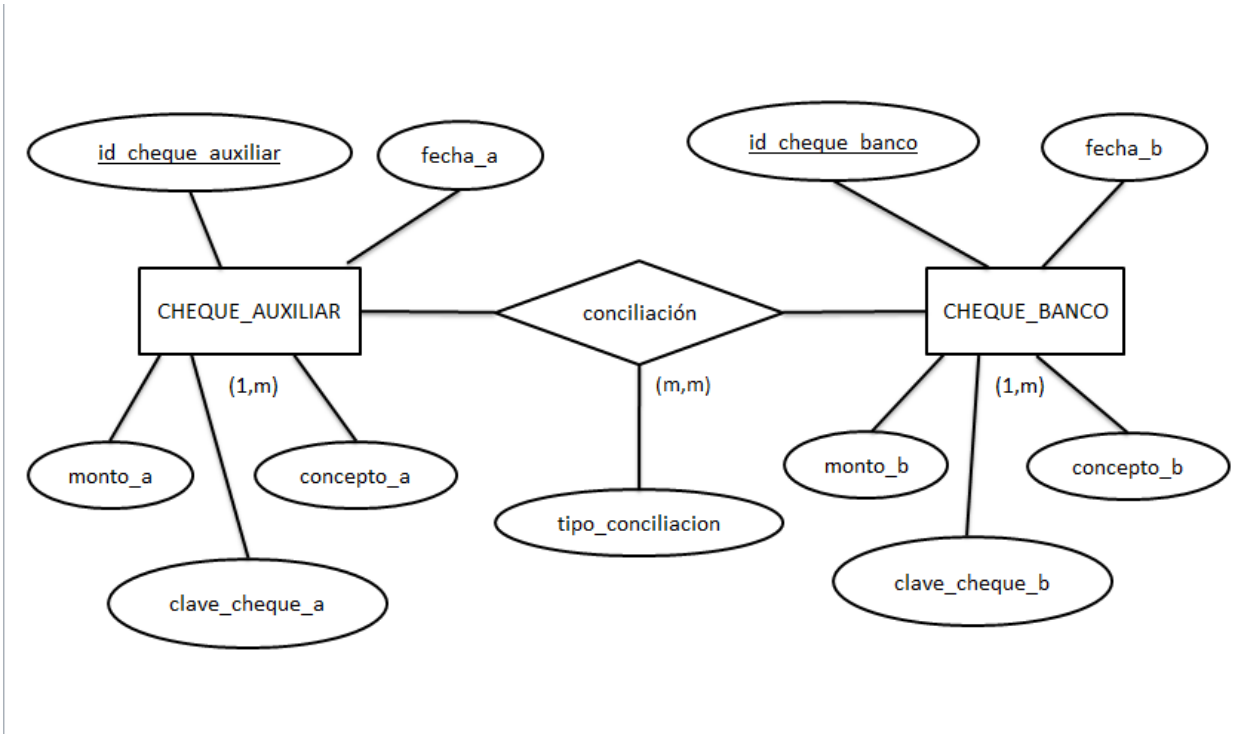


Imagen 4.0.0 Imagen que muestra las entidades cheque_auxiliar y cheque_banco junto con la relación "conciliación" después de normalizar.

El modelo relacional es el siguiente:

CHEQUE_AUXILIAR = {id_cheque_auxiliar (PK), fecha_a, monto_a, concepto_a, clave_cheque_a}

CHEQUE_BANCO = { id_cheque_banco (PK), fecha_b, monto_b, concepto_b, clave_cheque_b}

CONCILIACION = {[id_cheque_auxiliar (FK), id_cheque_banco (FK)](PK), tipo_conciliacion}

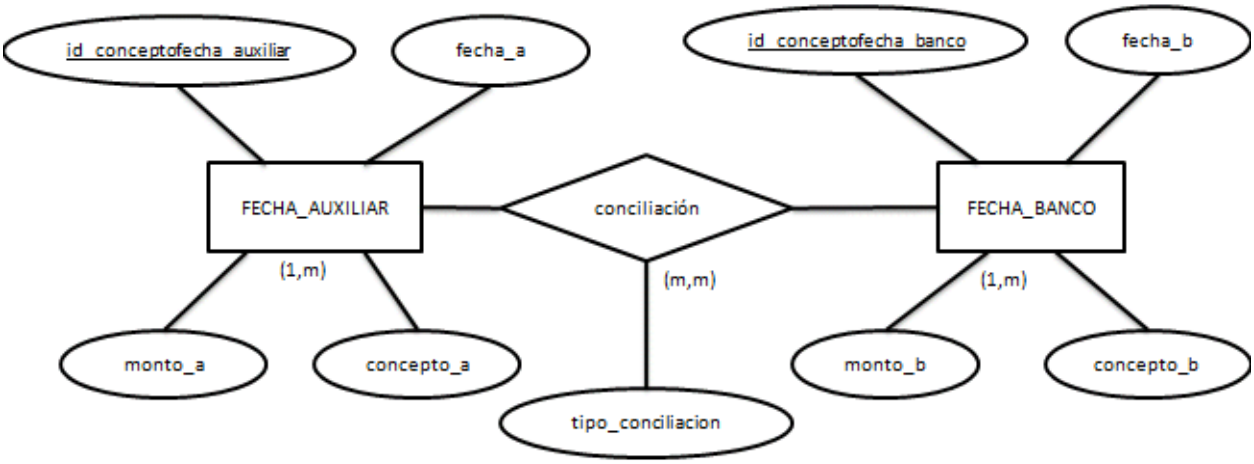


Imagen 4.0.1 Imagen que muestra las entidades fecha_auxiliar y fecha_banco junto con la relación "conciliación" después de normalizar.

El modelo relacional es el siguiente:

FECHA_AUXILIAR = {id_conceptofecha_auxiliar (PK), fecha_a, monto_a, concepto_a}

FECHA_BANCO = {id_conceptofecha_banco (PK), fecha_b, monto_b, concepto_b}

CONCILIACION = {[id_conceptofecha_auxiliar(FK),id_conceptofecha_banco(FK)](PK),

tipo_conciliacion}

Previo al diseño y desarrollo de la interfaz gráfica, se deciden desarrollar las clases funcionales en Java para lograr la conexión entre la base de datos y la aplicación. Es importante mencionar que en esta parte del desarrollo aún no existe interacción con el usuario a través de una interfaz gráfica, por lo tanto se comienza por visualizar el resultado de la conexión a la base de datos, así como las consultas a través de la consola de comandos del IDE (Integrated Development Environment. Entorno de Desarrollo Integrado por sus siglas en inglés) NetBeans; mientras que los datos se integran desde el código fuente tal como se puede apreciar en la imagen 4.0.2.

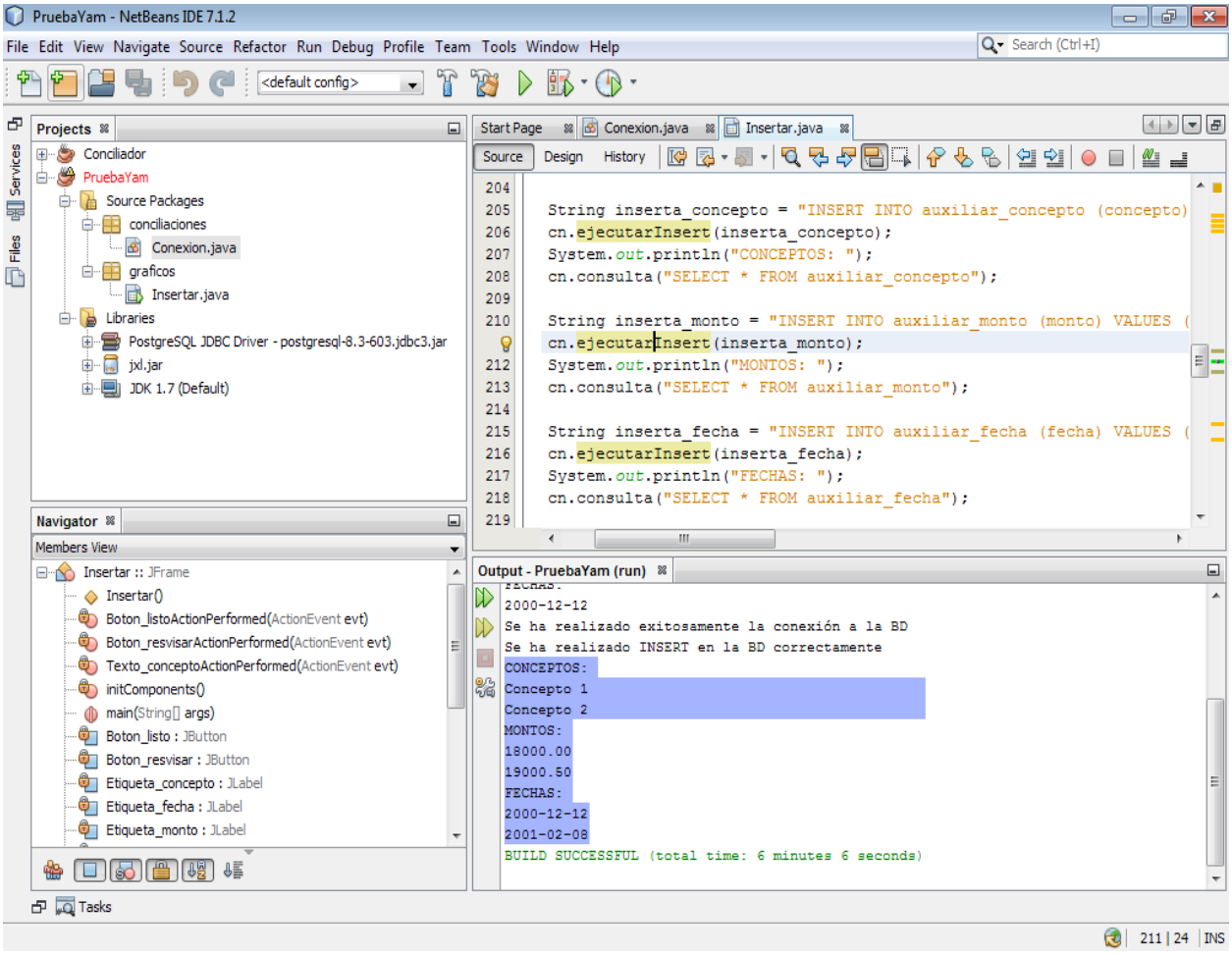


Imagen 4.0.2 Imagen que muestra la conexión a la base de datos de manera exitosa por medio del IDE NetBeans.

El siguiente paso consistió en adaptar lo antes mencionado de tal manera que existiera una interfaz gráfica que permita al usuario realizar estas actividades de manera interactiva; para ello el usuario podrá acceder a la aplicación a través de un ícono de acceso directo en el escritorio.

Una vez que el usuario inicia la aplicación, se desplegará un menú con el cual podrá interactuar y obtener los resultados esperados. A continuación mostramos el procedimiento.

En la imagen 4.0.3 se muestra cómo el usuario inicialmente puede insertar dato por dato correspondiente al libro auxiliar de bancos. Basta con escribir en el cuadro de texto los datos correspondientes a Concepto, Monto y Fecha y guardar la información usando el botón “Insertar”.

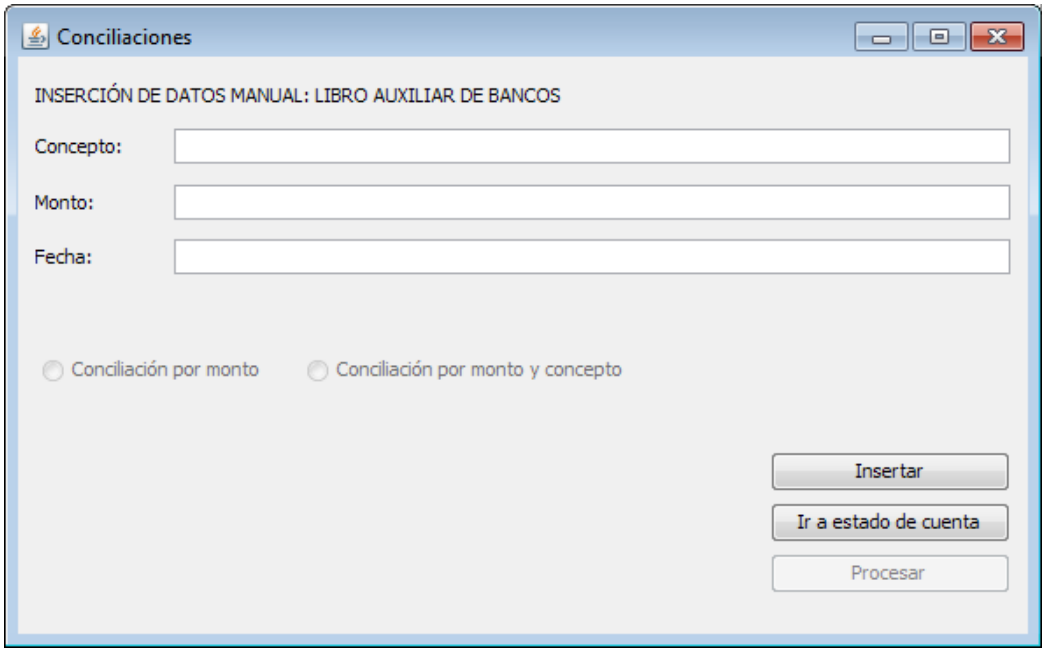


Imagen 4.0.3 Imagen correspondiente a los datos del libro auxiliar de bancos.

Nótese que el botón de “Procesar” no se encuentra habilitado debido a que aún no se cuenta con información en el estado de cuenta. A través del botón “Ir a estado de cuenta”, el usuario puede comenzar a insertar los datos correspondientes.

Estando en la ventana del estado de cuenta el usuario puede registrar la información correspondiente realizando los mismos pasos que se hicieron en la ventana del libro auxiliar de bancos.

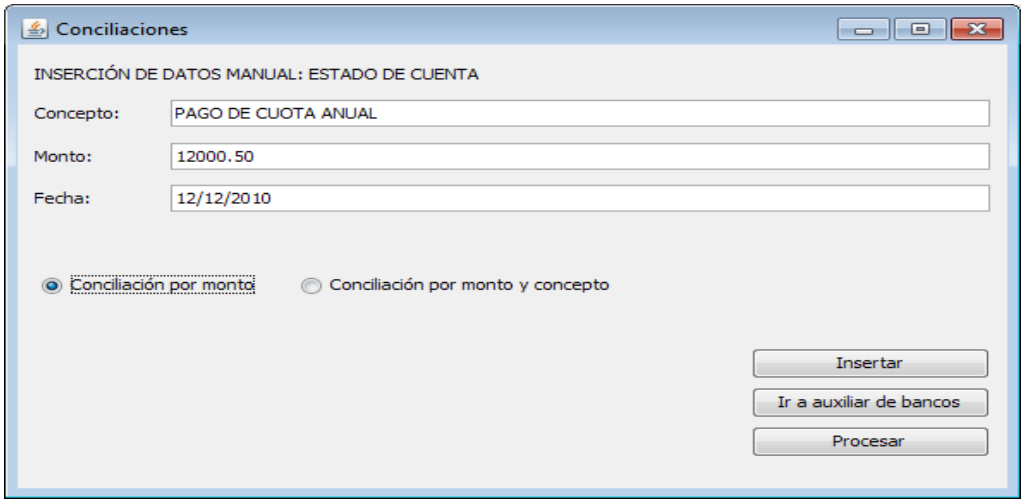


Imagen 4.0.4 Imagen que muestra los campos correspondientes al estado de cuenta.

Hasta que existan valores dentro de la base de datos tanto del libro auxiliar como del estado de cuenta, se activarán los radiobuttons que permitirán realizar el procesamiento de la información por monto o por monto y concepto; la decisión dependerá del usuario y cuando se decida que método ejecutar deberá oprimir el botón “Procesar” para obtener los resultados.

Al continuar con el desarrollo de la aplicación, tomando en cuenta la opinión del cliente y con base en observaciones puntuales acerca del tiempo demorado en la carga de datos, se determina que existe la gran posibilidad de que el usuario cometa errores al insertar dato por dato. Por lo tanto se decidió agilizar el proceso de carga de información, integrando la opción de utilizar los archivos de Excel como fuente inicial de datos, siendo esta una herramienta de uso cotidiano para los usuarios.

Es así como se decide añadir al bloque anterior una nueva sección que le brinde al usuario final la posibilidad de elegir entre cargar la información registro por registro o de manera masiva a través del archivo en Excel.

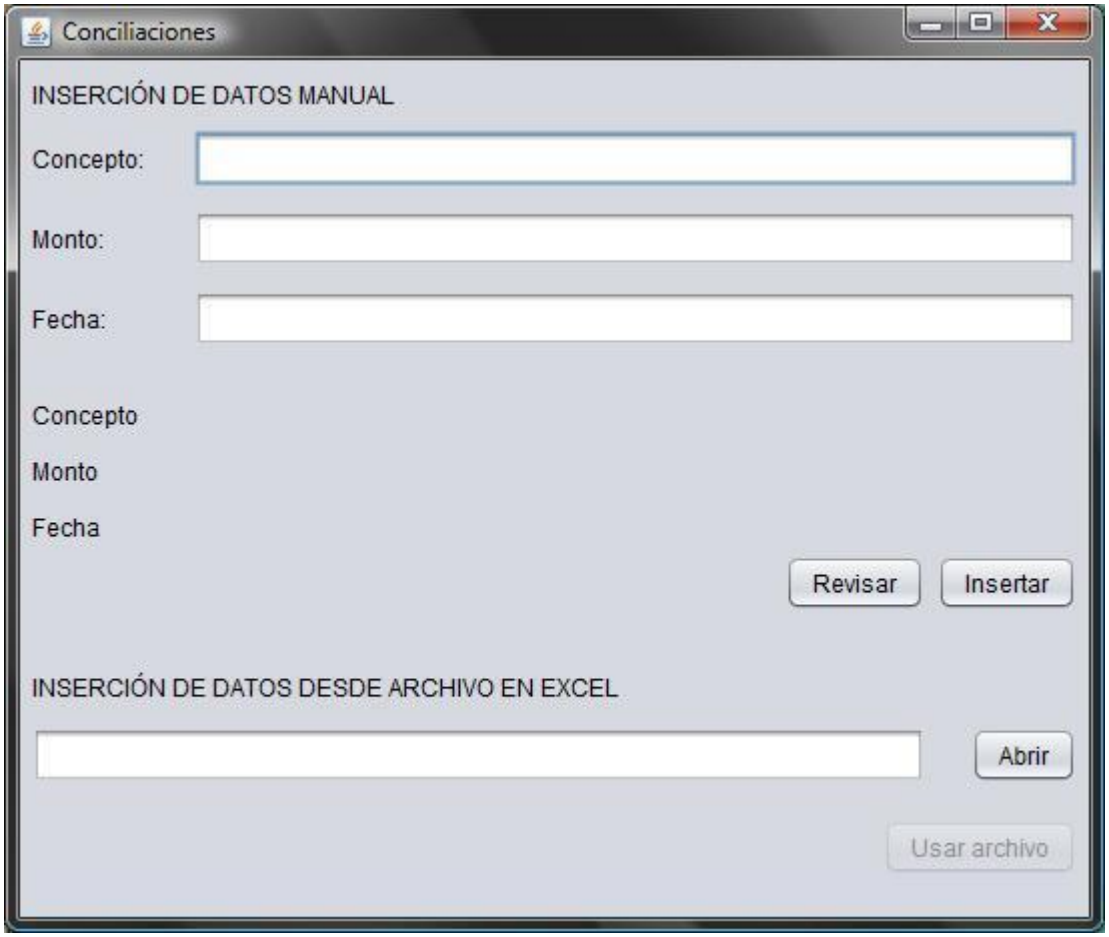


Imagen 4.0.5 Imagen que muestra ambas opciones para ingresar datos tanto manual como por medio de un archivo de Excel.

El procedimiento para insertar registro por registro sigue siendo el mismo que el anterior, mientras que para insertar datos de manera masiva se abre automáticamente una nueva ventana como la que se muestra en la imagen 4.0.6

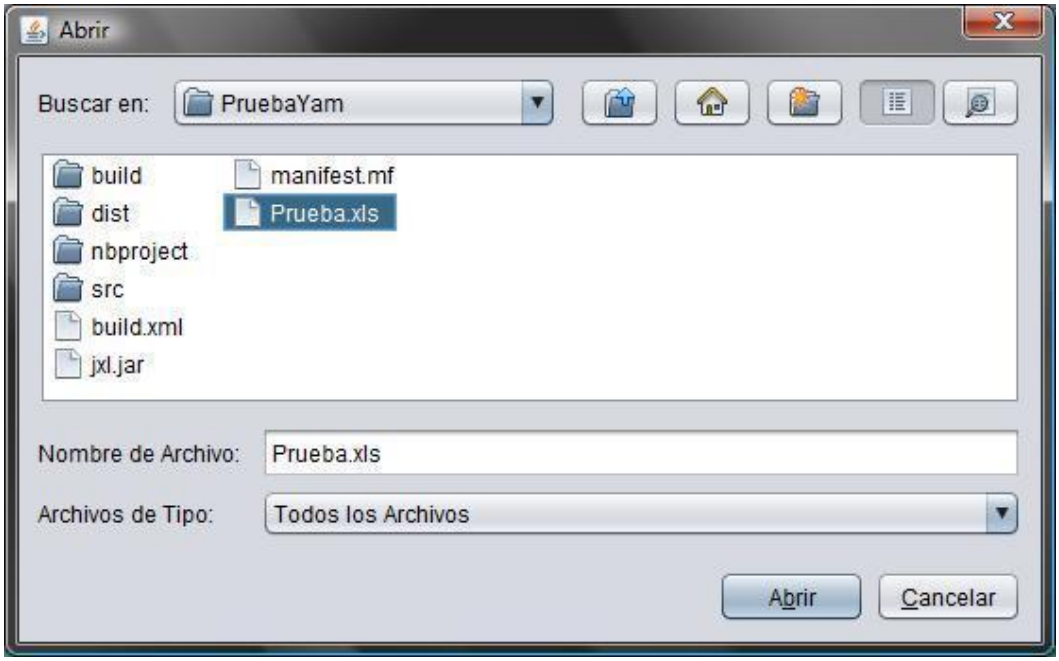


Imagen 4.0.6 Imagen en la que se muestra la interacción con el usuario para seleccionar el archivo a procesar.

Conforme se adaptaba la nueva interfaz gráfica al código que ya se tenía desarrollado, en ese momento, se realizaban pruebas para que la funcionalidad de dicha interfaz afectara lo menos posible a lo ya desarrollado.

Como se puede apreciar en la imagen 4.0.7, se ajusta la aplicación de tal manera que el usuario ya tiene la opción de realizar la carga de la información desde un archivo de Excel; durante su diseño se consideró que se debería de mostrar la ruta de origen de éste para que el usuario corrobore que ha sido seleccionado correctamente.

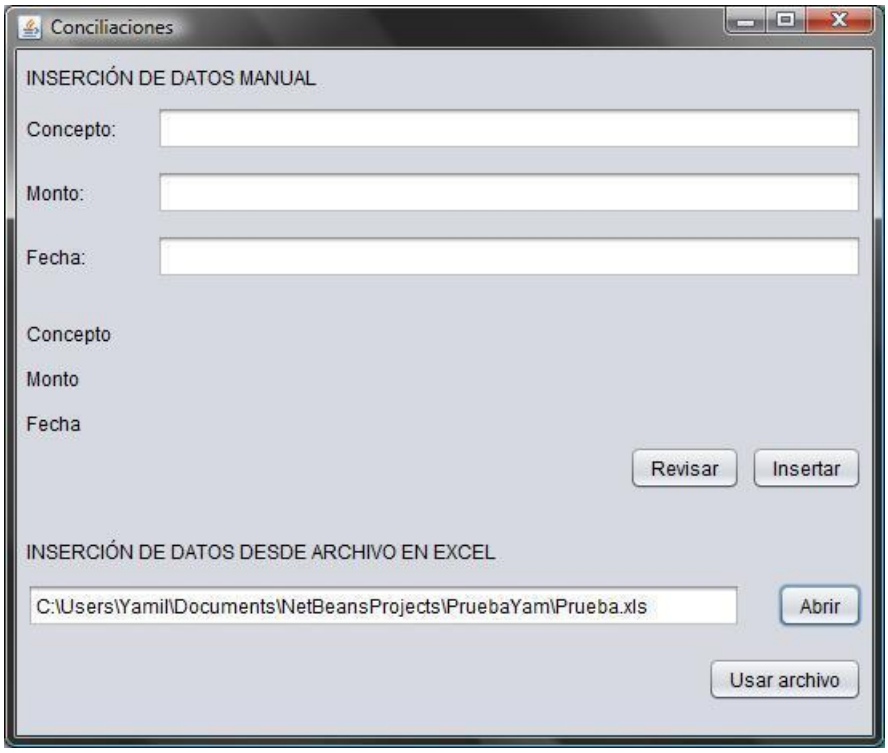


Imagen 4.0.7 Imagen en la que se muestra la ruta del archivo Excel seleccionado para corroborar que sea la correcta.

En la imagen 4.0.8 se puede apreciar la versión final en la cual se observa de manera general las opciones que el usuario puede ejecutar, las cajas de texto son para detallar el reporte final que será presentado al usuario:

- La razón social: identificará a la empresa que está realizando la conciliación bancaria.
- La fecha: identifica cuando es realizada la conciliación.
- El banco y cuenta: es para identificar la cuenta registrada en el banco contra la cual se realiza la conciliación.

En esta versión se puede apreciar un botón que realiza un cambio en la ubicación del archivo a procesar, en caso de que la ruta sea incorrecta, también se añadió un botón para comenzar a procesar los datos, una barra que indica el progreso de avance del proceso y una vez que el proceso es finalizado se habilita un botón para ver resultados, el cual muestra el reporte final.

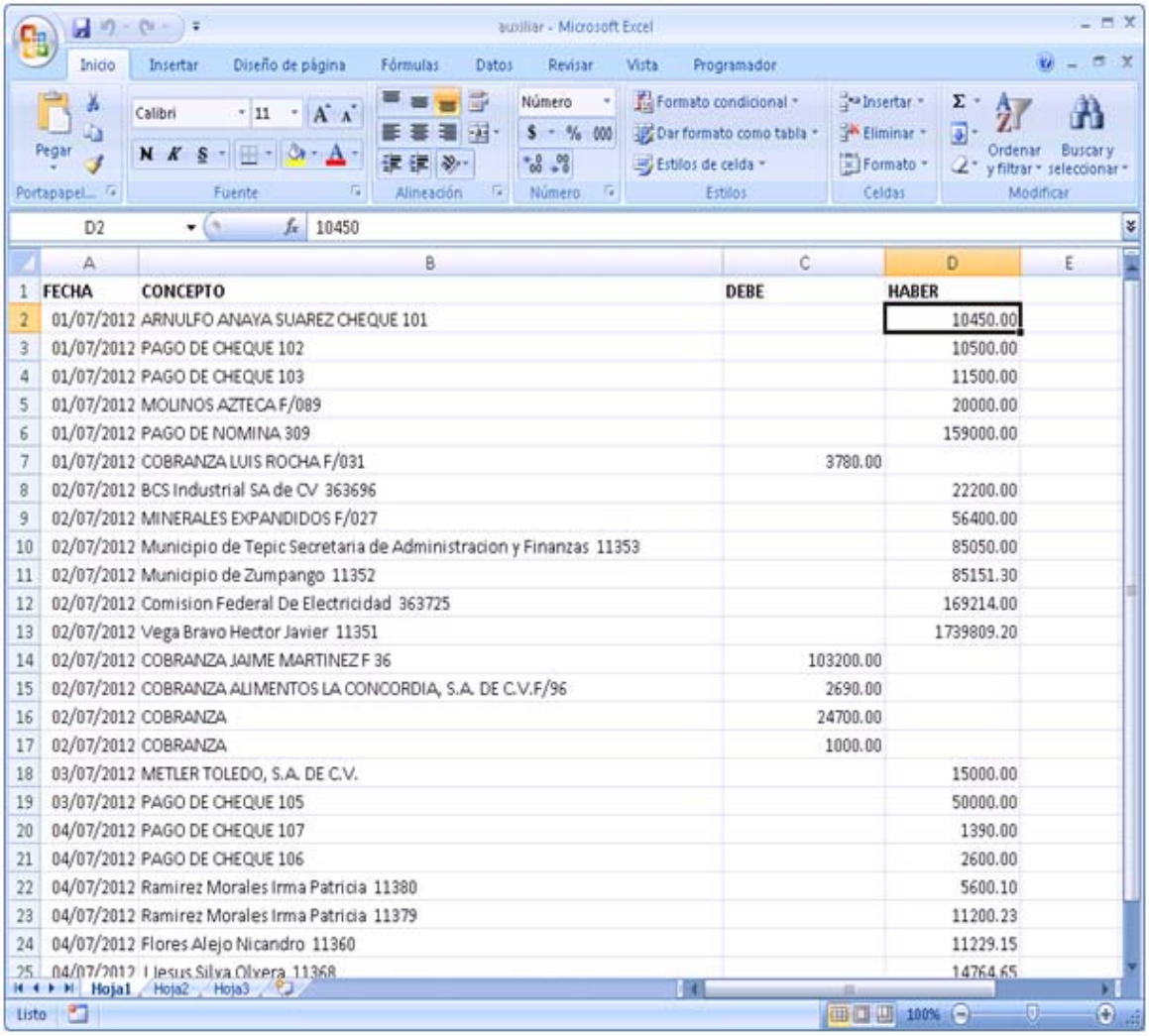
Imagen 4.0.8 Imagen que muestra la versión final de la interfaz gráfica creada.

Con base a los últimos ajustes realizados en el proyecto se determinó lo siguiente:

Antes de iniciar la aplicación el usuario debe tener preparados los dos archivos de Excel que contendrán los datos que van a ser cargados al sistema.

Un archivo de Excel llevará en la primera hoja la información del libro auxiliar de bancos el cual forzosamente debe llevar el siguiente formato:

Columna hoja de datos	Información
A	Fecha (dd/mm/yyyy)
B	Concepto
C	Debe (monto)
D	Haber (monto)

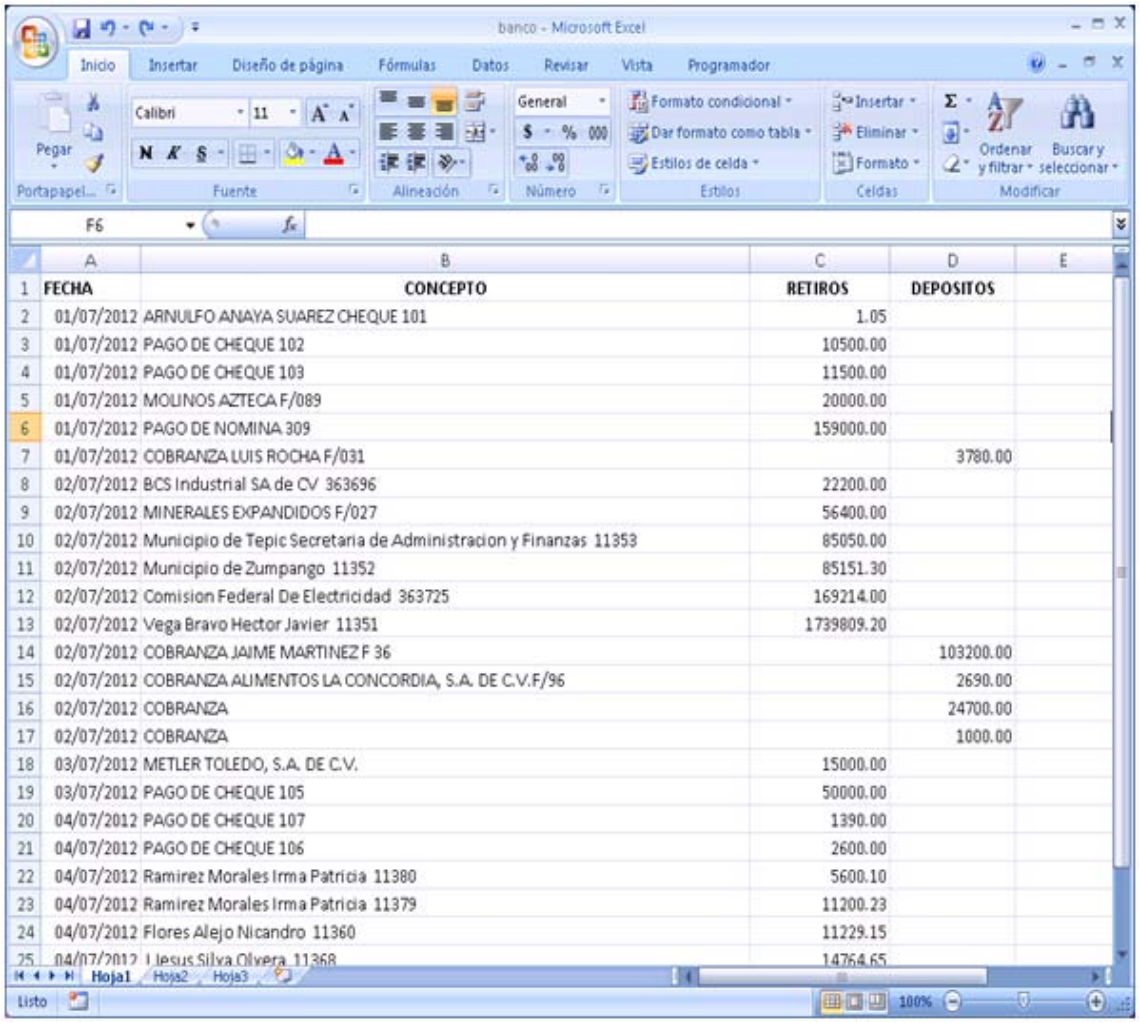


	A	B	C	D	E
1	FECHA	CONCEPTO	DEBE	HABER	
2	01/07/2012	ARNULFO ANAYA SUAREZ CHEQUE 101		10450.00	
3	01/07/2012	PAGO DE CHEQUE 102		10500.00	
4	01/07/2012	PAGO DE CHEQUE 103		11500.00	
5	01/07/2012	MOLINOS AZTECA F/089		20000.00	
6	01/07/2012	PAGO DE NOMINA 309		159000.00	
7	01/07/2012	COBRANZA LUIS ROCHA F/031	3780.00		
8	02/07/2012	BCS Industrial SA de CV 363696		22200.00	
9	02/07/2012	MINERALES EXPANDIDOS F/027		56400.00	
10	02/07/2012	Municipio de Tepic Secretaria de Administracion y Finanzas 11353		85050.00	
11	02/07/2012	Municipio de Zumpango 11352		85151.30	
12	02/07/2012	Comision Federal De Electricidad 363725		169214.00	
13	02/07/2012	Vega Bravo Hector Javier 11351		1739809.20	
14	02/07/2012	COBRANZA JAIME MARTINEZ F 36	103200.00		
15	02/07/2012	COBRANZA ALIMENTOS LA CONCORDIA, S.A. DE C.V.F/96	2690.00		
16	02/07/2012	COBRANZA	24700.00		
17	02/07/2012	COBRANZA	1000.00		
18	03/07/2012	METLER TOLEDO, S.A. DE C.V.		15000.00	
19	03/07/2012	PAGO DE CHEQUE 105		50000.00	
20	04/07/2012	PAGO DE CHEQUE 107		1390.00	
21	04/07/2012	PAGO DE CHEQUE 106		2600.00	
22	04/07/2012	Ramirez Morales Irma Patricia 11380		5600.10	
23	04/07/2012	Ramirez Morales Irma Patricia 11379		11200.23	
24	04/07/2012	Flores Alejo Nicandro 11360		11229.15	
25	04/07/2012	Jesus Silva Olvera 11368		14764.65	

Imagen 4.0.9 Imagen que muestra el formato que debe tener el archivo auxiliar de bancos.

El otro archivo de Excel debe contener la información del estado de cuenta y este debe seguir el siguiente formato.

Columna hoja de datos	Información
A	Fecha (dd/mm/yyyy)
B	Concepto
C	Retiros (monto)
D	Depósitos (monto)



banco - Microsoft Excel

	A	B	C	D	E
1	FECHA	CONCEPTO	RETIROS	DEPOSITOS	
2	01/07/2012	ARNULFO ANAYA SUAREZ CHEQUE 101	1.05		
3	01/07/2012	PAGO DE CHEQUE 102	10500.00		
4	01/07/2012	PAGO DE CHEQUE 103	11500.00		
5	01/07/2012	MOLINOS AZTECA F/089	20000.00		
6	01/07/2012	PAGO DE NOMINA 309	159000.00		
7	01/07/2012	COBRANZA LUIS ROCHA F/031		3780.00	
8	02/07/2012	BCS Industrial SA de CV 363696	22200.00		
9	02/07/2012	MINERALES EXPANDIDOS F/027	56400.00		
10	02/07/2012	Municipio de Tepic Secretaria de Administracion y Finanzas 11353	85050.00		
11	02/07/2012	Municipio de Zumpango 11352	85151.30		
12	02/07/2012	Comision Federal De Electricidad 363725	169214.00		
13	02/07/2012	Vega Bravo Hector Javier 11351	1739809.20		
14	02/07/2012	COBRANZA JAIME MARTINEZ F 36		103200.00	
15	02/07/2012	COBRANZA ALIMENTOS LA CONCORDIA, S.A. DE C.V.F/96		2690.00	
16	02/07/2012	COBRANZA		24700.00	
17	02/07/2012	COBRANZA		1000.00	
18	03/07/2012	METLER TOLEDO, S.A. DE C.V.	15000.00		
19	03/07/2012	PAGO DE CHEQUE 105	50000.00		
20	04/07/2012	PAGO DE CHEQUE 107	1390.00		
21	04/07/2012	PAGO DE CHEQUE 106	2600.00		
22	04/07/2012	Ramirez Morales Irma Patricia 11380	5600.10		
23	04/07/2012	Ramirez Morales Irma Patricia 11379	11200.23		
24	04/07/2012	Flores Alejo Nicandro 11360	11229.15		
25	04/07/2012	Jesus Silva Olvera 11368	14764.65		

Imagen 4.0.10 Imagen que muestra el formato que debe tener el archivo estado de cuenta.

Cuando el usuario inicia la aplicación, se despliega una ventana en la cual se escoge la ubicación (carpeta) en donde se creará el archivo final con los resultados obtenidos tal y como se muestra en la siguiente imagen:

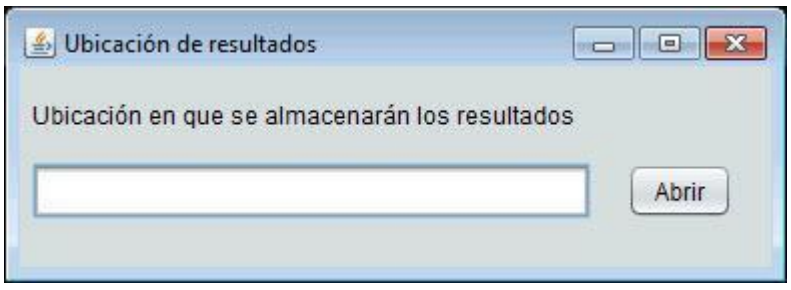


Imagen 4.0.11 Imagen en la que se muestra la ventana inicial de la aplicación.

Al iniciar la aplicación internamente se realiza un borrado de los registros de todas las tablas utilizadas durante el proceso de la conciliación, para no tener datos que fueron utilizados en conciliaciones anteriores.

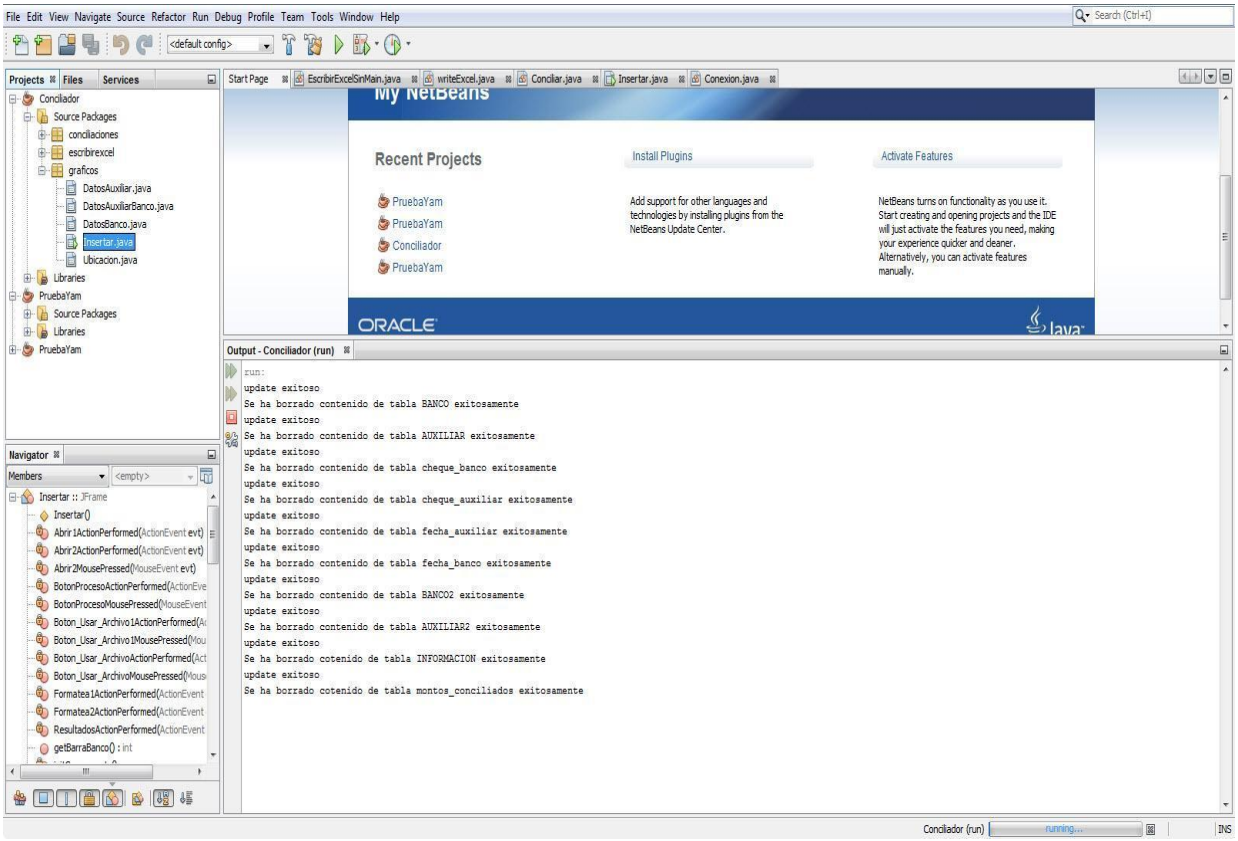


Imagen 4.0.12 Imagen que muestra la depuración de las tablas una vez que se abre la ventana de ubicación de resultados.

Al oprimir el botón abrir se puede seleccionar la ruta deseada en donde se indicará la ubicación final, este menú permite acceder entre los directorios existentes o incluso nuevos directorios creados por el usuario.

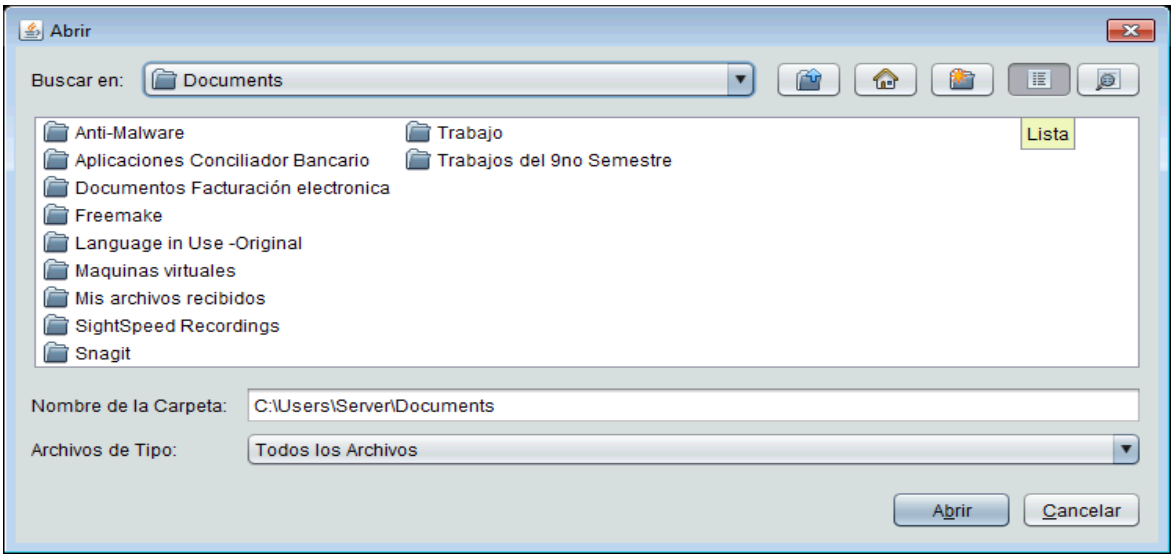


Imagen 4.0.13 Imagen que muestra la ventana para seleccionar la ruta donde se creará el archivo con los resultados de la conciliación.

Después de seleccionar la ruta y como se mencionó anteriormente el usuario podrá visualizar la ventana final Imagen 4.0.14 en la cual debe seleccionar los archivos en el formato indicado.

En el momento en el que el usuario de un clic en el botón abrir que se encuentra habilitado se desplegará una ventana en donde podrá indicar a la aplicación el directorio y el nombre en donde se encuentra el archivo estado de cuenta.

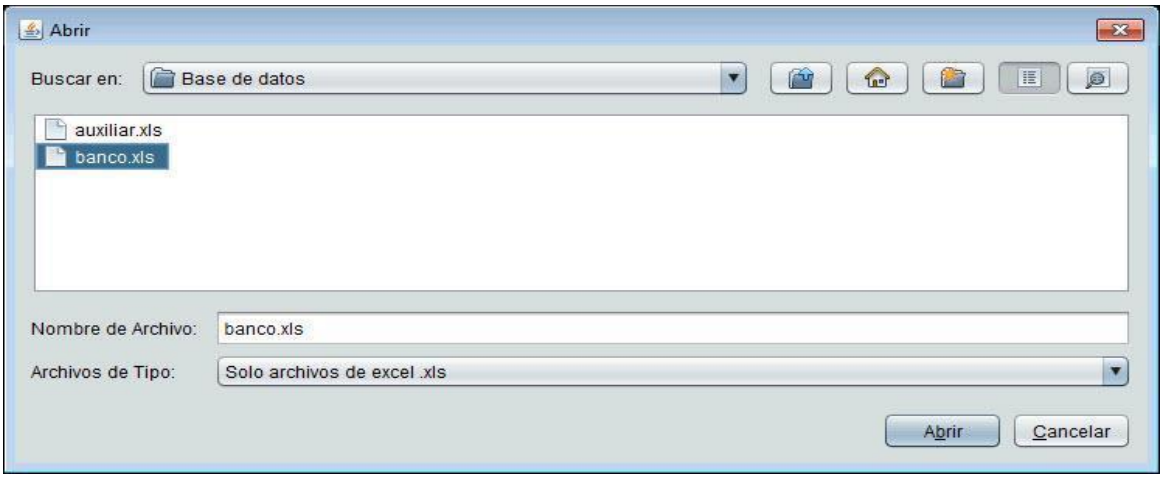


Imagen 4.0.14 Imagen que representa la acción de seleccionar el estado de cuenta.

Después de seleccionar el archivo se habilita el botón “Cargar datos” con el cual se almacenará la información en la base de datos.

Para que la ruta no pudiera ser modificada desde la caja de texto, se tuvo que bloquear la edición, en caso de querer cambiar la ruta el usuario debe cargar los datos para habilitar la opción de “Cambiar ubicación de archivo”.

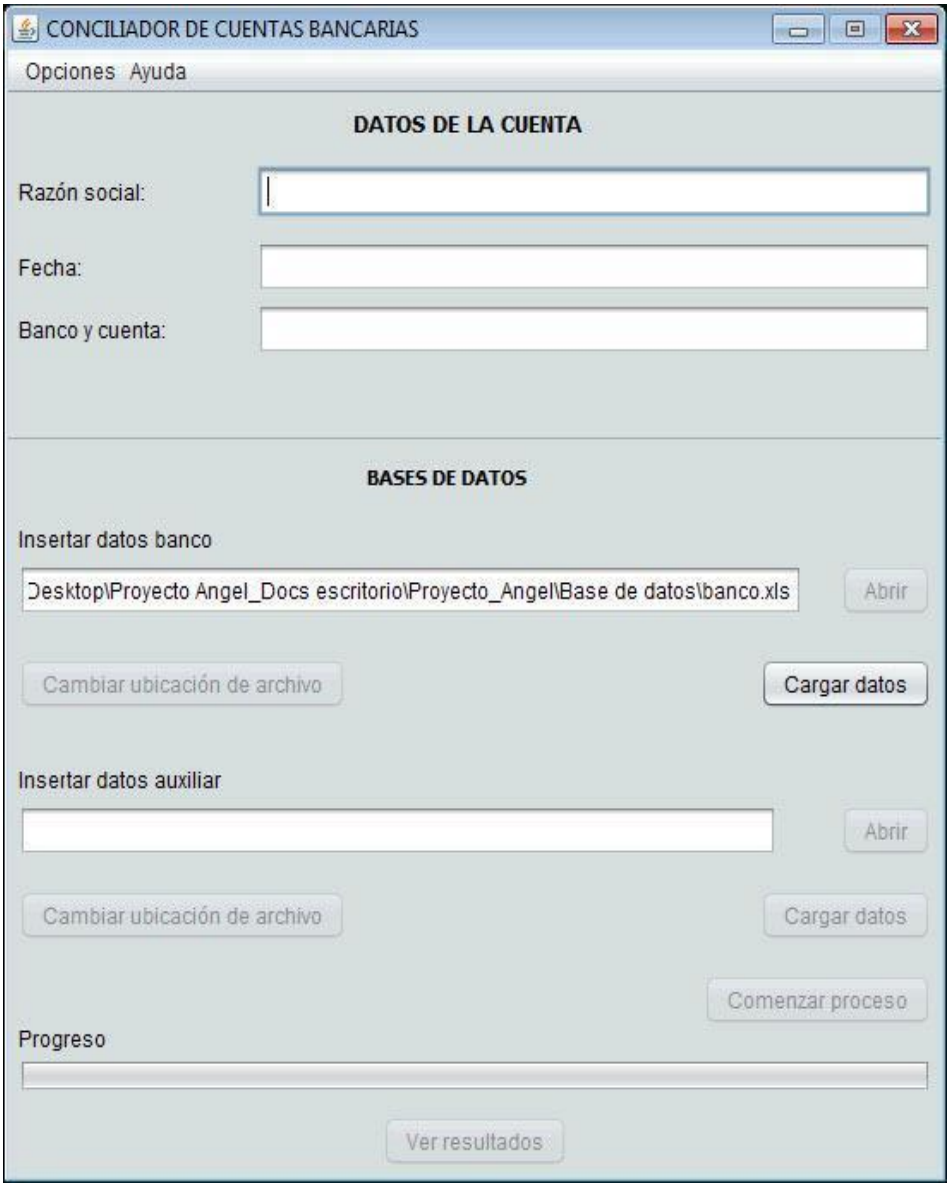


Imagen 4.0.15 Imagen en la que se muestra la ruta seleccionada teniendo habilitado el botón cargar datos.

Una vez que el usuario oprima el botón cargar datos se realiza una carga inicial del archivo Excel a la base de datos interna que la aplicación estará consultando a lo largo del proceso. Además la barra de progreso incrementará mostrando el porcentaje restante para que se muestre el reporte final.

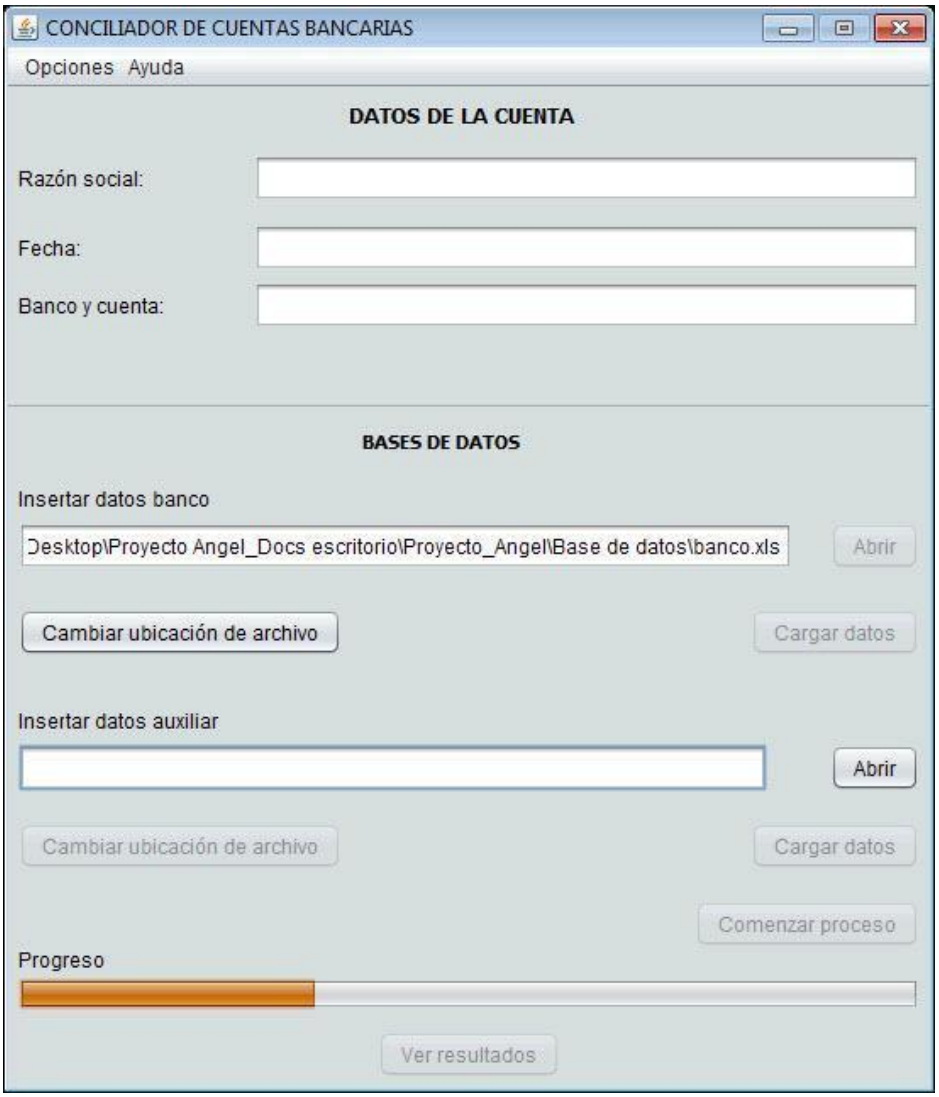


Imagen 4.0.16 Imagen que muestra la carga inicial del estado de cuenta representado en la barra de progreso.

Si el archivo cargado fue uno incorrecto se utiliza la opción “Cambiar ubicación de archivo”, esta opción además de corregir la barra de progreso a un estado inicial borra el contenido de las tablas pobladas con los datos del archivo Excel cargados anteriormente.

Imagen que muestra la acción del botón “Cambiar ubicación de archivo” en la consola de NetBeans, esta fue una de las pruebas realizadas mientras se integraba la interfaz gráfica al código para evaluar que las funciones realizarán lo que se esperaba.

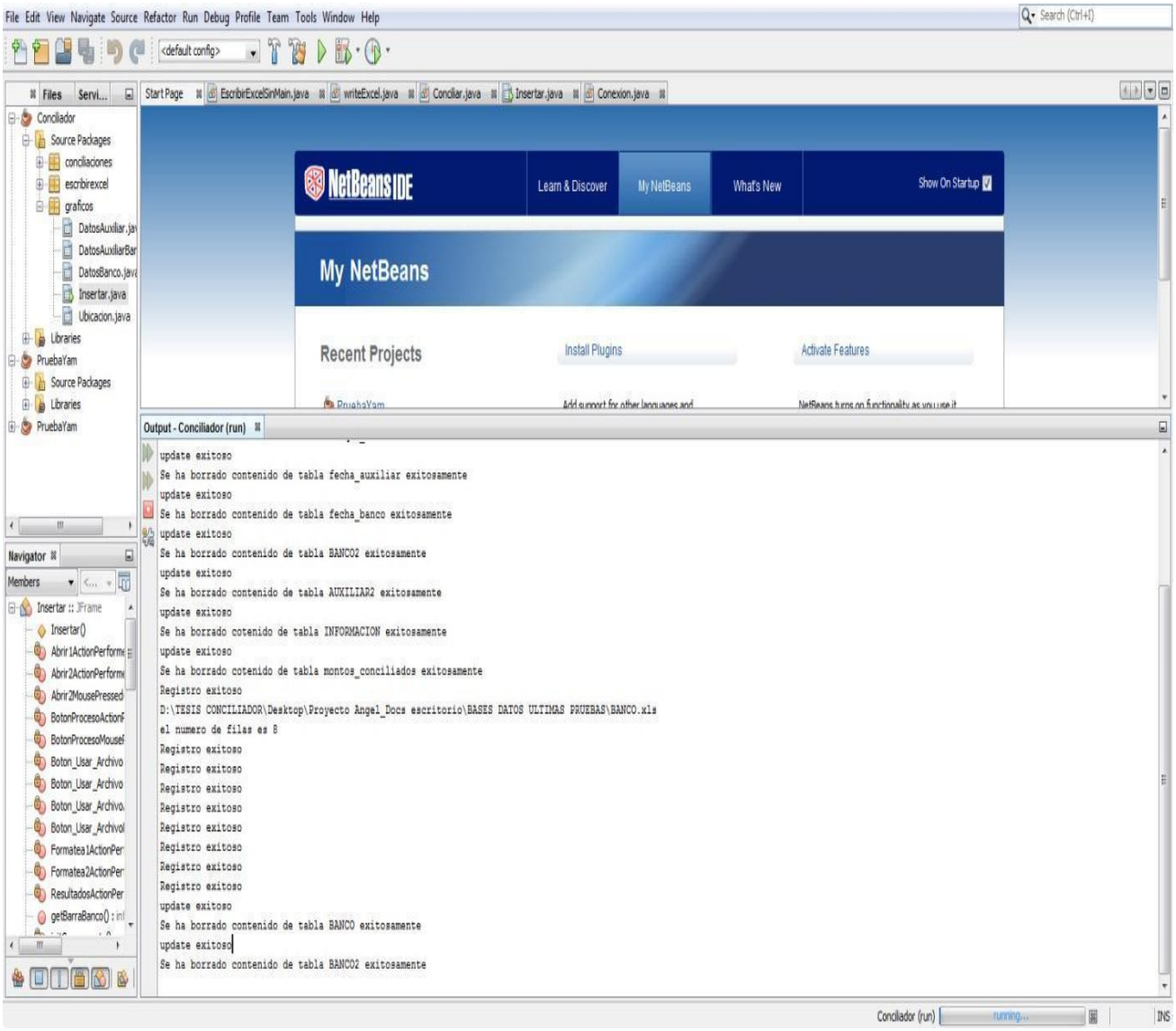


Imagen 4.0.17 Imagen que muestra la depuración de las tablas pobladas anteriormente.

Imagen que muestra la acción gráfica del botón “Cambiar ubicación de archivo” en la cual se puede apreciar que el botón “Abrir” se habilita para seleccionar nuevamente el archivo correcto y una vez seleccionado se volverá a habilitar únicamente el botón “Cargar datos”.

Imagen 4.0.18 Imagen en la que se aprecia el botón abrir habilitado después de haber utilizado el botón Cambiar ubicación de archivo.

Una vez cargado el primer archivo el usuario podrá guardar los datos del libro que contenga la información del auxiliar de bancos así que dará clic en la función “Abrir”.

Se desplegará una ventana en donde el usuario indica a la aplicación el directorio y el nombre en donde se encuentra el archivo a cargar.

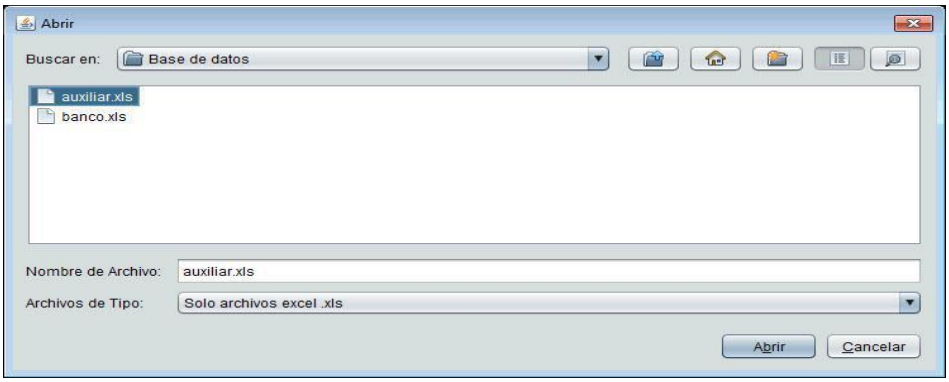


Imagen 4.0.19 Imagen que representa la acción de seleccionar el archivo auxiliar de bancos.

Después de seleccionar el archivo el usuario podrá cargar los datos con el botón “Cargar datos”.

Una vez cargado el segundo archivo la barra de proceso incrementará y el botón “Cambiar ubicación de archivo” y “Comenzar proceso” se habilitarán, también el usuario deberá llenar los campos “DATOS DE LA CUENTA” para que aparezcan en el reporte final como se aprecia en la siguiente imagen.

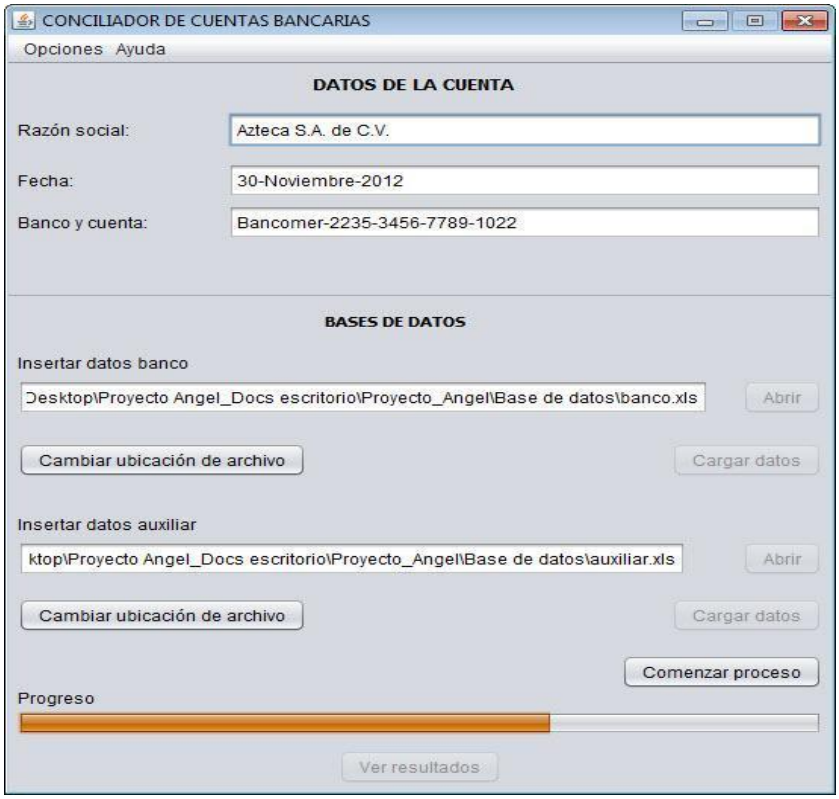


Imagen 4.0.20 Imagen que muestra la carga inicial de ambos archivos y el botón comenzar proceso habilitado.

Cuando se realiza la carga de datos sin importar que únicamente se ingrese el primer libro, el usuario puede validar por medio de una ventana gráfica los datos que fueron cargados, para ello se puede usar el menú “Opciones” y después seleccionar la opción que el usuario desee.

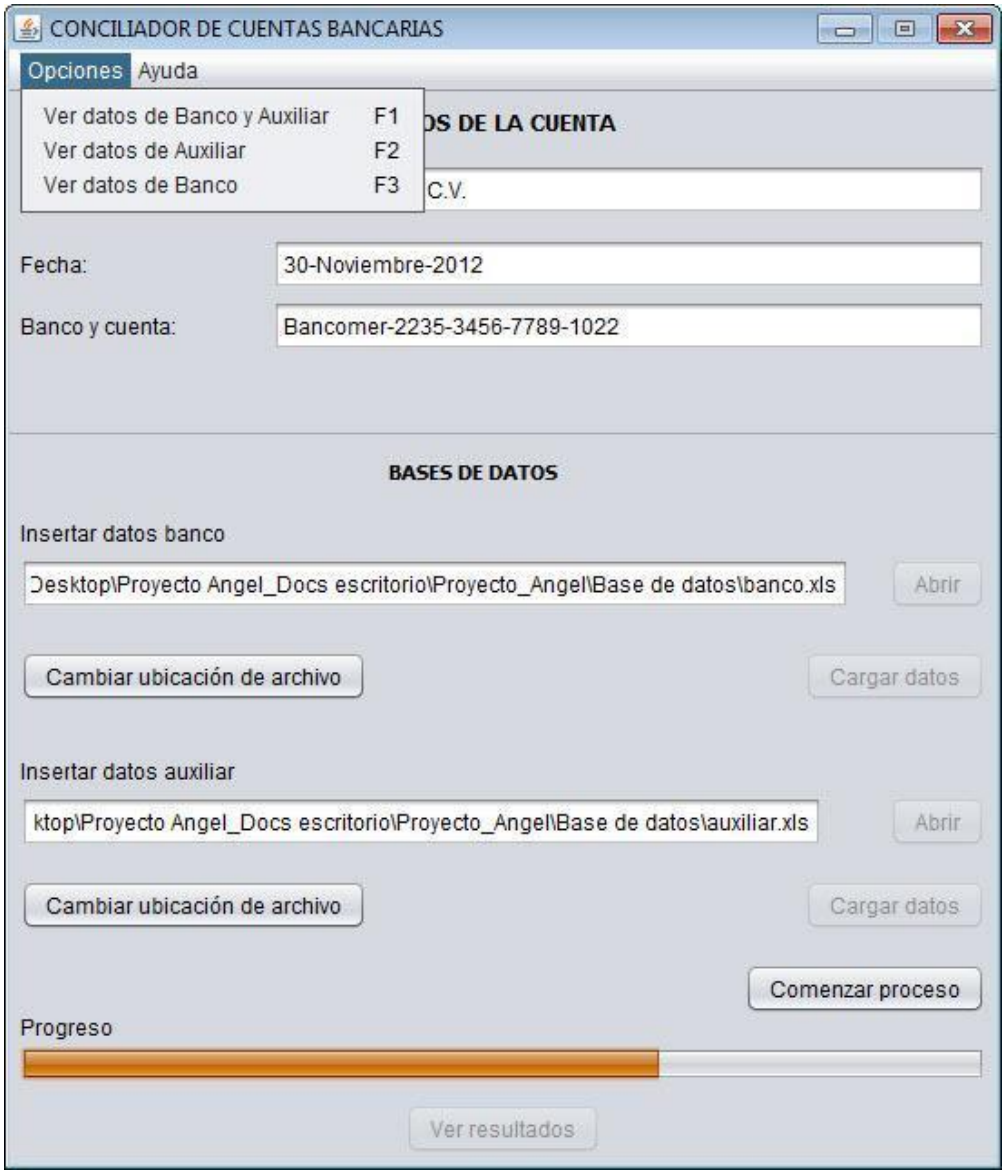


Imagen 4.0.21 Imagen que muestra el menú opciones, por medio del cual se pueden visualizar los datos almacenados.

En la imagen 4.0.22 se muestran los datos del libro auxiliar de bancos y estado de cuenta una vez que han sido cargados a la base de datos, el usuario puede visualizar la información recorriendo la tabla completa por medio de la interfaz.

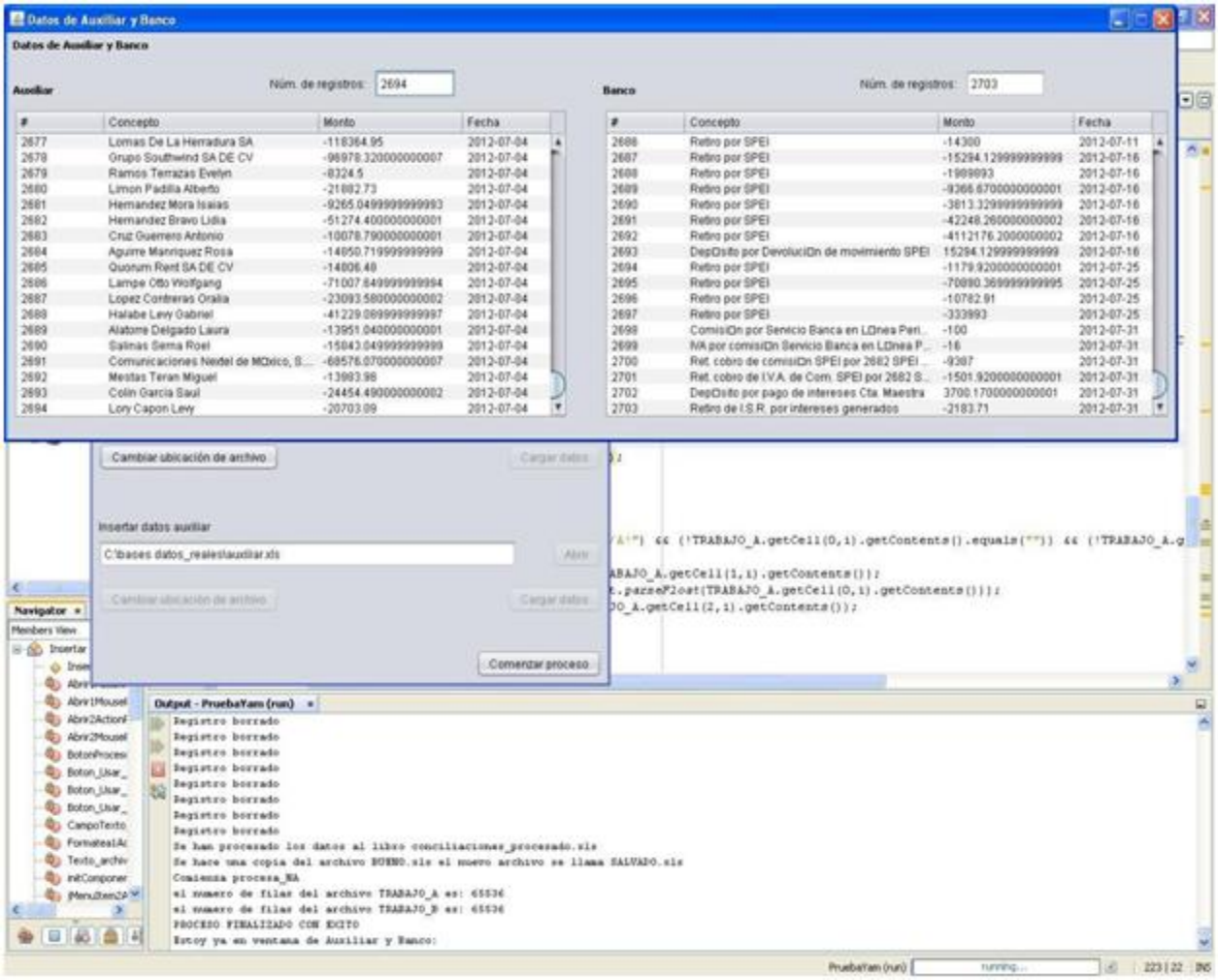


Imagen 4.0.22 Imagen que muestra ambos libros almacenados en la base de datos por medio de la interfaz gráfica.

En los primeros diseños de las ventanas para visualizar los datos cargados se realizaron diferentes modificaciones, de modo que se fueron ajustando conforme avanzó el proyecto, ya que nos dimos cuenta que en algunas pantallas de computadoras más pequeñas las ventanas no se podían manipular bien, lo cual dificultaba la visualización de los datos y en algunos casos era casi imposible cerrar la ventana para continuar con el proceso.

En la imagen 4.0.23 se muestran los datos del libro auxiliar una vez que han sido cargados también se visualiza el campo número de registros el cual indica la cantidad de registros almacenados para que el usuario vea si realmente se realizó la carga completa.

Imagen 4.0.23 Imagen que muestra los datos de la tabla auxiliar para la versión final de la aplicación.

La imagen 4.0.24 muestra los datos del libro estado de cuenta una vez que han sido cargados, junto con el campo número de registros el cual indica al usuario si realmente se realizó la carga completa.

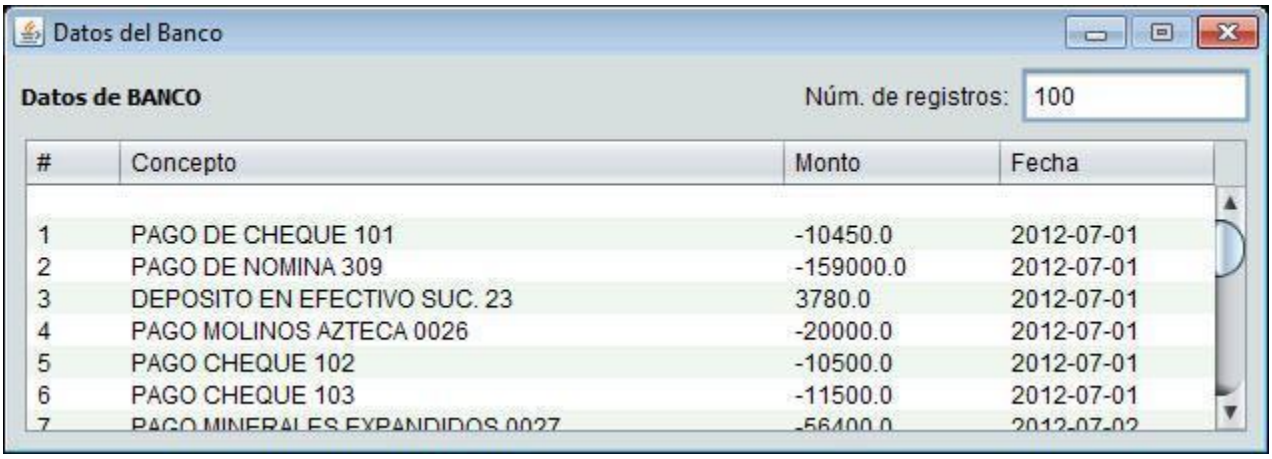


Imagen 4.0.24 Imagen que muestra los datos de la tabla banco para la versión final de la aplicación.

Como se puede ver se realizaron ajustes a la interfaz para que el usuario pudiera visualizar los datos de una mejor manera, al final ambas tablas son mostradas de manera horizontal.

En la imagen 4.0.25 se muestran los datos de ambas tablas visualizadas con la versión final de la aplicación.

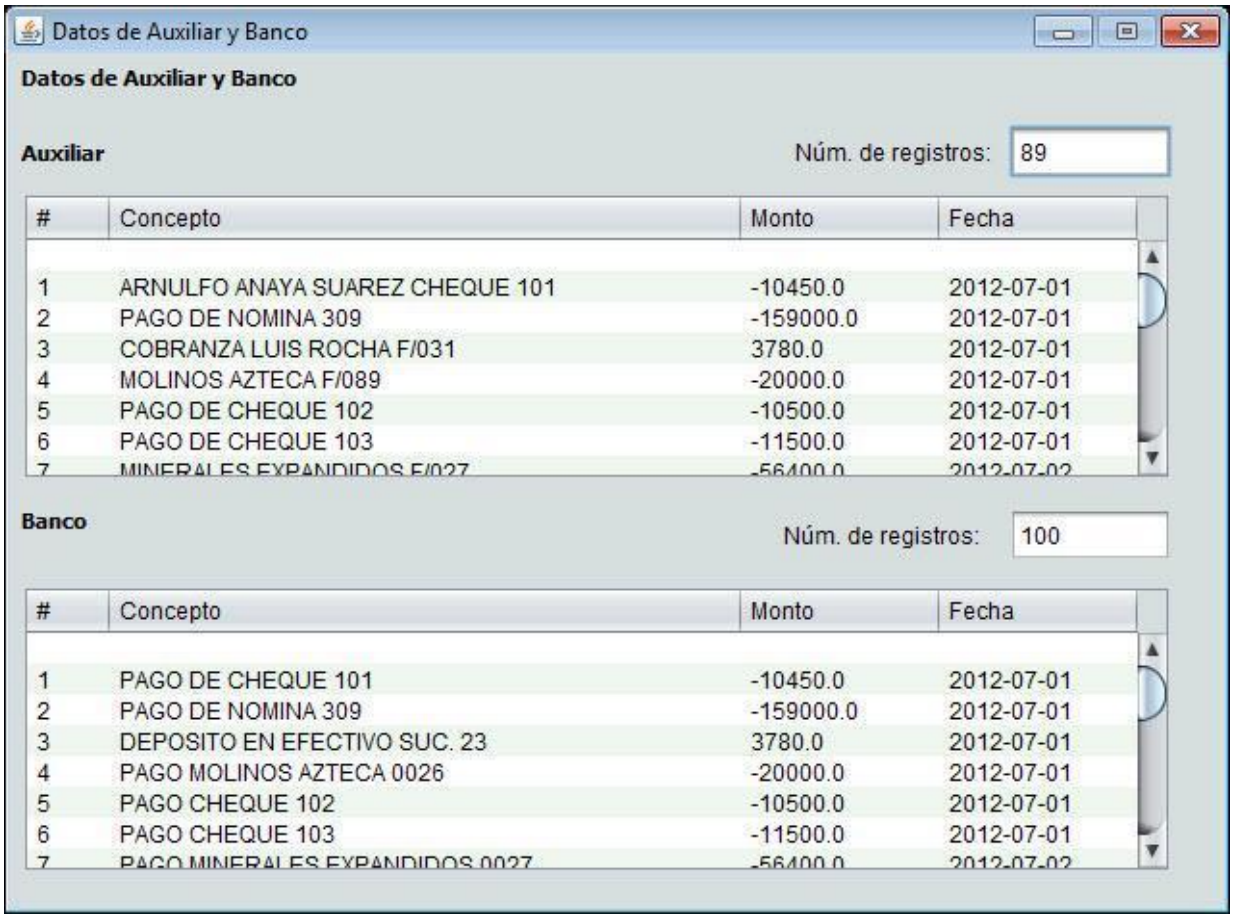


Imagen 4.0.25 Imagen que muestra la versión final donde se pueden apreciar ambos archivos cargados a la base de datos junto con el campo número de registros.

La interfaz también tiene un menú de ayuda en el cual se muestra una guía paso a paso de la interacción con la aplicación, la cual le puede servir al usuario para familiarizarse con el software.

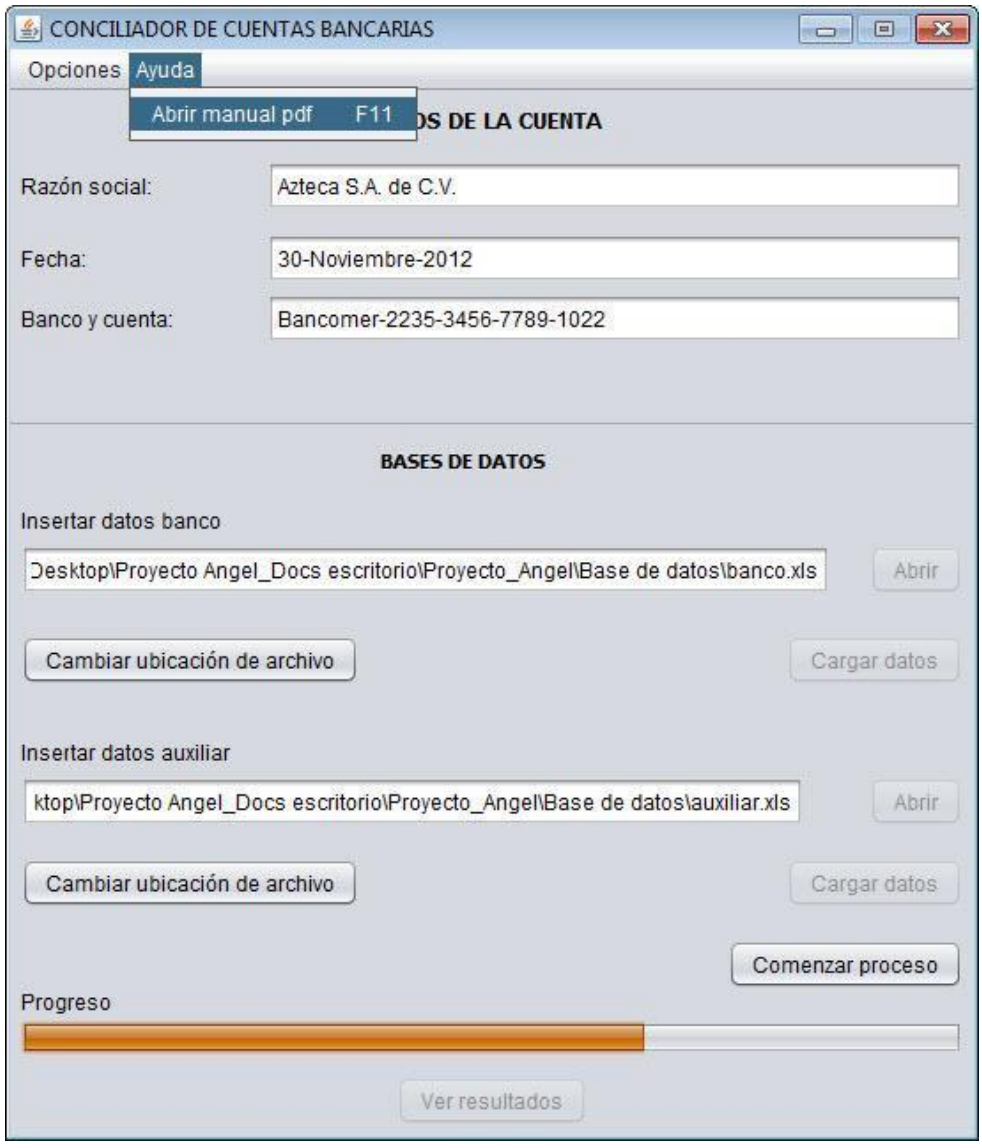


Imagen 4.0.26 Imagen que muestra la opción para consultar el manual pdf.

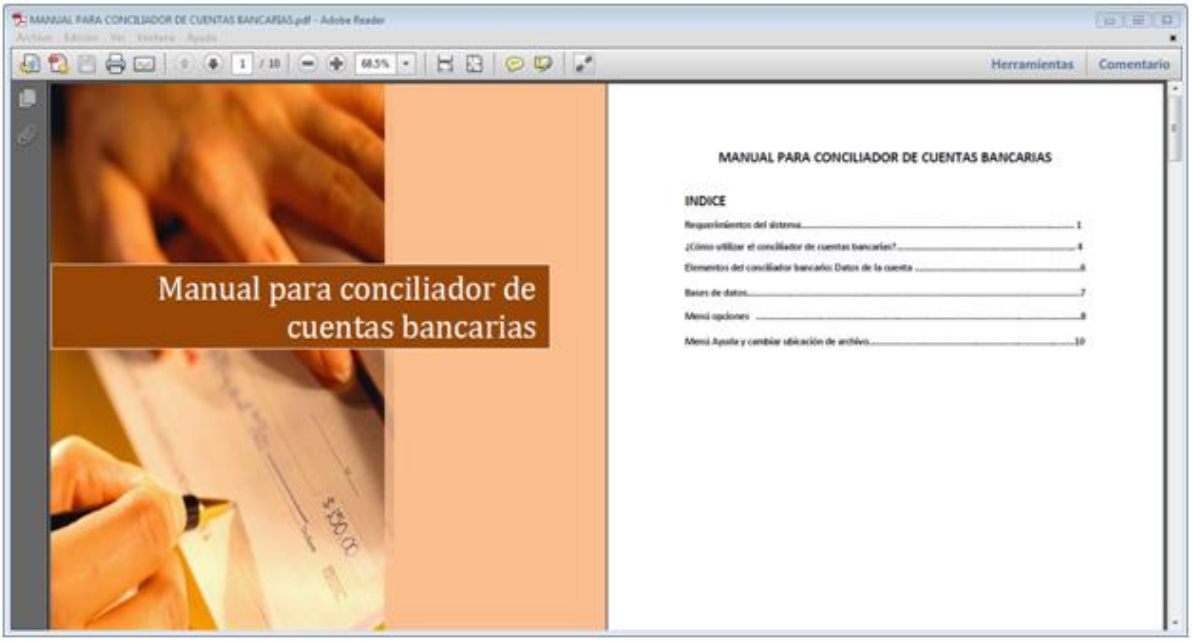


Imagen 4.0.27 Imagen que muestra el manual de conciliador bancario con su correspondiente índice.

Tras cargar los dos archivos el usuario usará el botón “Comenzar proceso” que ejecutará toda la lógica interna pasando por los tres métodos y por una serie de scripts para poder presentar el resultado final en un libro de Excel.

En la imagen 4.0.28 se puede apreciar cuando el proceso ha sido finalizado debido a que se habilita el botón “Ver resultados” para poder visualizar el reporte final.

Imagen 4.0.28 Imagen que muestra la barra de progreso al 100% cuando la conciliación se ha completado exitosamente.

Una vez que el usuario oprime el botón ver resultados el archivo es mostrado para que él pueda revisarlo a detalle.
En la imagen 4.0.29 se muestra el archivo con el reporte final después de haber procesado los datos por los tres métodos.

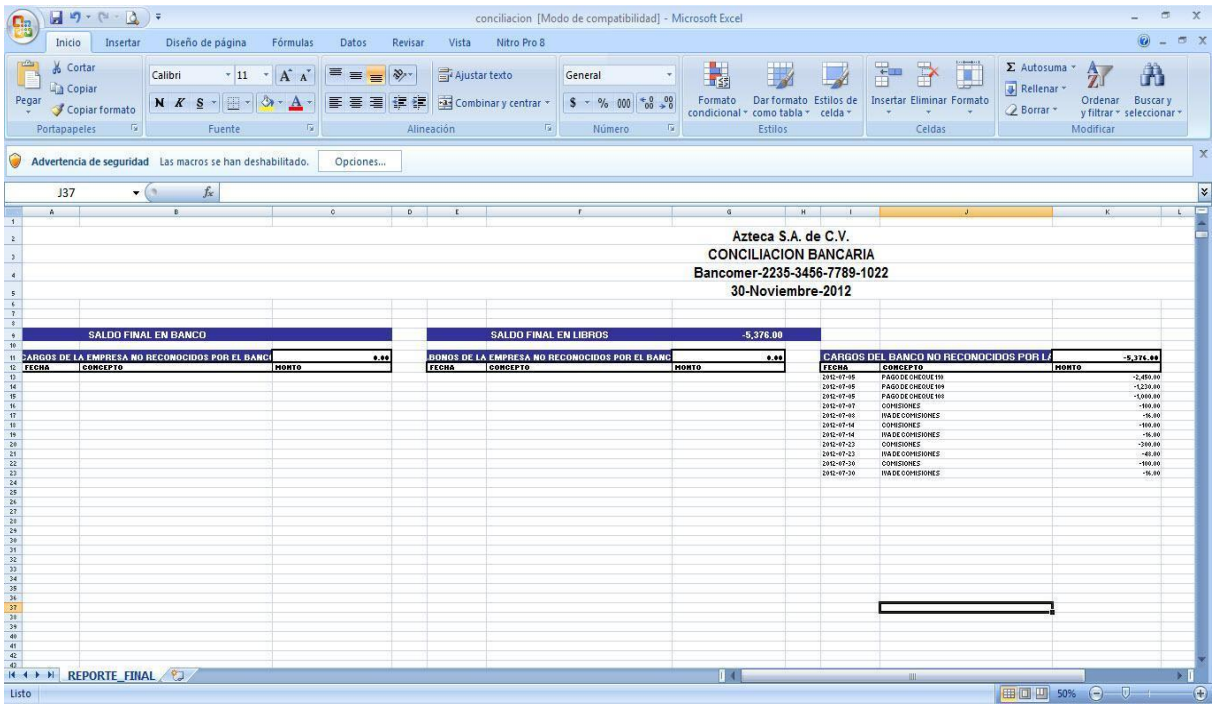


Imagen 4.0.29 Imagen que muestra el reporte final.

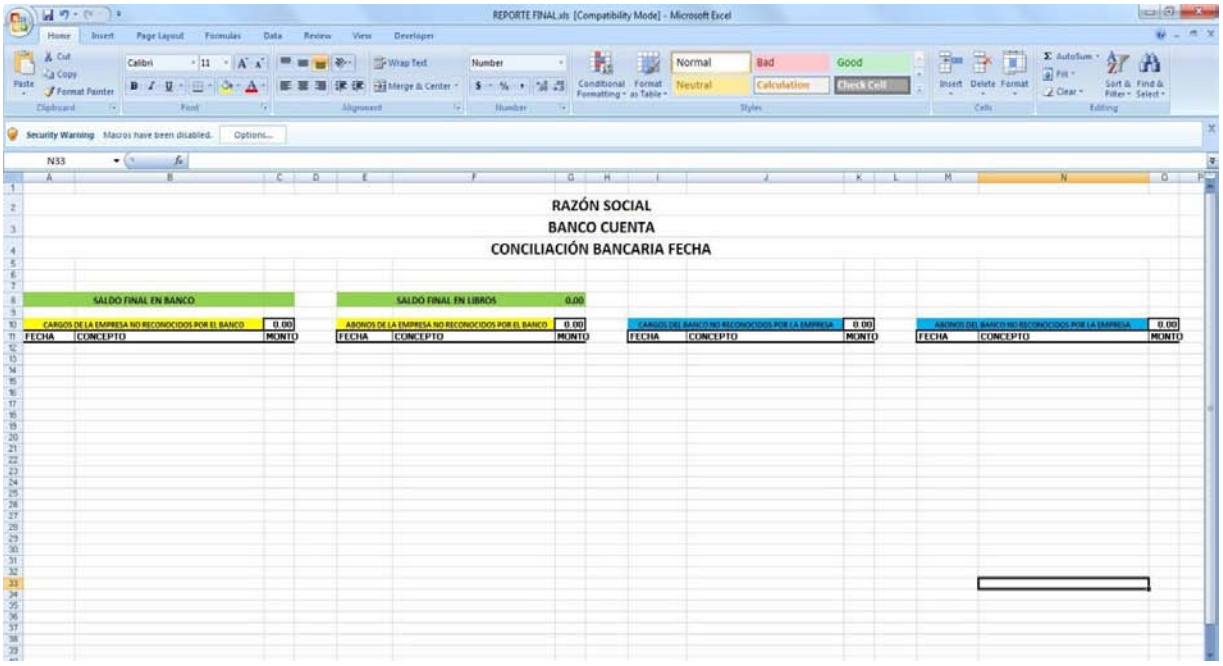


Imagen 4.0.30 Imagen que muestra el formato final del reporte que se muestra al usuario después de algunos ajustes en los encabezados.

Uno de los primeros problemas a lo que nos enfrentamos fue conectar la aplicación con Microsoft Excel. Realizamos una investigación al respecto para saber si ya existía alguna forma de poder realizarlo desde Java, después de investigar en varias páginas web encontramos una librería que nos permitía realizar la conexión y además manipular los libros de Excel.

En la imagen 4.0.31 se muestra un problema al realizar las pruebas correspondientes en la conexión entre Java y Excel.

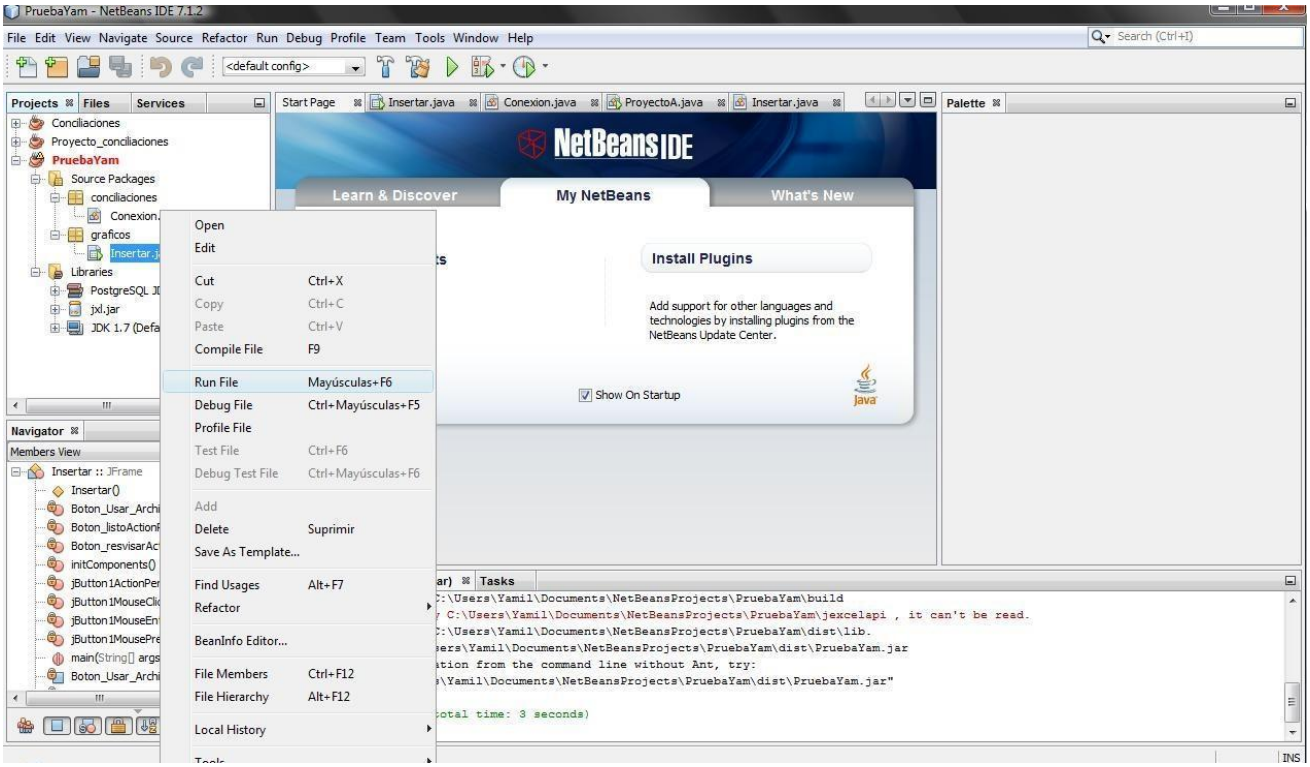


Imagen 4.0.31 Imagen que muestra un problema al intentar conectar Java con Excel.

El siguiente problema lo encontramos al extraer e insertar datos en el archivo de Excel por los tipos de datos que tenían que ser declarados en Java y además por el formato que los datos debían de tener en los libros para poder ser reconocidos y manipulados mediante Java.

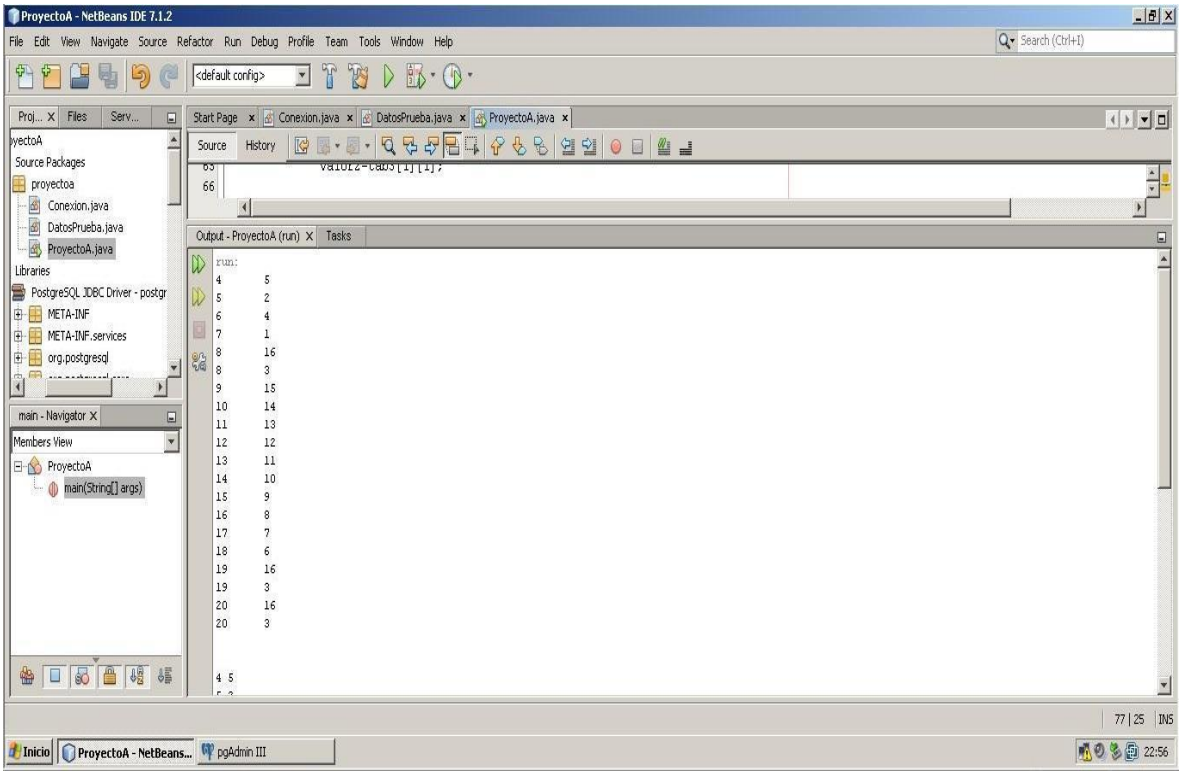


Imagen 4.0.32 Imagen que muestra una prueba realizada para imprimir los datos almacenados en las estructuras.

En la imagen 4.0.33 se muestra una de las tantas pruebas que se realizaron para validar que los datos extraídos no sufrieran cambios y además se pudieran almacenar en los campos correspondientes de las tablas de la base de datos.

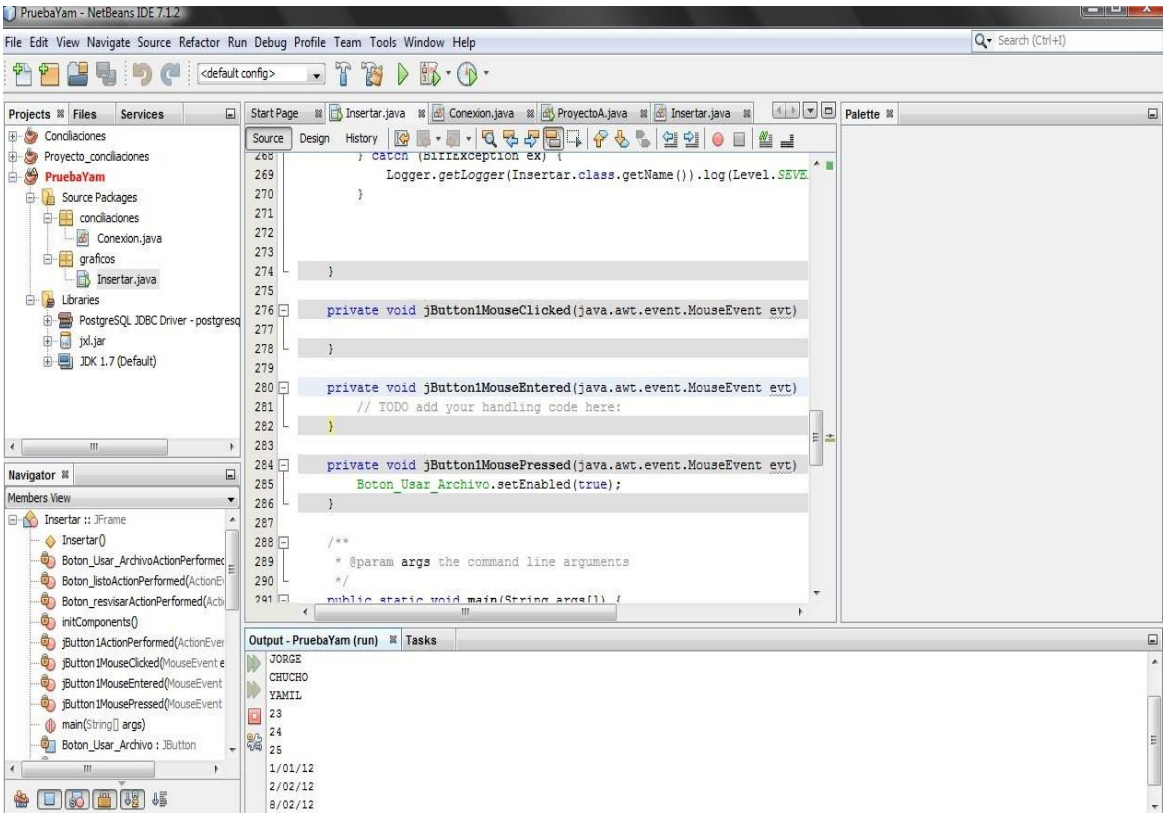


Imagen 4.0.33 Imagen en la cual se muestran los datos extraídos de los libros antes de ser insertados a la base de datos.

En esta etapa realizamos varias pruebas hasta conseguir leer todos los datos contenidos en los libros de Excel y procesarlos por medio de Java para poder insertarlos en la base de datos.

De la imagen 4.0.34 a la imagen 4.0.38 se muestran pruebas realizadas a lo largo del proyecto para poder lograr la carga inicial de datos por medio de los archivos de Excel.

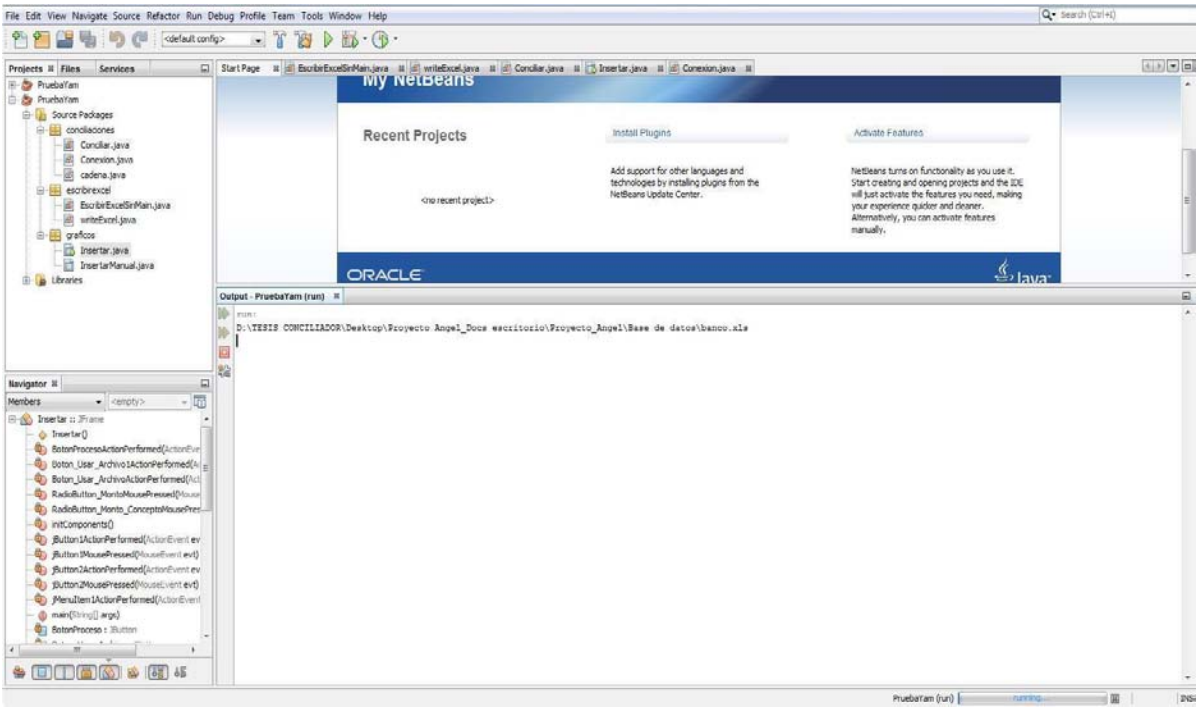


Imagen 4.0.34 En esta imagen se puede apreciar la impresión en consola de la ruta donde el archivo de Excel fue tomado.

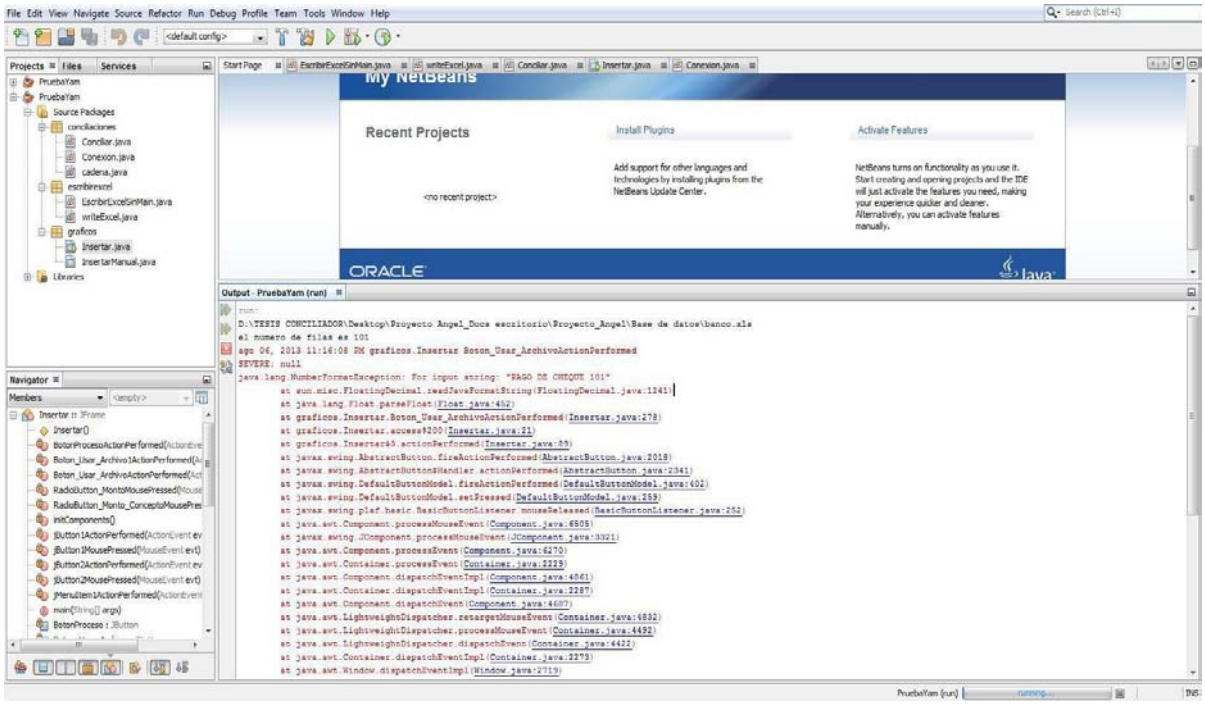


Imagen 4.0.35 Imagen que muestra un error al intentar extraer los datos del libro de Excel estado de cuenta.

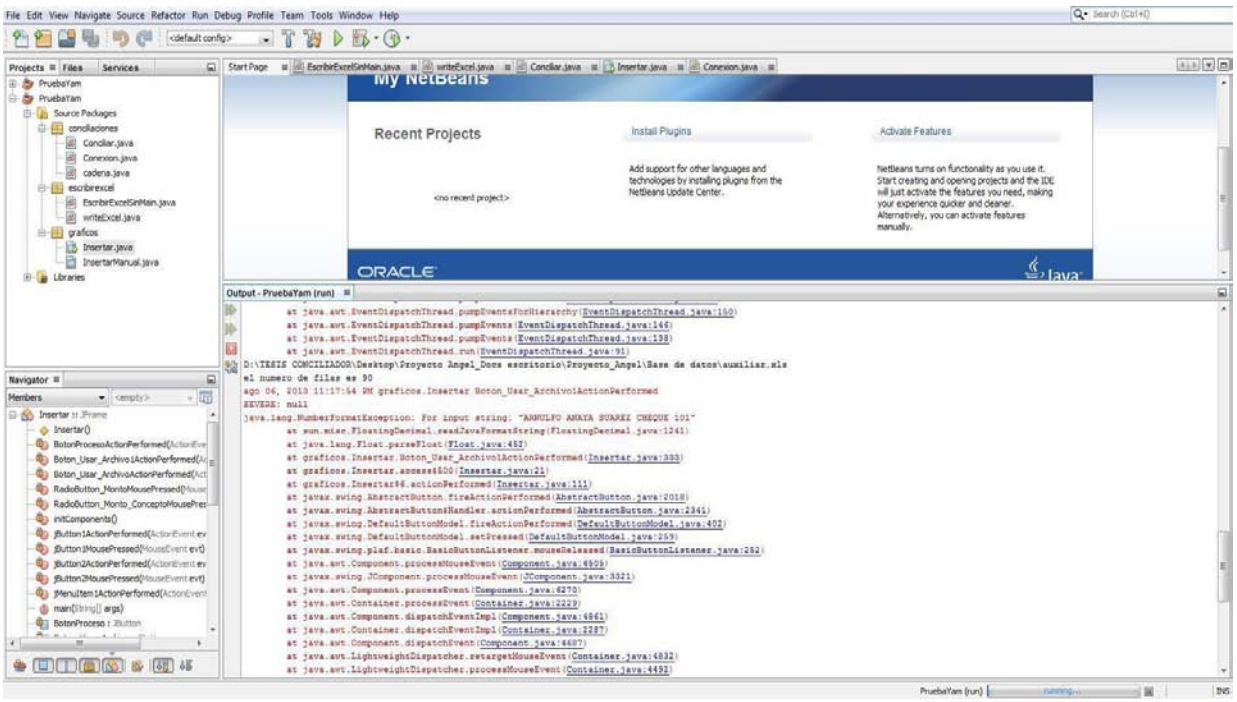


Imagen 4.0.36 Imagen que muestra un error al intentar extraer los datos del libro de Excel auxiliar de bancos.

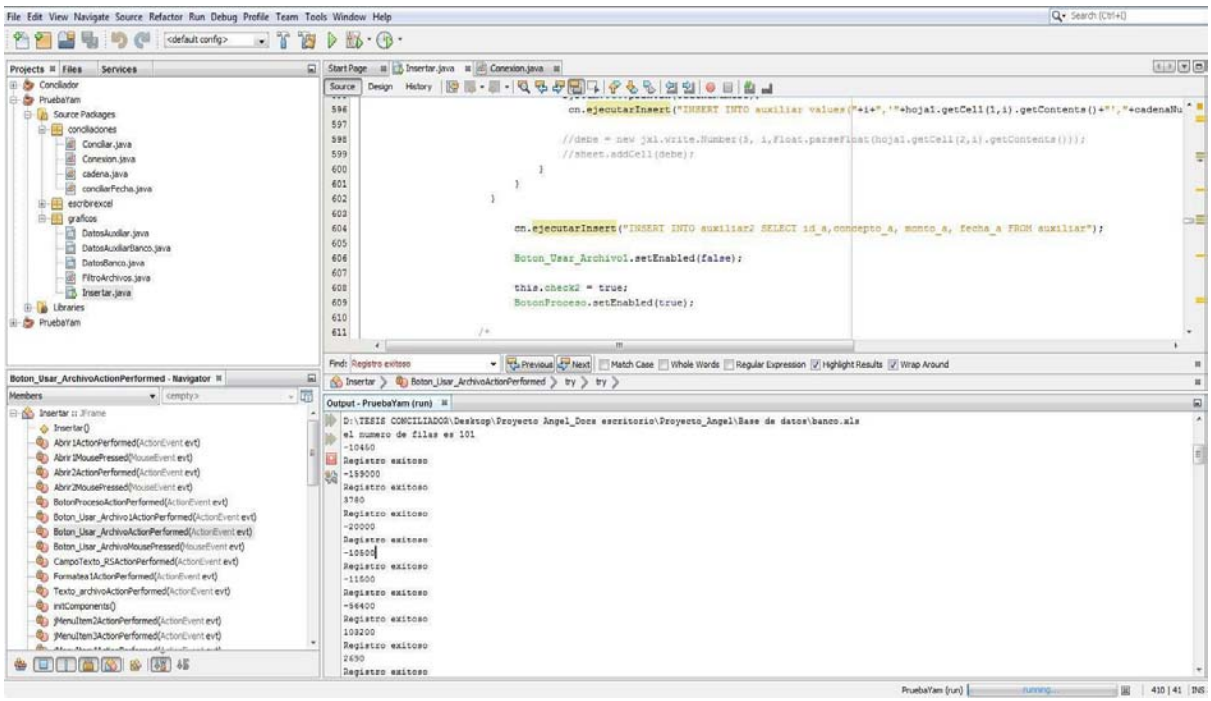


Imagen 4.0.37 Imagen que muestra la transformación de las cantidades tomadas del libro estado de cuenta, para insertarlas en la base de datos con el signo correspondiente, retiros (cantidades negativas) y depósitos (cantidades positivas).

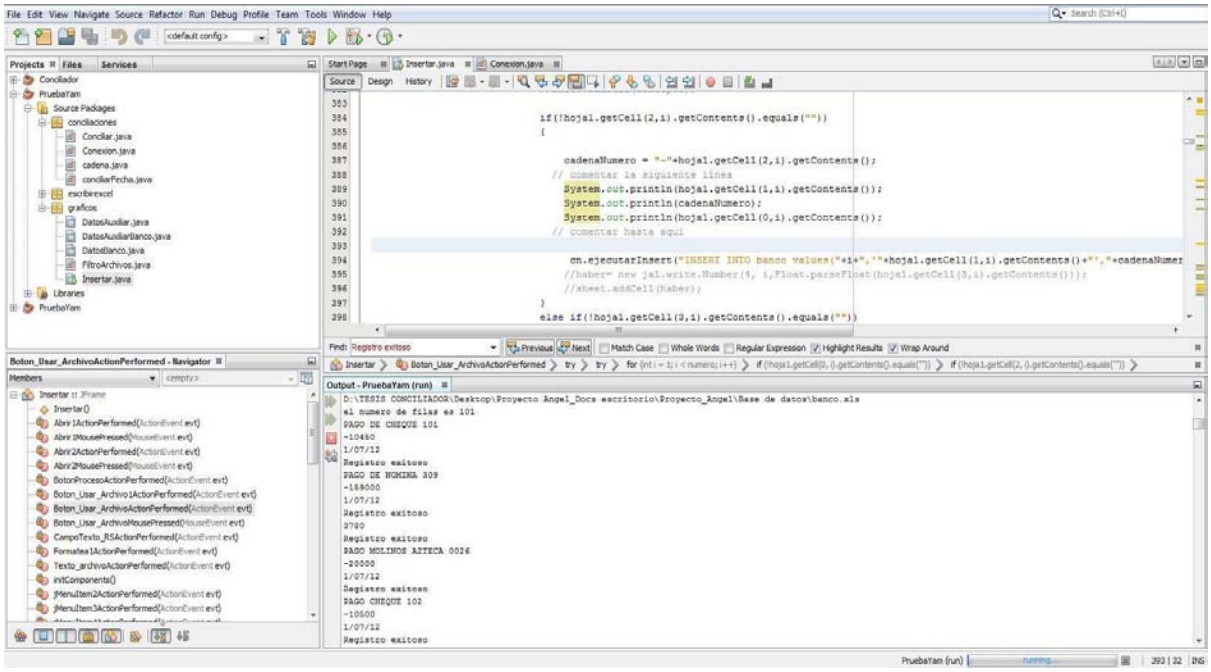
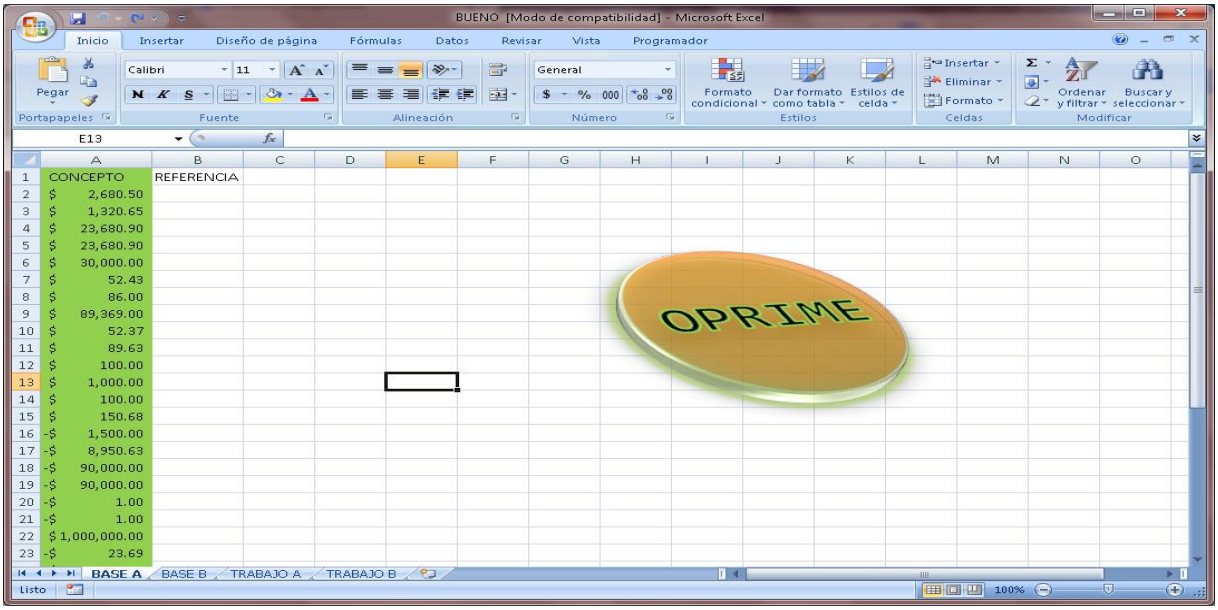


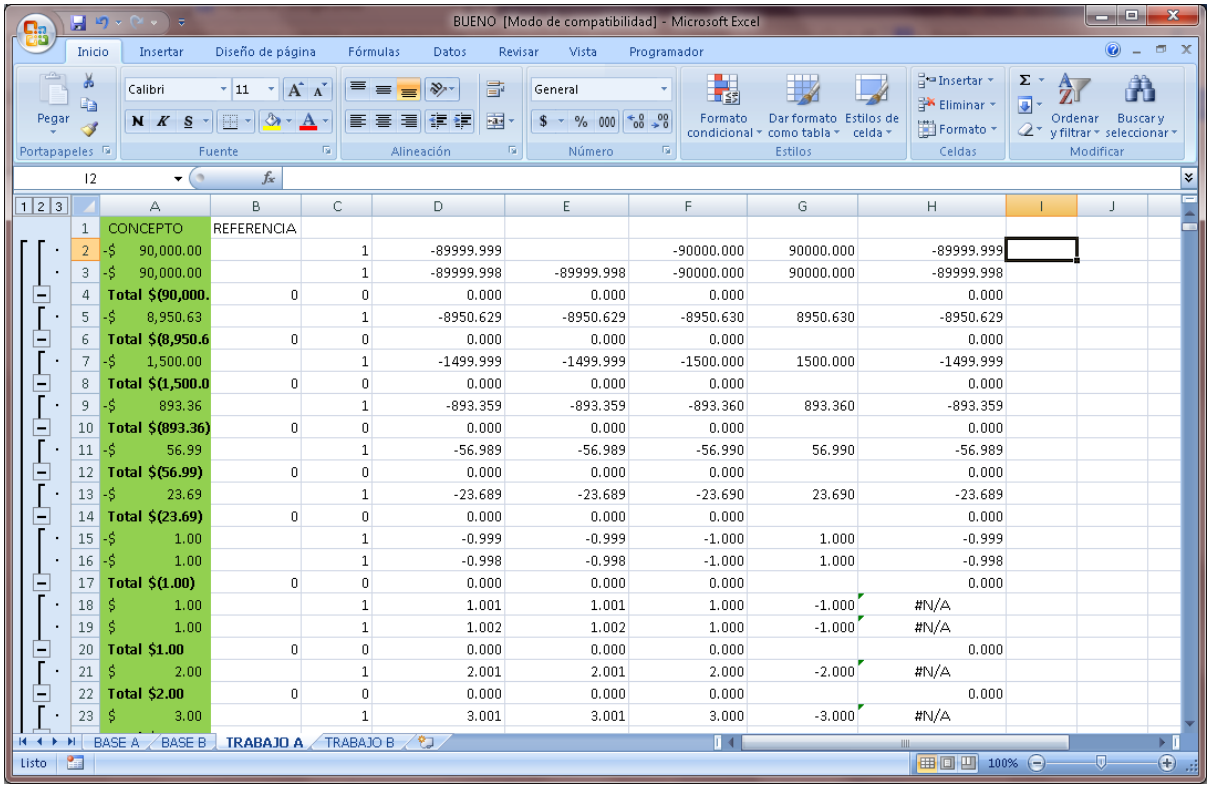
Imagen 4.0.38 Imagen que muestra los datos tomados del archivo Excel (concepto, monto y fecha) y que han sido insertados en la base de datos.

El siguiente reto fue lograr ejecutar desde Java macros creadas en Excel, esto debido a los requerimientos del cliente y ajustar los resultados obtenidos por medio de la ejecución de tales macros al lenguaje Java.



	A	B
1	CONCEPTO	REFERENCIA
2	$ 2,680.50	
3	$ 1,320.65	
4	$ 23,680.90	
5	$ 23,680.90	
6	$ 30,000.00	
7	$ 52.43	
8	$ 86.00	
9	$ 89,369.00	
10	$ 52.37	
11	$ 89.63	
12	$ 100.00	
13	$ 1,000.00	
14	$ 100.00	
15	$ 150.68	
16	-$ 1,500.00	
17	-$ 8,950.63	
18	-$ 90,000.00	
19	-$ 90,000.00	
20	-$ 1.00	
21	-$ 1.00	
22	$ 1,000,000.00	
23	-$ 23.69	

Imagen 4.0.39 Imagen que muestra el archivo con la macro original que entregó el cliente la cual solo realiza la conciliación por monto.



	A	B	C	D	E	F	G	H	I	J
1	CONCEPTO	REFERENCIA								
2	-$ 90,000.00		1	-89999.999		-90000.000	90000.000	-89999.999		
3	-$ 90,000.00		1	-89999.998	-89999.998	-90000.000	90000.000	-89999.998		
4	**Total $(90,000.**	0	0	0.000	0.000	0.000		0.000		
5	-$ 8,950.63		1	-8950.629	-8950.629	-8950.630	8950.630	-8950.629		
6	**Total $(8,950.6**	0	0	0.000	0.000	0.000		0.000		
7	-$ 1,500.00		1	-1499.999	-1499.999	-1500.000	1500.000	-1499.999		
8	**Total $(1,500.0**	0	0	0.000	0.000	0.000		0.000		
9	-$ 893.36		1	-893.359	-893.359	-893.360	893.360	-893.359		
10	**Total $(893.36)**	0	0	0.000	0.000	0.000		0.000		
11	-$ 56.99		1	-56.989	-56.989	-56.990	56.990	-56.989		
12	**Total $(56.99)**	0	0	0.000	0.000	0.000		0.000		
13	-$ 23.69		1	-23.689	-23.689	-23.690	23.690	-23.689		
14	**Total $(23.69)**	0	0	0.000	0.000	0.000		0.000		
15	-$ 1.00		1	-0.999	-0.999	-1.000	1.000	-0.999		
16	-$ 1.00		1	-0.998	-0.998	-1.000	1.000	-0.998		
17	**Total $(1.00)**	0	0	0.000	0.000	0.000		0.000		
18	$ 1.00		1	1.001	1.001	1.000	-1.000	#N/A		
19	$ 1.00		1	1.002	1.002	1.000	-1.000	#N/A		
20	**Total $1.00**	0	0	0.000	0.000	0.000		0.000		
21	$ 2.00		1	2.001	2.001	2.000	-2.000	#N/A		
22	**Total $2.00**	0	0	0.000	0.000	0.000		0.000		
23	$ 3.00		1	3.001	3.001	3.000	-3.000	#N/A		

Imagen 4.0.40 Imagen que muestra los resultados que arrojaba la macro entregada por el cliente.

Al no encontrar una forma directa de ejecutar las macros de Excel desde Java, se tuvo que investigar la manera en que éstas se ejecutarán, inclusive en segundo plano de tal manera que no fuera necesario abrir el archivo y las operaciones fueran transparentes para el usuario final para que éste no pudiera intervenir en dicho proceso y que el resultado fuera el esperado sin haber sido modificado o cancelado durante su ejecución.

Tras investigar encontramos que existía el lenguaje Visual Basic Script que permite manipular archivos y aplicaciones de Microsoft por medio de pequeños programas y que además ofrecía la ventaja de que no se necesitaba instalar ninguna librería ya que el sistema operativo Windows ya lo trae instalado.

Si bien se pudo resolver el primer problema se generó uno nuevo, como correr programas con extensión .vbs desde Java, al continuar con la investigación vimos que los archivos de Visual Basic Script se pueden ejecutar desde la aplicación “cmd” por otra parte Java tiene implementado algunos métodos que le permiten correr ejecutables externos tomando control de la misma aplicación "cmd", esto fue una gran ventaja ya que Java podía saber cuándo terminaban de correr los procesos externos y así poder avanzar sin generar conflictos ya que los resultados obtenidos de forma externa eran requeridos para un nuevo proceso.

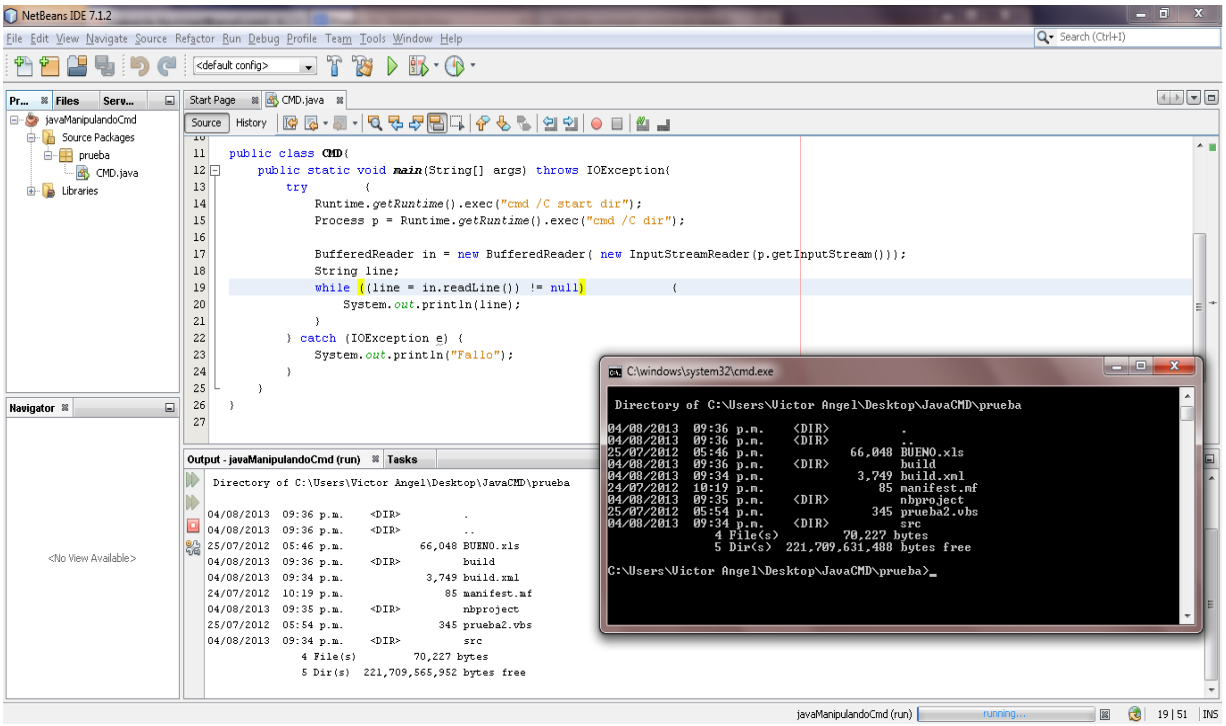


Imagen 4.0.41 Imagen en la que se muestra como Java manipula y obtiene información de cmd.

Esto nos permitió poder operar con los archivos de Excel que se usarían para mostrar el reporte final, gracias a este lenguaje logramos correr las macros existentes para poder aplicar el formato debido a los datos en los archivos y con ello poder procesarlos y entregar los datos requeridos en el formato específico para poder ser trasladados al lenguaje Java y con ello continuar hasta tener los datos que finalmente serían presentados en el reporte final.

Al principio se tomó la decisión de crear únicamente dos tablas para almacenar la información, una almacenaría los datos del libro auxiliar y otra los datos del estado de cuenta, conforme avanzó el desarrollo, nos dimos cuenta de que dos tablas no eran suficientes para poder procesar la información por medio de los tres métodos, fuimos creando nuevas tablas para poder reacomodar los datos que no habían sido conciliados por el primer método, lo mismo se realizó para los datos que no fueron conciliados por el segundo método.

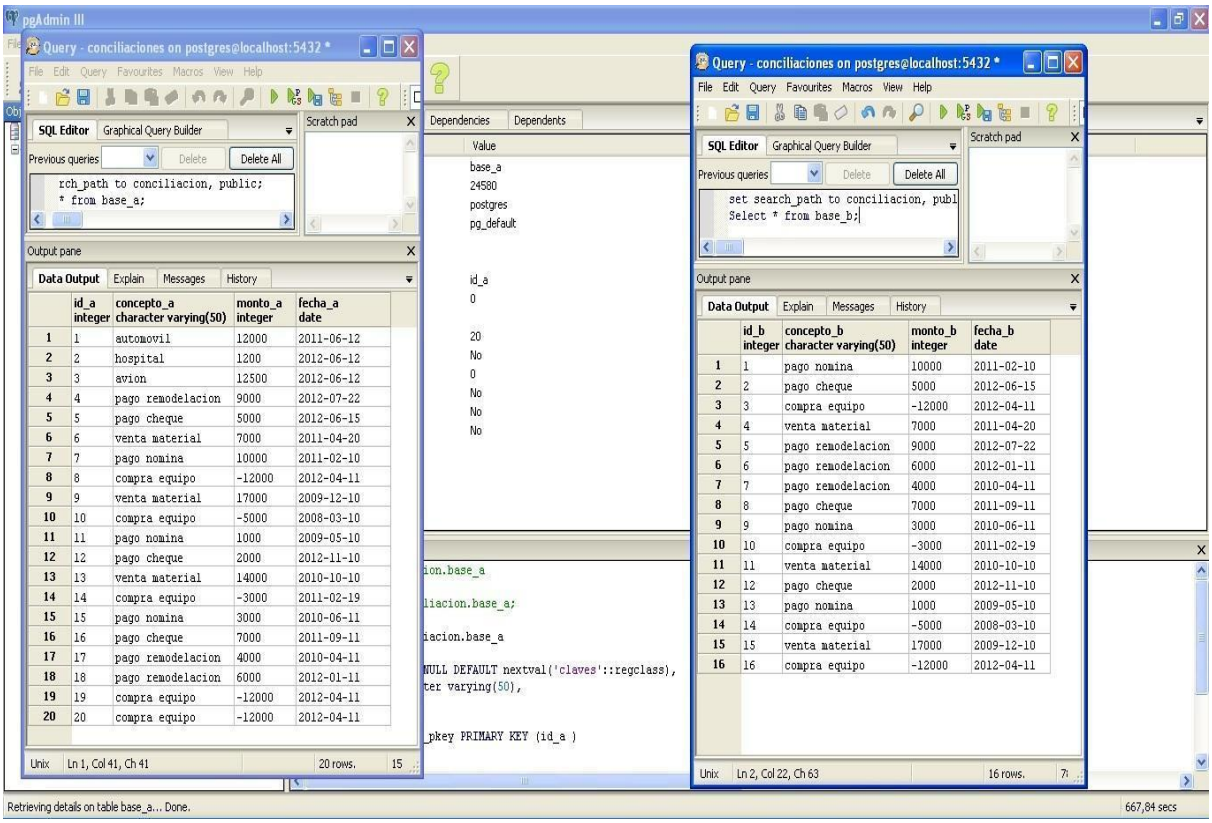
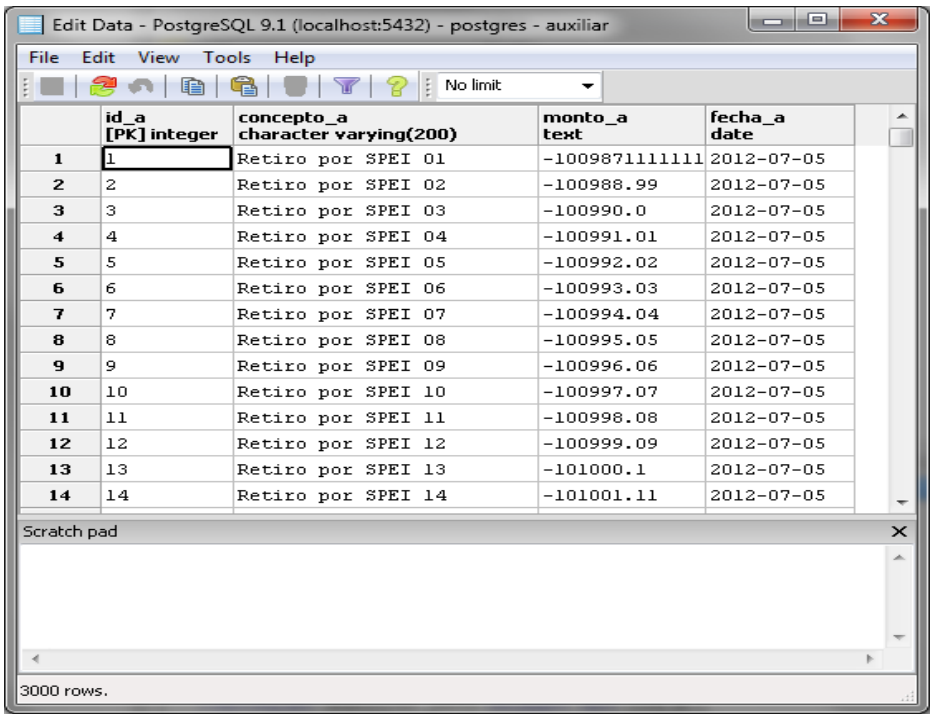


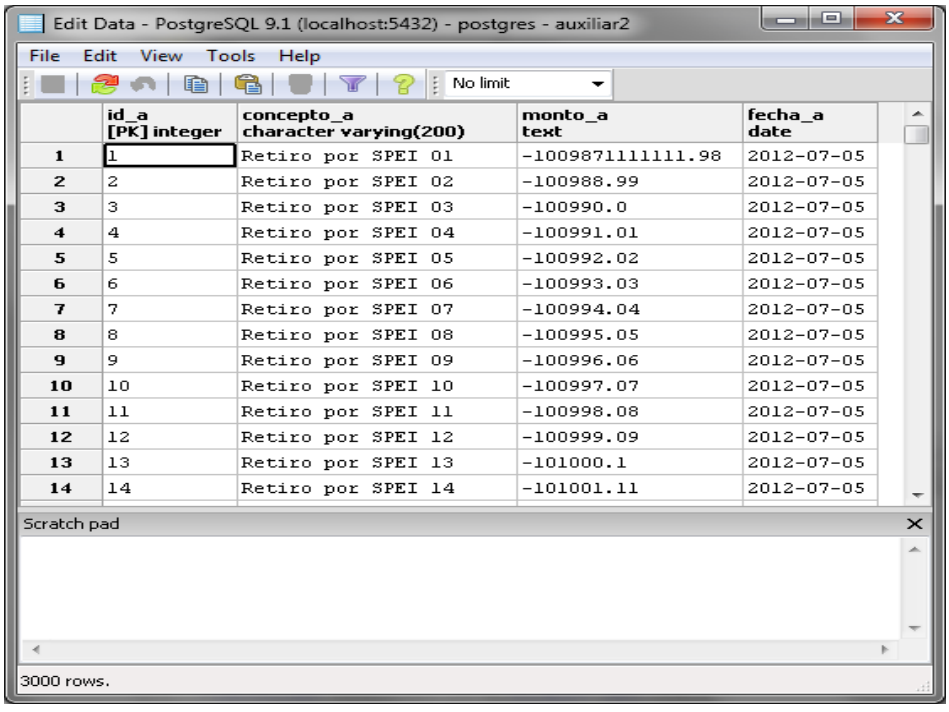
Imagen 4.0.42 Imagen que muestra la estructura inicial de las tablas representando la información de ambos libros de Excel.

Las siguientes imágenes muestran la estructura de todas las tablas usadas en la versión final para poder procesar la información por medio de los tres métodos.



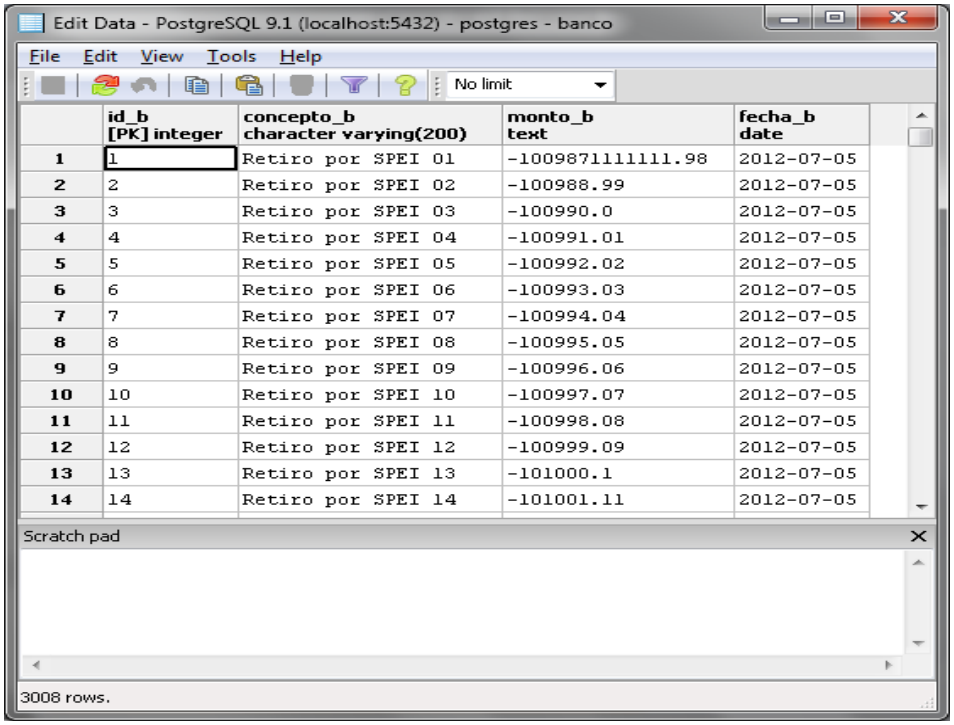
	id_a [PK] integer	concepto_a character varying(200)	monto_a text	fecha_a date
1	1	Retiro por SPEI 01	-1009871111111	2012-07-05
2	2	Retiro por SPEI 02	-100988.99	2012-07-05
3	3	Retiro por SPEI 03	-100990.0	2012-07-05
4	4	Retiro por SPEI 04	-100991.01	2012-07-05
5	5	Retiro por SPEI 05	-100992.02	2012-07-05
6	6	Retiro por SPEI 06	-100993.03	2012-07-05
7	7	Retiro por SPEI 07	-100994.04	2012-07-05
8	8	Retiro por SPEI 08	-100995.05	2012-07-05
9	9	Retiro por SPEI 09	-100996.06	2012-07-05
10	10	Retiro por SPEI 10	-100997.07	2012-07-05
11	11	Retiro por SPEI 11	-100998.08	2012-07-05
12	12	Retiro por SPEI 12	-100999.09	2012-07-05
13	13	Retiro por SPEI 13	-101000.1	2012-07-05
14	14	Retiro por SPEI 14	-101001.11	2012-07-05

Imagen 4.0.43 Imagen que muestra la estructura de la tabla auxiliar que será consultada con el botón Opciones "Ver datos de auxiliar" o F2.



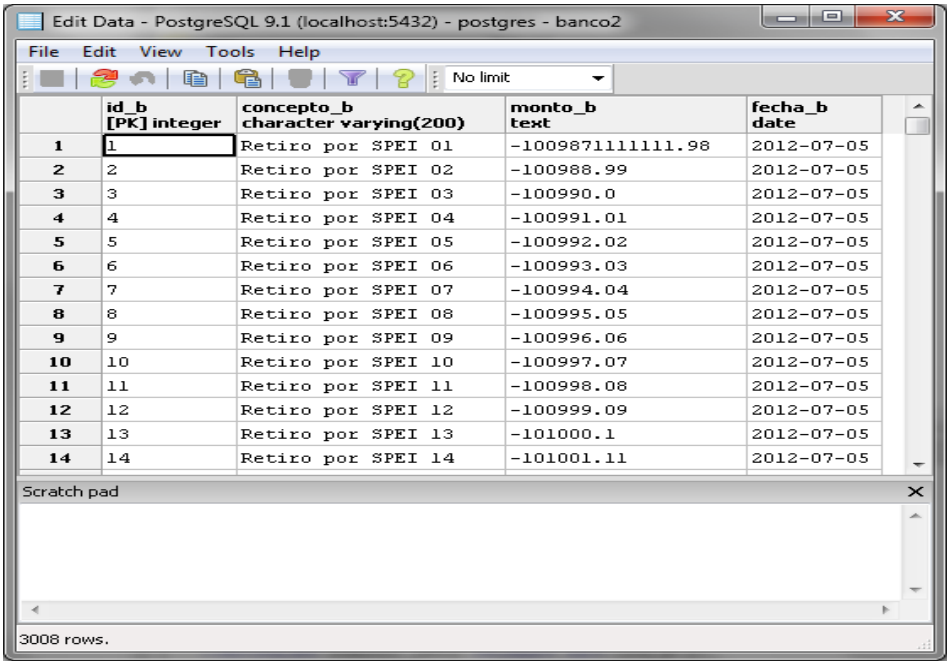
	id_a [PK] integer	concepto_a character varying(200)	monto_a text	fecha_a date
1	1	Retiro por SPEI 01	-1009871111111.98	2012-07-05
2	2	Retiro por SPEI 02	-100988.99	2012-07-05
3	3	Retiro por SPEI 03	-100990.0	2012-07-05
4	4	Retiro por SPEI 04	-100991.01	2012-07-05
5	5	Retiro por SPEI 05	-100992.02	2012-07-05
6	6	Retiro por SPEI 06	-100993.03	2012-07-05
7	7	Retiro por SPEI 07	-100994.04	2012-07-05
8	8	Retiro por SPEI 08	-100995.05	2012-07-05
9	9	Retiro por SPEI 09	-100996.06	2012-07-05
10	10	Retiro por SPEI 10	-100997.07	2012-07-05
11	11	Retiro por SPEI 11	-100998.08	2012-07-05
12	12	Retiro por SPEI 12	-100999.09	2012-07-05
13	13	Retiro por SPEI 13	-101000.1	2012-07-05
14	14	Retiro por SPEI 14	-101001.11	2012-07-05

Imagen 4.0.44 Imagen que muestra la estructura de la tabla auxiliar que será utilizada durante el proceso.



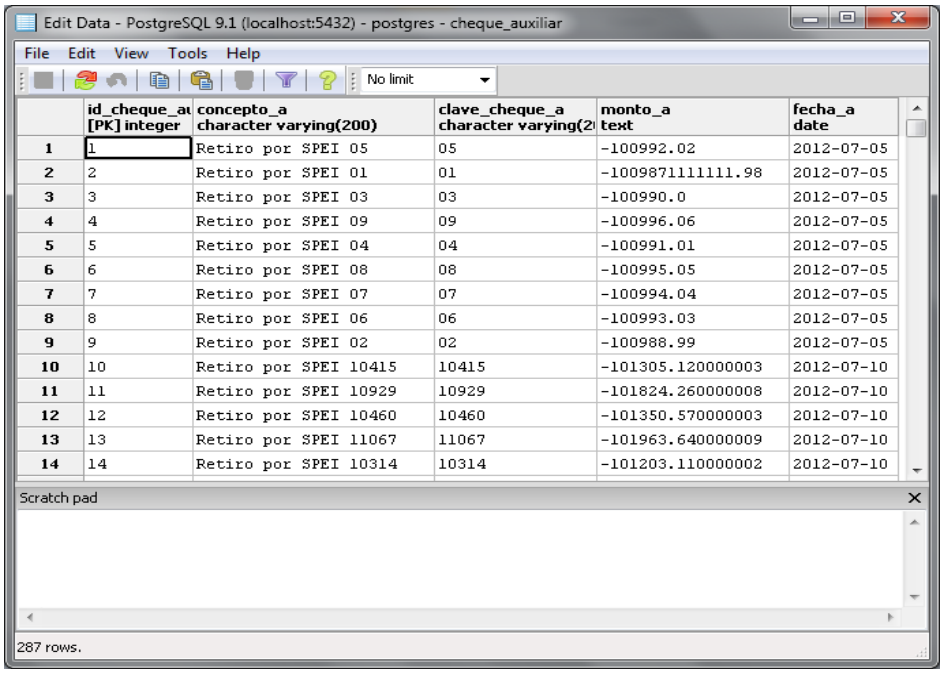
	id_b [PK] integer	concepto_b character varying(200)	monto_b text	fecha_b date
1	1	Retiro por SPEI 01	-1009871111111.98	2012-07-05
2	2	Retiro por SPEI 02	-100988.99	2012-07-05
3	3	Retiro por SPEI 03	-100990.0	2012-07-05
4	4	Retiro por SPEI 04	-100991.01	2012-07-05
5	5	Retiro por SPEI 05	-100992.02	2012-07-05
6	6	Retiro por SPEI 06	-100993.03	2012-07-05
7	7	Retiro por SPEI 07	-100994.04	2012-07-05
8	8	Retiro por SPEI 08	-100995.05	2012-07-05
9	9	Retiro por SPEI 09	-100996.06	2012-07-05
10	10	Retiro por SPEI 10	-100997.07	2012-07-05
11	11	Retiro por SPEI 11	-100998.08	2012-07-05
12	12	Retiro por SPEI 12	-100999.09	2012-07-05
13	13	Retiro por SPEI 13	-101000.1	2012-07-05
14	14	Retiro por SPEI 14	-101001.11	2012-07-05

Imagen 4.0.45 Imagen que muestra la estructura de la tabla banco que será consultada con el botón Opciones “Ver datos de banco” o F3.



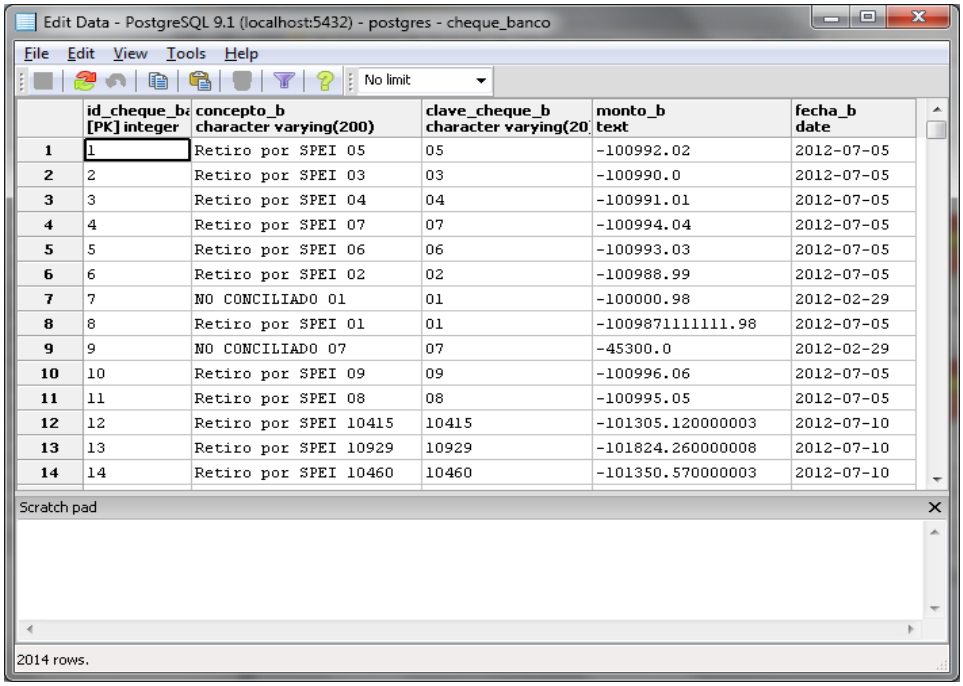
	id_b [PK] integer	concepto_b character varying(200)	monto_b text	fecha_b date
1	1	Retiro por SPEI 01	-1009871111111.98	2012-07-05
2	2	Retiro por SPEI 02	-100988.99	2012-07-05
3	3	Retiro por SPEI 03	-100990.0	2012-07-05
4	4	Retiro por SPEI 04	-100991.01	2012-07-05
5	5	Retiro por SPEI 05	-100992.02	2012-07-05
6	6	Retiro por SPEI 06	-100993.03	2012-07-05
7	7	Retiro por SPEI 07	-100994.04	2012-07-05
8	8	Retiro por SPEI 08	-100995.05	2012-07-05
9	9	Retiro por SPEI 09	-100996.06	2012-07-05
10	10	Retiro por SPEI 10	-100997.07	2012-07-05
11	11	Retiro por SPEI 11	-100998.08	2012-07-05
12	12	Retiro por SPEI 12	-100999.09	2012-07-05
13	13	Retiro por SPEI 13	-101000.1	2012-07-05
14	14	Retiro por SPEI 14	-101001.11	2012-07-05

Imagen 4.0.46 Imagen que muestra la estructura de la tabla banco que será utilizada durante el proceso.



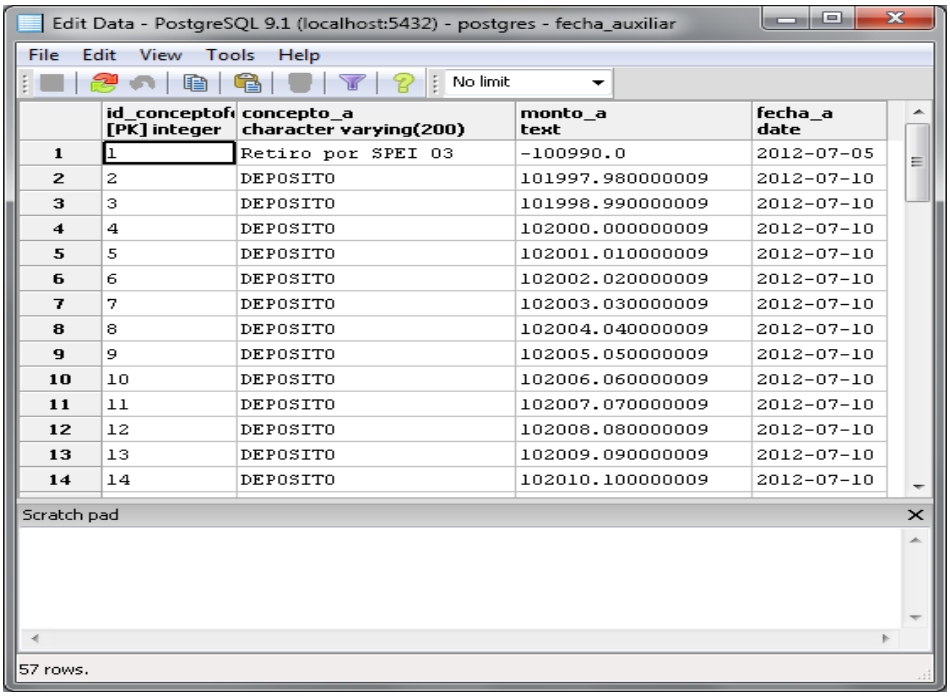
	id_cheque_a [PK] integer	concepto_a character varying(200)	clave_cheque_a character varying(2	monto_a text	fecha_a date
1	1	Retiro por SPEI 05	05	-100992.02	2012-07-05
2	2	Retiro por SPEI 01	01	-1009871111111.98	2012-07-05
3	3	Retiro por SPEI 03	03	-100990.0	2012-07-05
4	4	Retiro por SPEI 09	09	-100996.06	2012-07-05
5	5	Retiro por SPEI 04	04	-100991.01	2012-07-05
6	6	Retiro por SPEI 08	08	-100995.05	2012-07-05
7	7	Retiro por SPEI 07	07	-100994.04	2012-07-05
8	8	Retiro por SPEI 06	06	-100993.03	2012-07-05
9	9	Retiro por SPEI 02	02	-100988.99	2012-07-05
10	10	Retiro por SPEI 10415	10415	-101305.120000003	2012-07-10
11	11	Retiro por SPEI 10929	10929	-101824.260000008	2012-07-10
12	12	Retiro por SPEI 10460	10460	-101350.570000003	2012-07-10
13	13	Retiro por SPEI 11067	11067	-101963.640000009	2012-07-10
14	14	Retiro por SPEI 10314	10314	-101203.110000002	2012-07-10

Imagen 4.0.47 Imagen que muestra la estructura de la tabla cheque auxiliar, la cual contiene los datos que sobraron después del proceso concepto y monto.



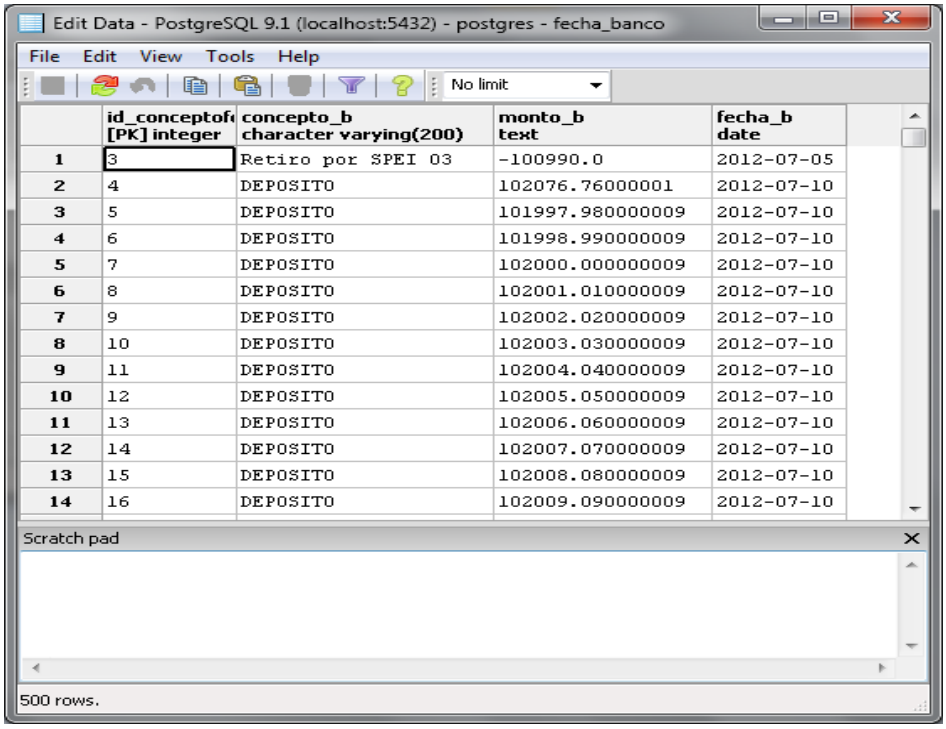
	id_cheque_b [PK] integer	concepto_b character varying(200)	clave_cheque_b character varying(20	monto_b text	fecha_b date
1	1	Retiro por SPEI 05	05	-100992.02	2012-07-05
2	2	Retiro por SPEI 03	03	-100990.0	2012-07-05
3	3	Retiro por SPEI 04	04	-100991.01	2012-07-05
4	4	Retiro por SPEI 07	07	-100994.04	2012-07-05
5	5	Retiro por SPEI 06	06	-100993.03	2012-07-05
6	6	Retiro por SPEI 02	02	-100988.99	2012-07-05
7	7	NO CONCILIADO 01	01	-100000.98	2012-02-29
8	8	Retiro por SPEI 01	01	-1009871111111.98	2012-07-05
9	9	NO CONCILIADO 07	07	-45300.0	2012-02-29
10	10	Retiro por SPEI 09	09	-100996.06	2012-07-05
11	11	Retiro por SPEI 08	08	-100995.05	2012-07-05
12	12	Retiro por SPEI 10415	10415	-101305.120000003	2012-07-10
13	13	Retiro por SPEI 10929	10929	-101824.260000008	2012-07-10
14	14	Retiro por SPEI 10460	10460	-101350.570000003	2012-07-10

Imagen 4.0.48 Imagen que muestra la estructura de la tabla cheque banco, la cual contiene los datos que sobraron después del proceso concepto y monto.



	id_conceptof [PK] integer	concepto_a character varying(200)	monto_a text	fecha_a date
1	1	Retiro por SPEI 03	-100990.0	2012-07-05
2	2	DEPOSITO	101997.980000009	2012-07-10
3	3	DEPOSITO	101998.990000009	2012-07-10
4	4	DEPOSITO	102000.000000009	2012-07-10
5	5	DEPOSITO	102001.010000009	2012-07-10
6	6	DEPOSITO	102002.020000009	2012-07-10
7	7	DEPOSITO	102003.030000009	2012-07-10
8	8	DEPOSITO	102004.040000009	2012-07-10
9	9	DEPOSITO	102005.050000009	2012-07-10
10	10	DEPOSITO	102006.060000009	2012-07-10
11	11	DEPOSITO	102007.070000009	2012-07-10
12	12	DEPOSITO	102008.080000009	2012-07-10
13	13	DEPOSITO	102009.090000009	2012-07-10
14	14	DEPOSITO	102010.100000009	2012-07-10

Imagen 4.0.49 Imagen que muestra la estructura de la tabla fecha auxiliar, la cual contiene los datos que sobraron después del proceso monto y fecha.



	id_conceptof [PK] integer	concepto_b character varying(200)	monto_b text	fecha_b date
1	3	Retiro por SPEI 03	-100990.0	2012-07-05
2	4	DEPOSITO	102076.76000001	2012-07-10
3	5	DEPOSITO	101997.980000009	2012-07-10
4	6	DEPOSITO	101998.990000009	2012-07-10
5	7	DEPOSITO	102000.000000009	2012-07-10
6	8	DEPOSITO	102001.010000009	2012-07-10
7	9	DEPOSITO	102002.020000009	2012-07-10
8	10	DEPOSITO	102003.030000009	2012-07-10
9	11	DEPOSITO	102004.040000009	2012-07-10
10	12	DEPOSITO	102005.050000009	2012-07-10
11	13	DEPOSITO	102006.060000009	2012-07-10
12	14	DEPOSITO	102007.070000009	2012-07-10
13	15	DEPOSITO	102008.080000009	2012-07-10
14	16	DEPOSITO	102009.090000009	2012-07-10

Imagen 4.0.50 Imagen que muestra la estructura de la tabla fecha banco, la cual contiene los datos que sobraron después del proceso monto y fecha.

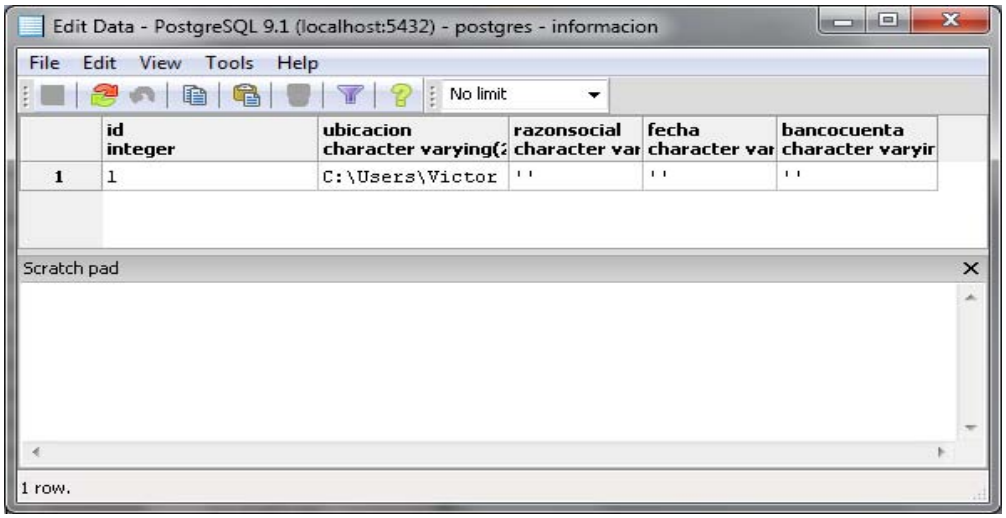


Imagen 4.0.51 Imagen que muestra la estructura de la tabla en la que se indica la ruta donde será almacenado el archivo.

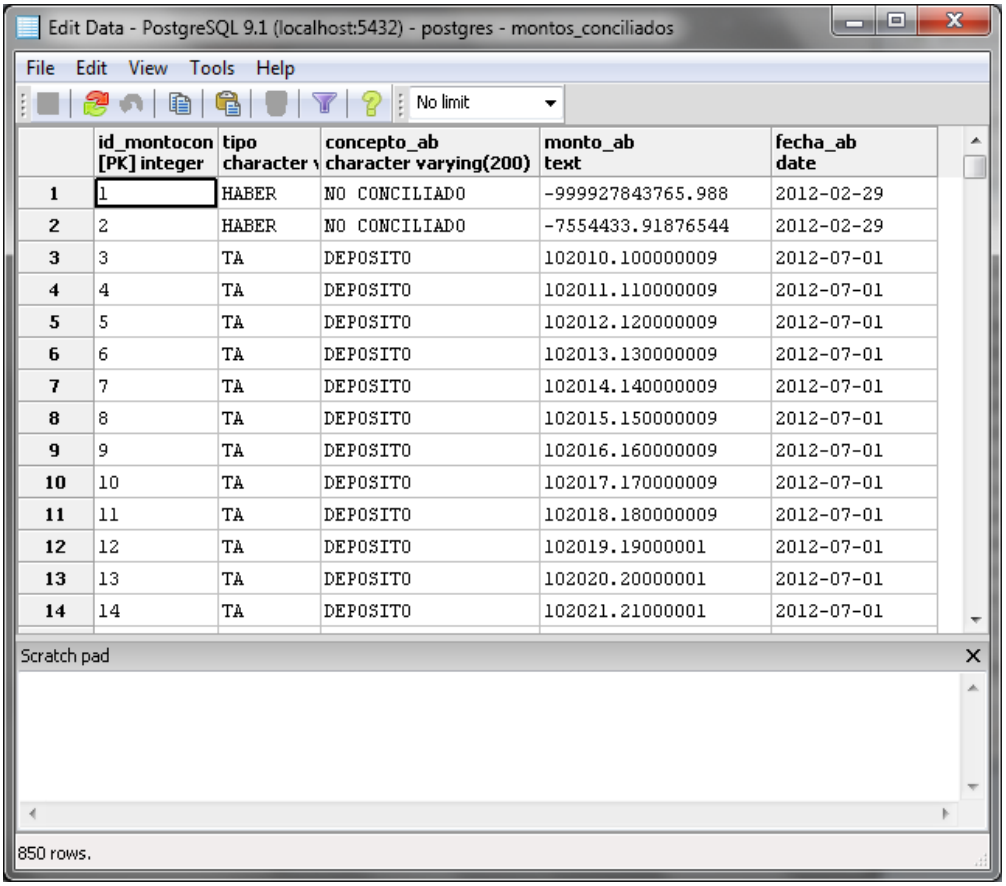


Imagen 4.0.52 Imagen que muestra la estructura de la tabla montos conciliados, la cual contiene los datos que sobraron después del proceso monto.

En la Imagen 4.0.53 se muestra el proceso final en consola una vez que los datos han sido cargados en la base de datos y las macros han sido ejecutadas junto con los tres métodos diseñados.

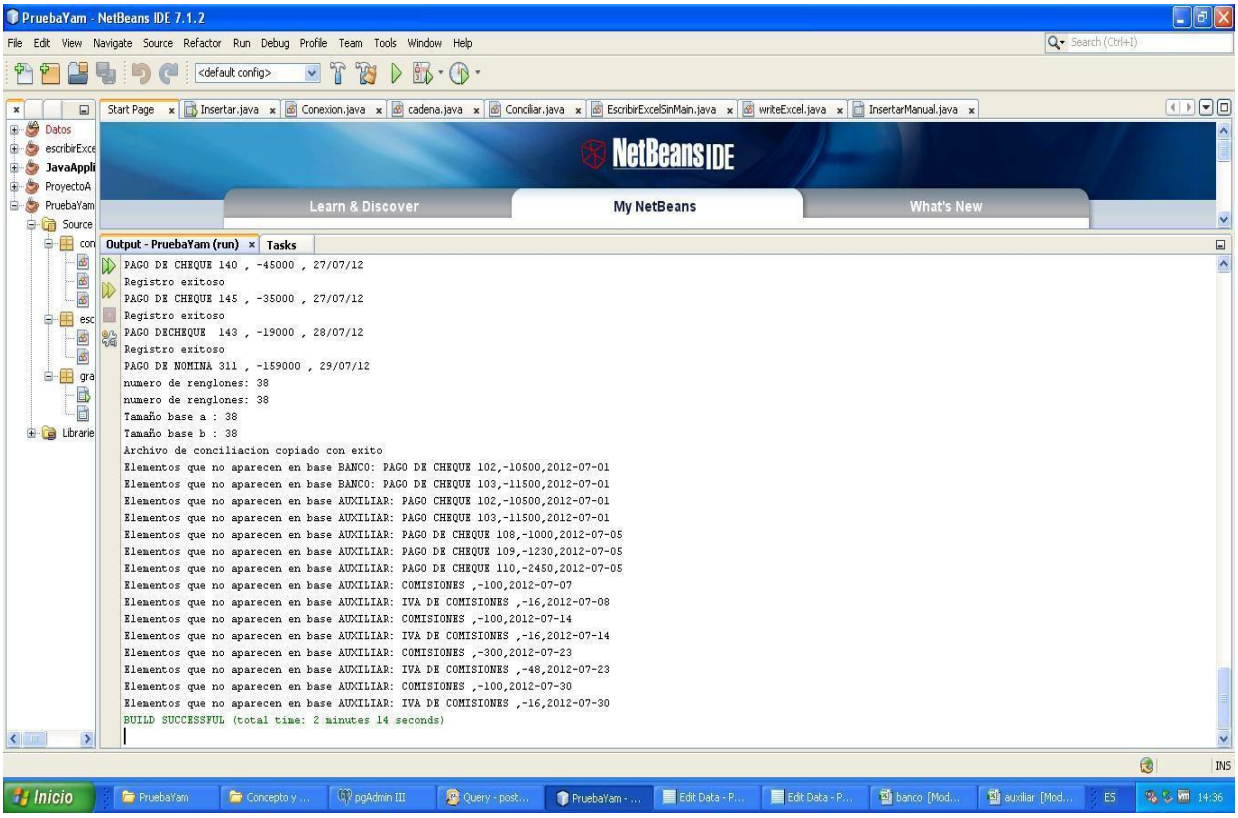


Imagen 4.0.53 Imagen que muestra el proceso final en consola.

5.0 Ajustes, pruebas y resultados experimentales

A lo largo del desarrollo de la aplicación fueron surgiendo varios cambios que algunas ocasiones eran necesarios por los nuevos requerimientos del cliente o algunas veces para poder resolver los problemas que iban surgiendo, como poder conectarnos a Excel, extraer los datos y poder insertarlos en la base de datos.

Por ejemplo una parte que costo mucho fue el poder insertar las cantidades numéricas de Excel a la base de datos por los tipos de datos que se representaban en las tablas y a la hora de volver a operar con ellos los resultados no eran los esperados, porque muchas veces le agregaba decimales de más internamente la base de datos a las cantidades por ser un tipo de dato flotante y al final con redondeo las cifras esperadas no eran las que la aplicación tendría que devolver.

En la imagen 5.0.0 se muestra una prueba al realizar la inserción de los datos contenidos en el libro Excel dentro de la base de datos, como se puede apreciar la imagen indica que algunos campos no se pudieron insertar correctamente dentro de la base de datos debido al formato que los mismos tenían en Excel.

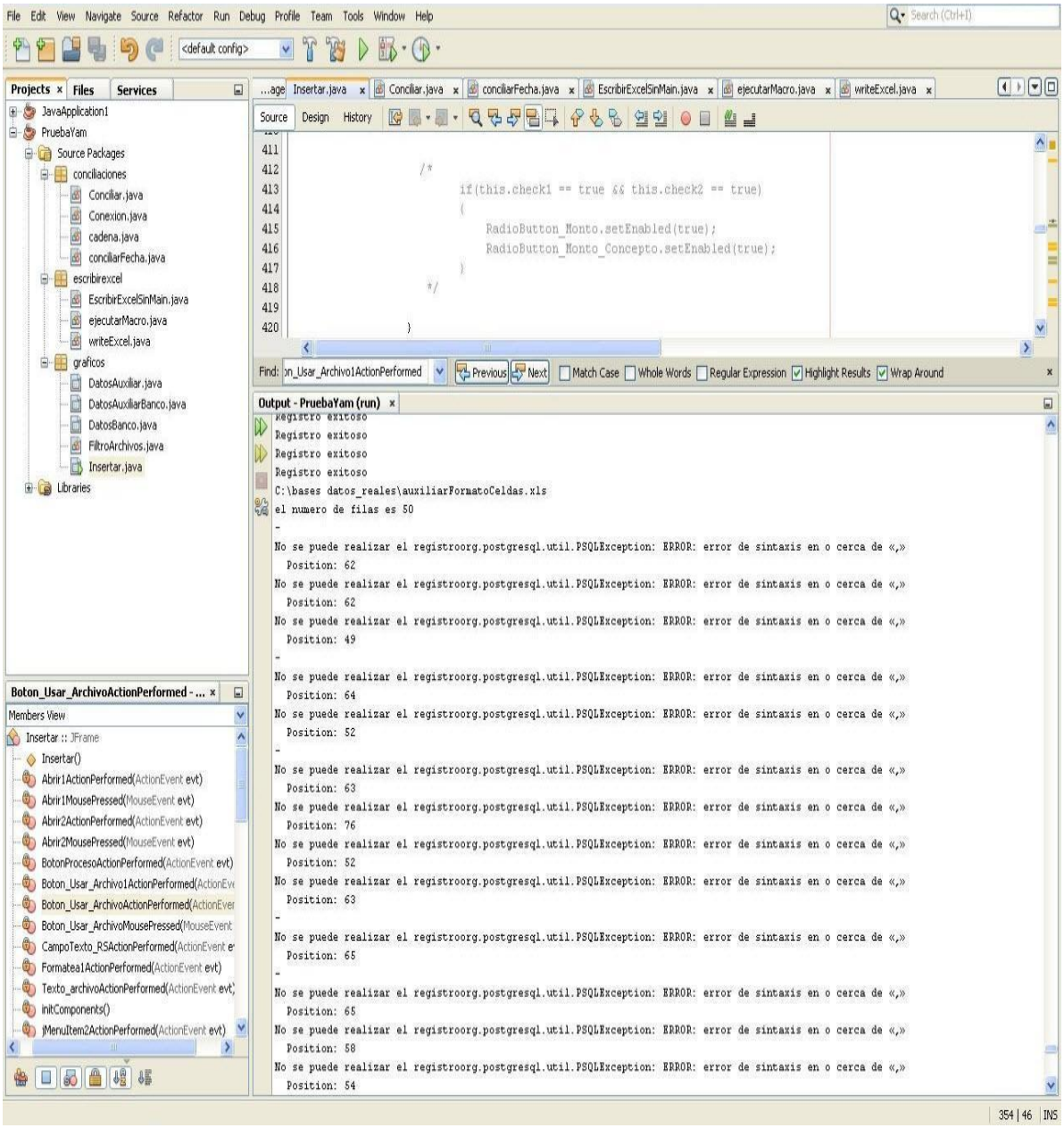


Imagen 5.0.0 Imagen que representa la inserción de datos del libro de Excel auxiliarFormatoCeldas.xls hacia la base de datos.

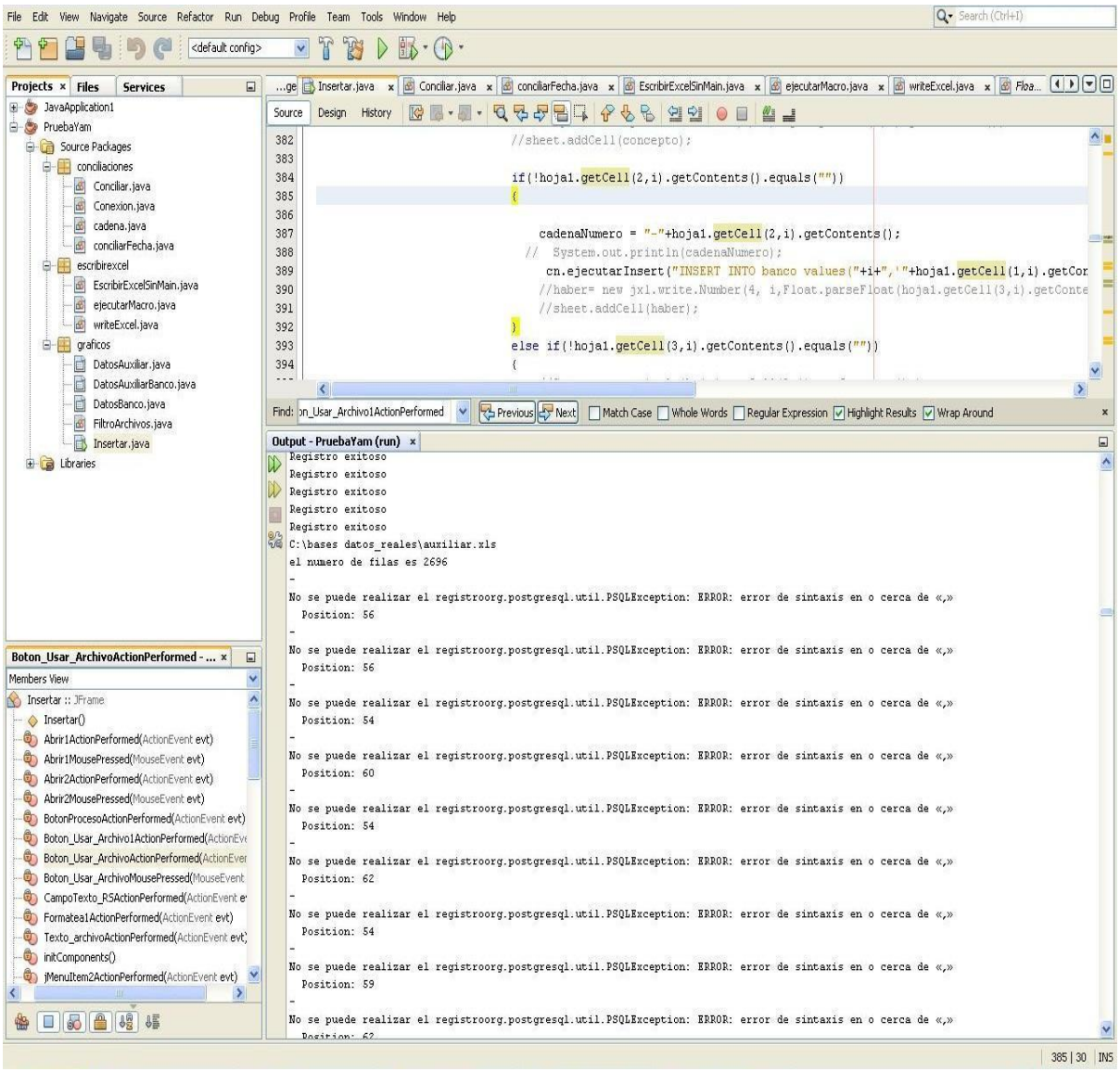


Imagen 5.0.1 Imagen que representa la inserción de los datos del libro de Excel auxiliar.xls a la base de datos.

A lo largo del proceso realizamos diversas pruebas con muchas bases de datos diferentes, algunas fueron para probar si el tipo de dato era almacenado correctamente dentro de la base de datos, para poder operar con él posteriormente, algunas de las bases fueron creadas por el equipo de desarrollo, otras fueron proporcionadas por el cliente para tener un ejemplo más apegado a la realidad.

En la imagen 5.0.2 se puede apreciar una base de datos para realizar pruebas de carga inicial, esto con la finalidad de observar el comportamiento de la aplicación al insertar 4500 registros del libro auxiliar y 5000 registros del libro extracto bancario.

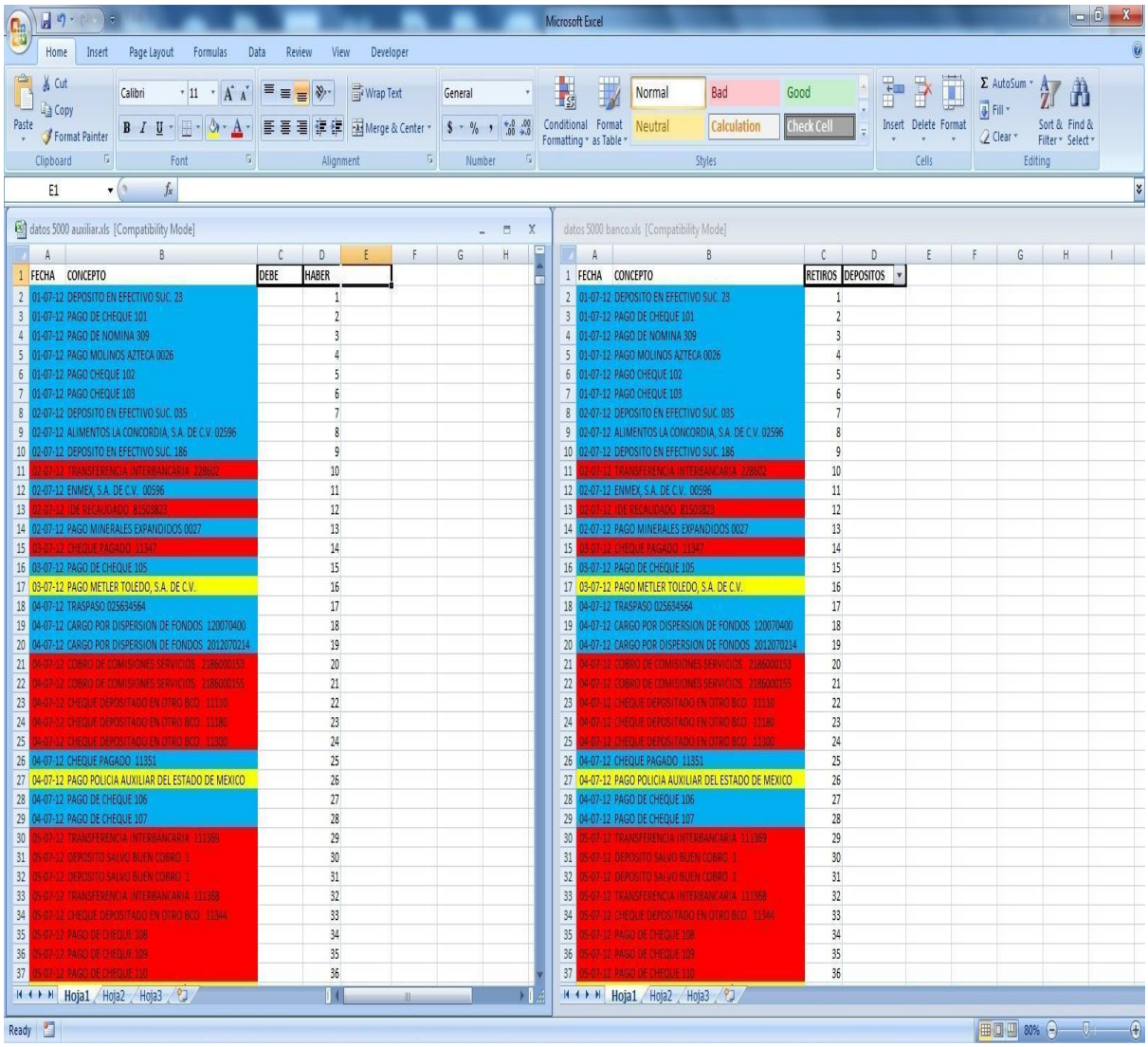


Imagen 5.0.2 Imagen en la que se aprecian los archivos de Excel que serán utilizados en la carga inicial.

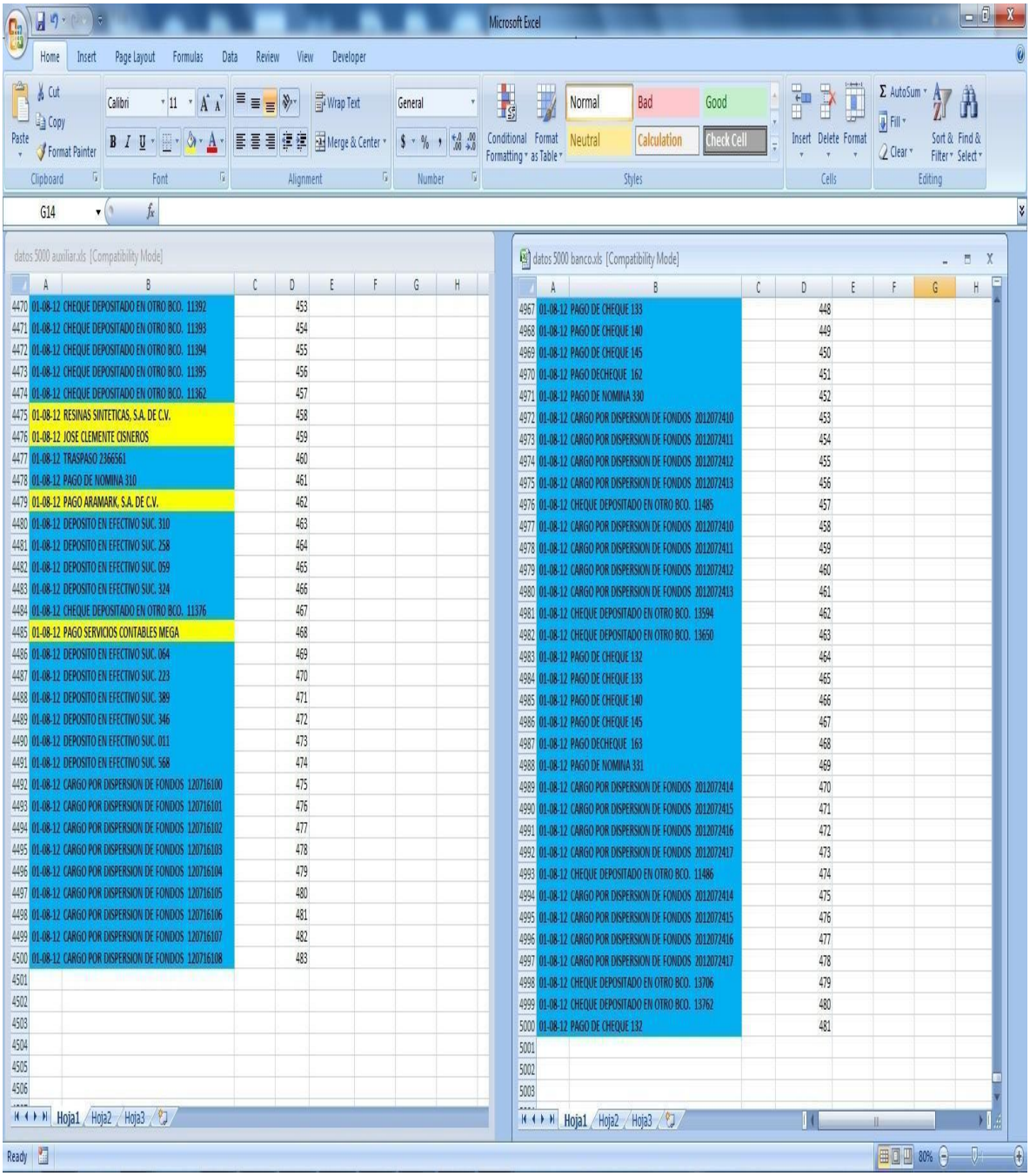


Imagen 5.0.3 Imagen en la que se aprecian ambos libros tanto auxiliar bancario (4500 registros) como estado de cuenta (5000 registros).

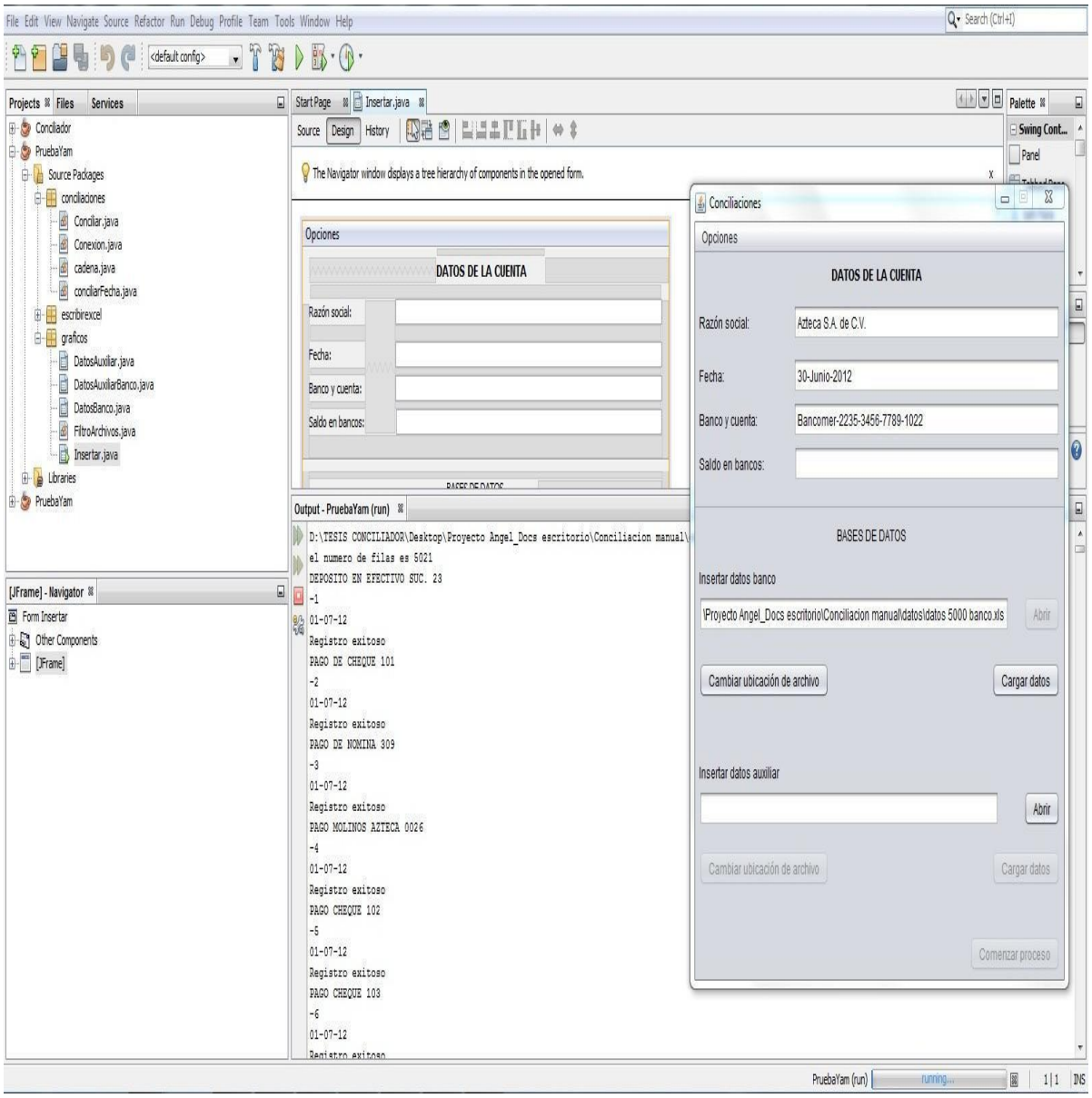


Imagen 5.0.4 Imagen en la que se puede apreciar desde la consola la carga inicial de datos del libro extracto bancario.

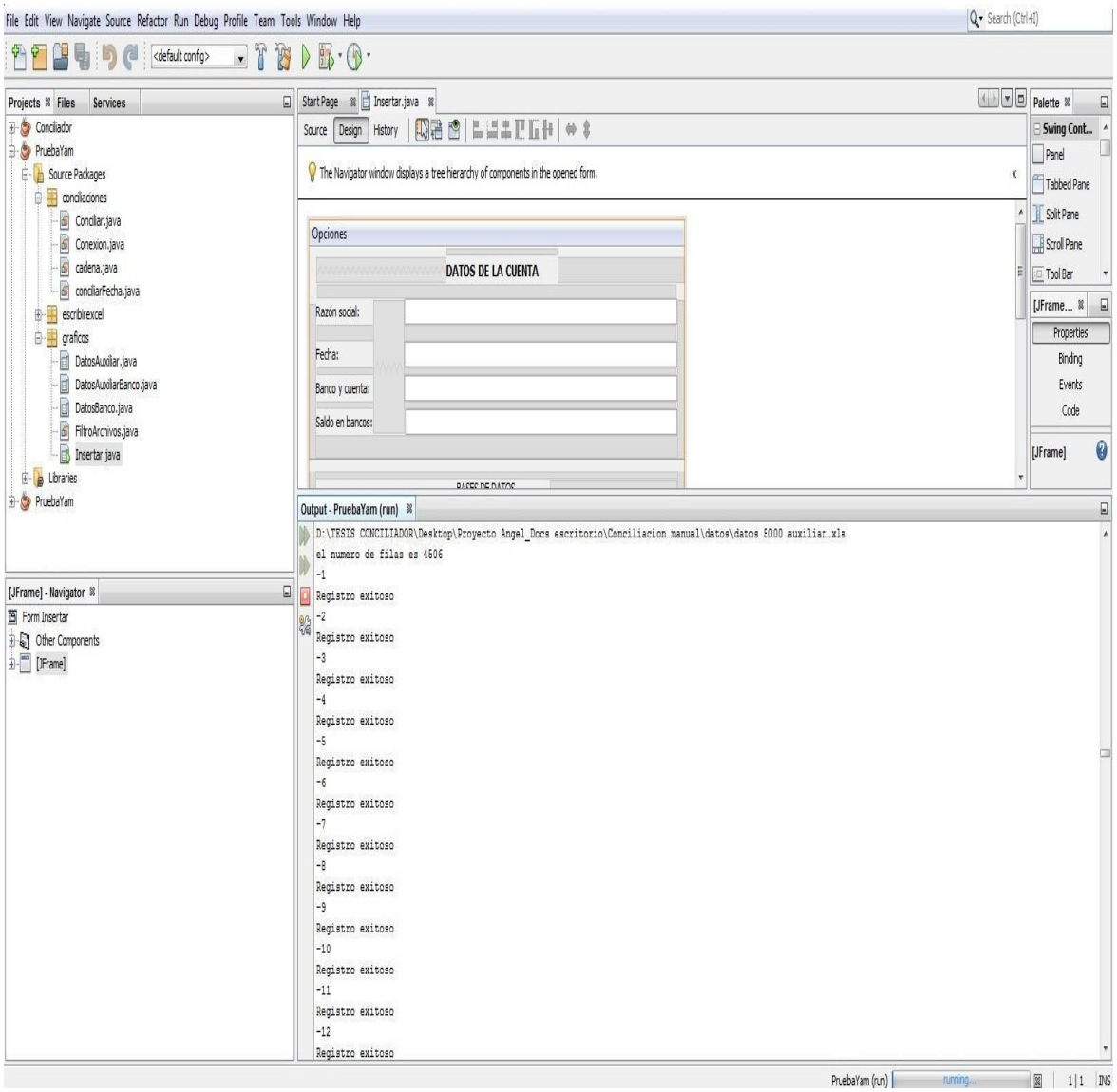


Imagen 5.0.5 Imagen que indica la carga inicial de datos extraída del libro auxiliar hacia la base de datos de la aplicación, en ella se puede apreciar registro por registro en la consola.

En la Imagen 5.0.6 se muestra un error en tiempo de ejecución debido a que durante el procesamiento el archivo de Visual Basic Script no pudo encontrar una ruta en la cual se encuentra un archivo de Excel que se utiliza para poder procesar datos.

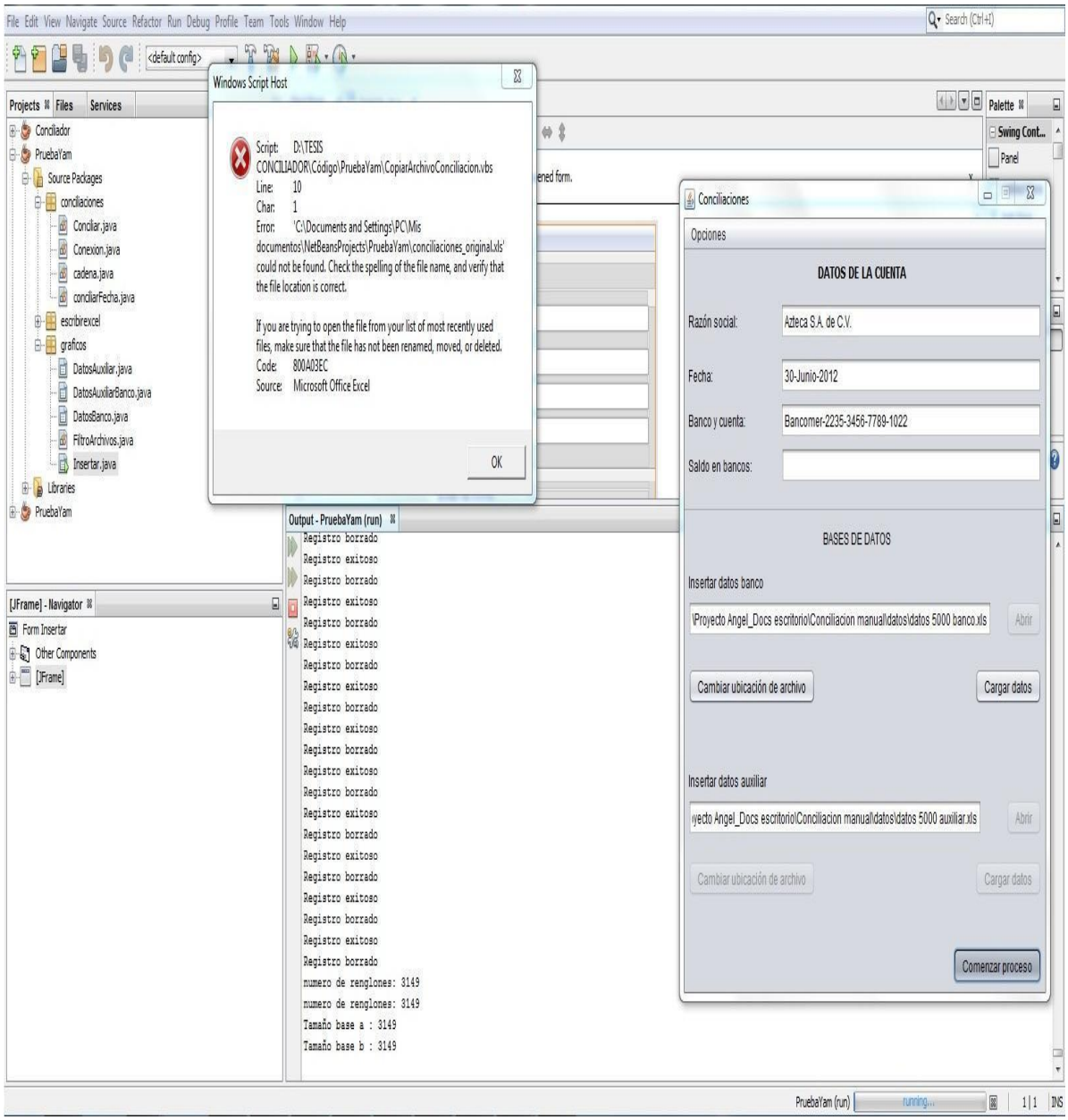


Imagen 5.0.6 Imagen que muestra el error en tiempo de ejecución.

En la imagen 5.0.7 se puede apreciar el reporte generado en la consola de Netbeans, la cual muestra el error de ejecución cuando el archivo de Visual Basic no pudo encontrar el archivo de Excel.

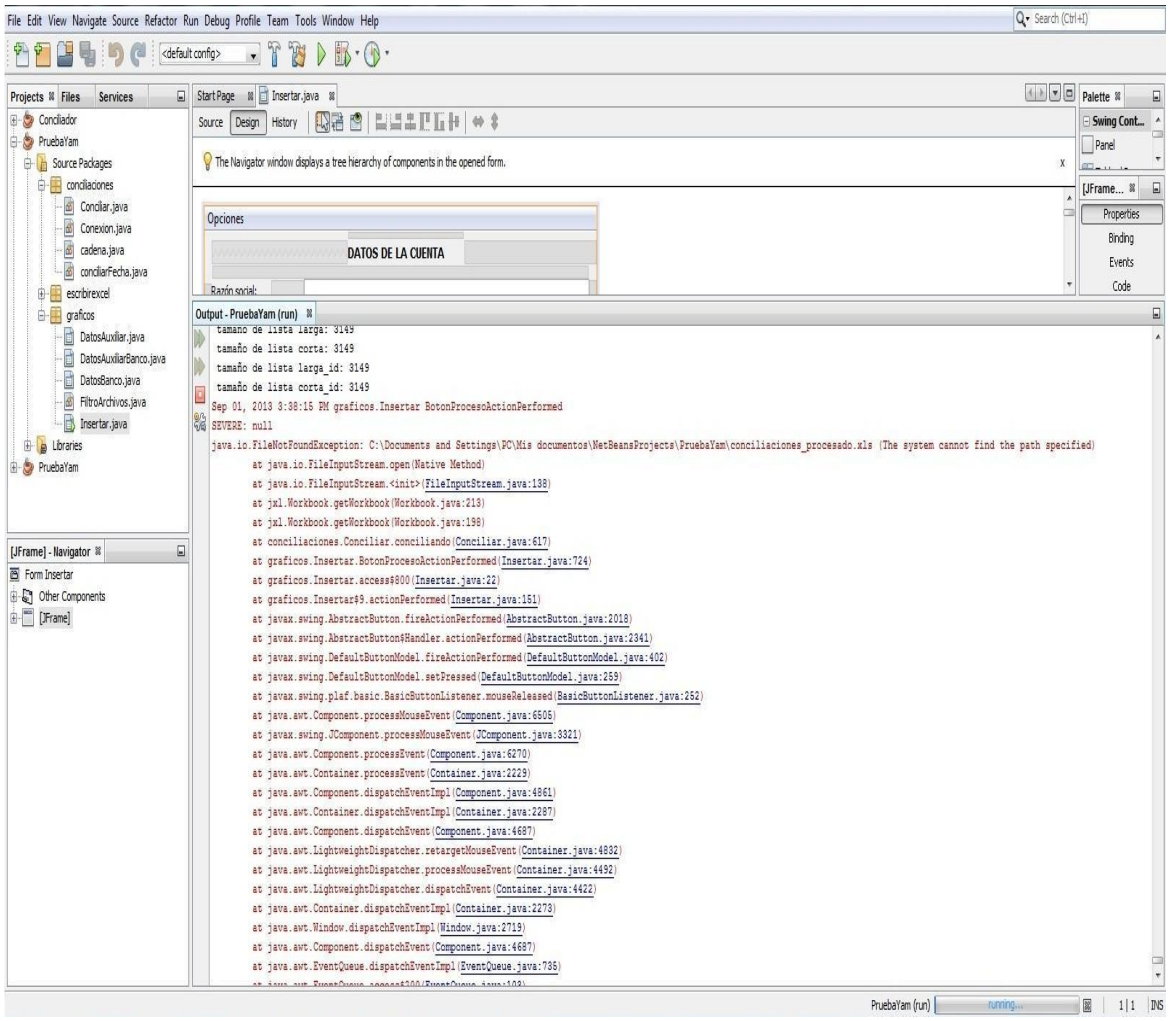


Imagen 5.0.7 Imagen que muestra en consola el error de ejecución.

En la Imagen 5.0.8 se muestran las bases de datos proporcionadas por el cliente con múltiples registros para poder evaluar el procesamiento de los datos con la aplicación diseñada hasta ese momento.

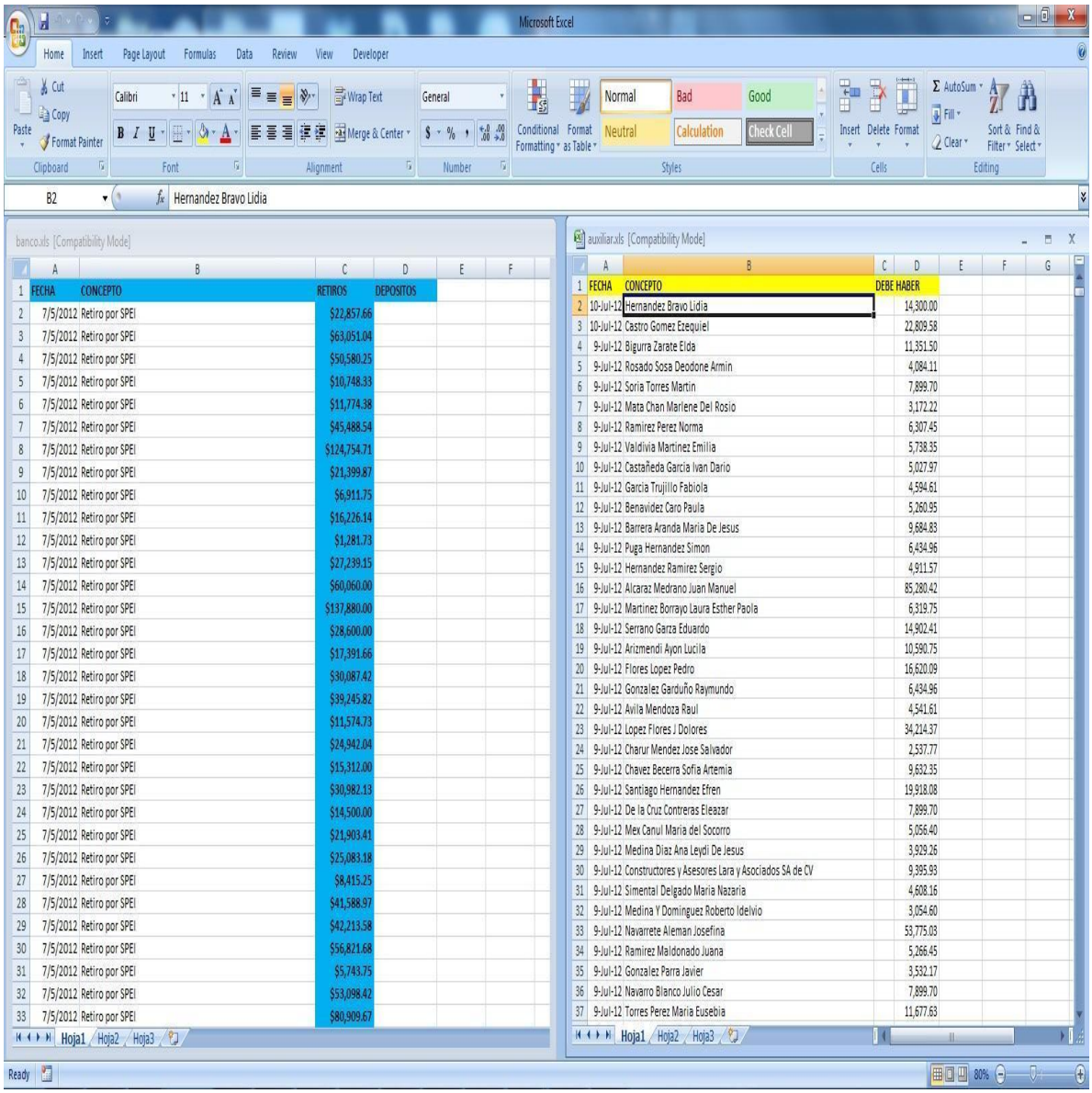


Imagen 5.0.8 Imagen que muestra ambas bases de datos proporcionadas por el cliente.

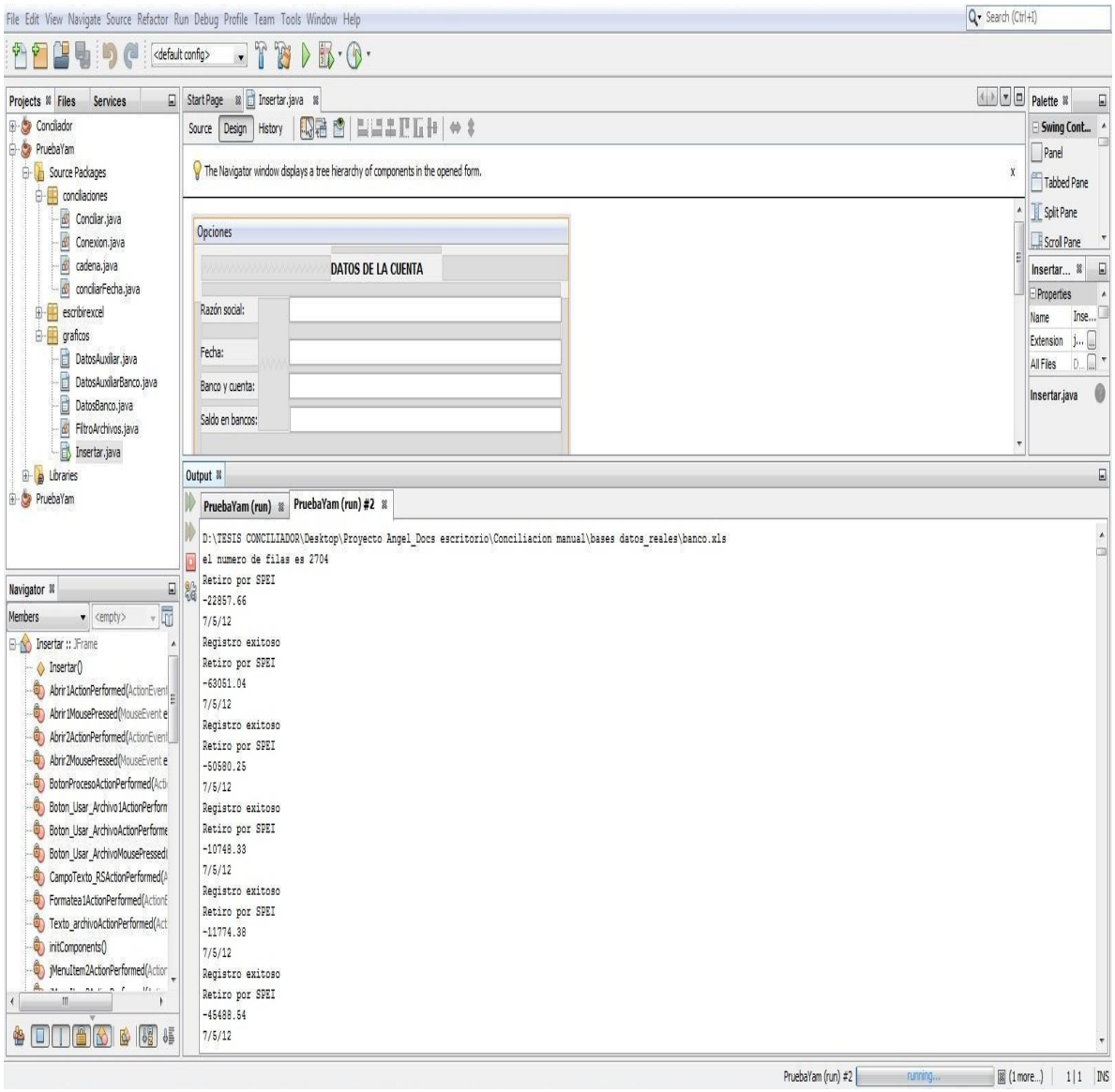


Imagen 5.0.9 Imagen que muestra la carga inicial del libro extracto bancario.

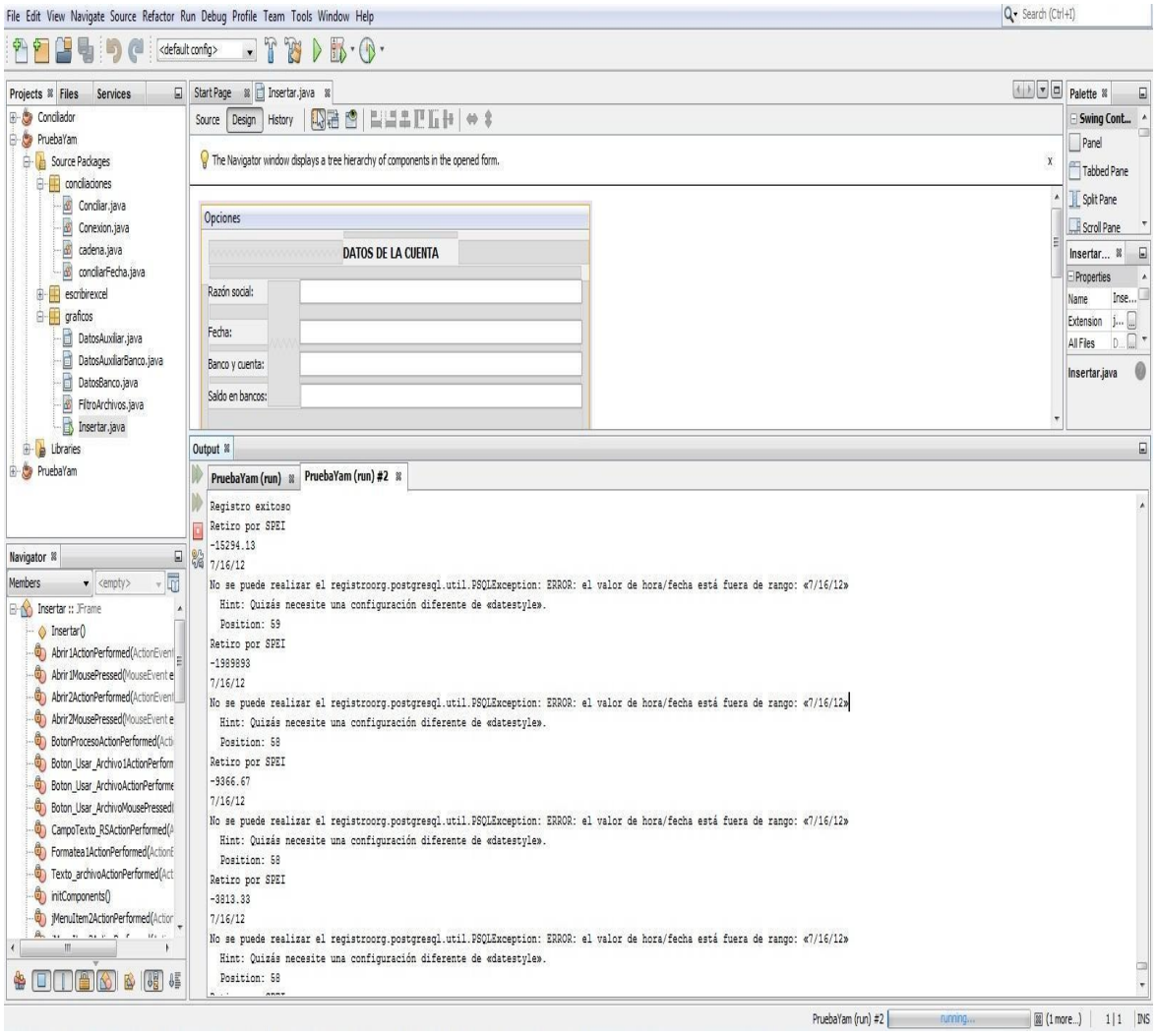


Imagen 5.0.10 Imagen que muestra excepciones al insertar datos del libro extracto bancario dentro de la base de datos.

En la imagen 5.0.11 se muestra la consola de Netbeans en la cual se puede apreciar que algunos registros del libro auxiliar no se pudieron insertar en la base de datos por el formato que presentan desde Excel.

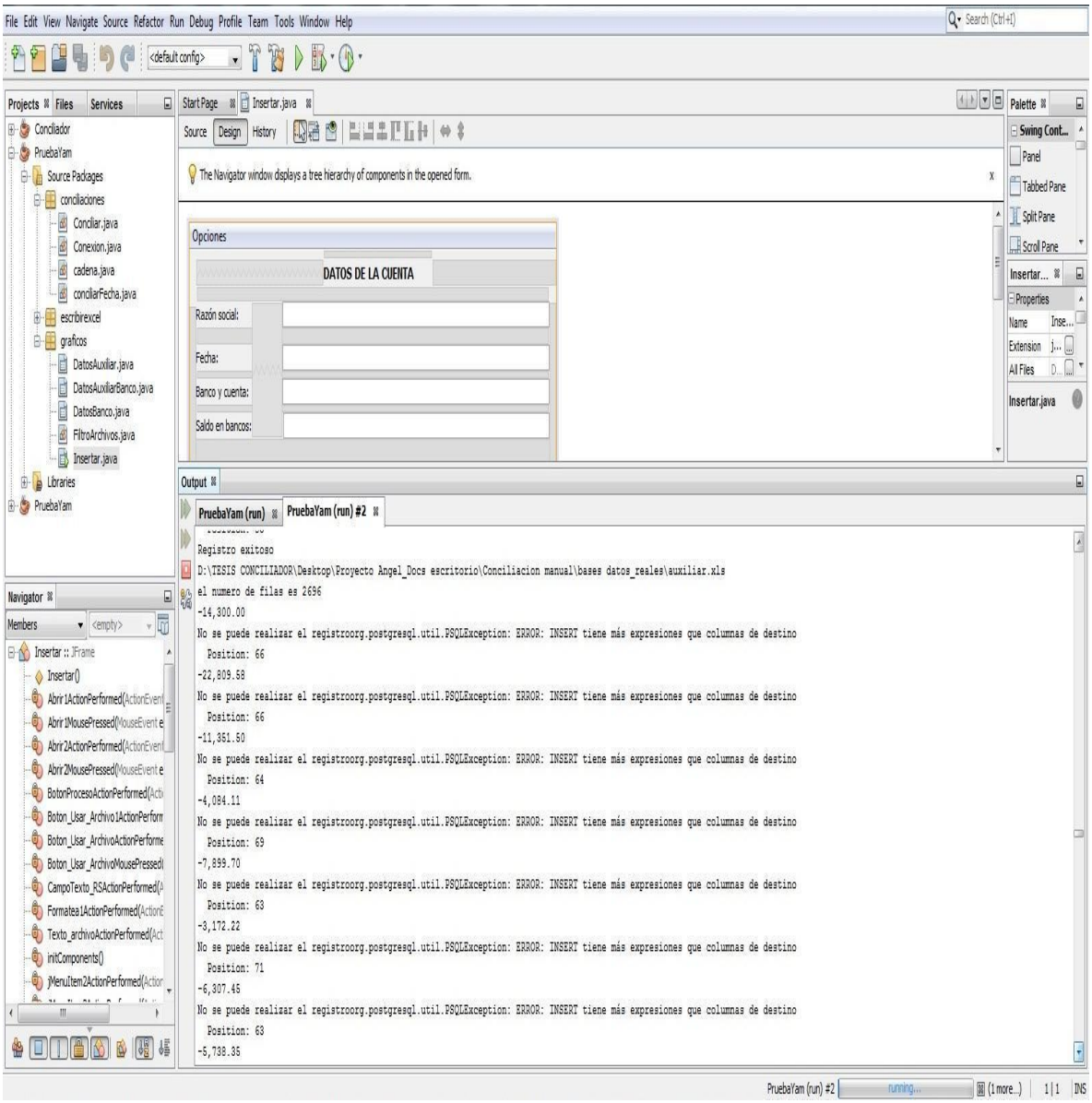


Imagen 5.0.11 Imagen que muestra el error al insertar los registros del libro auxiliar.

En la imagen 5.0.12 se muestra una prueba con datos más grandes para poder evaluar si el procesamiento de los mismos es el correcto.

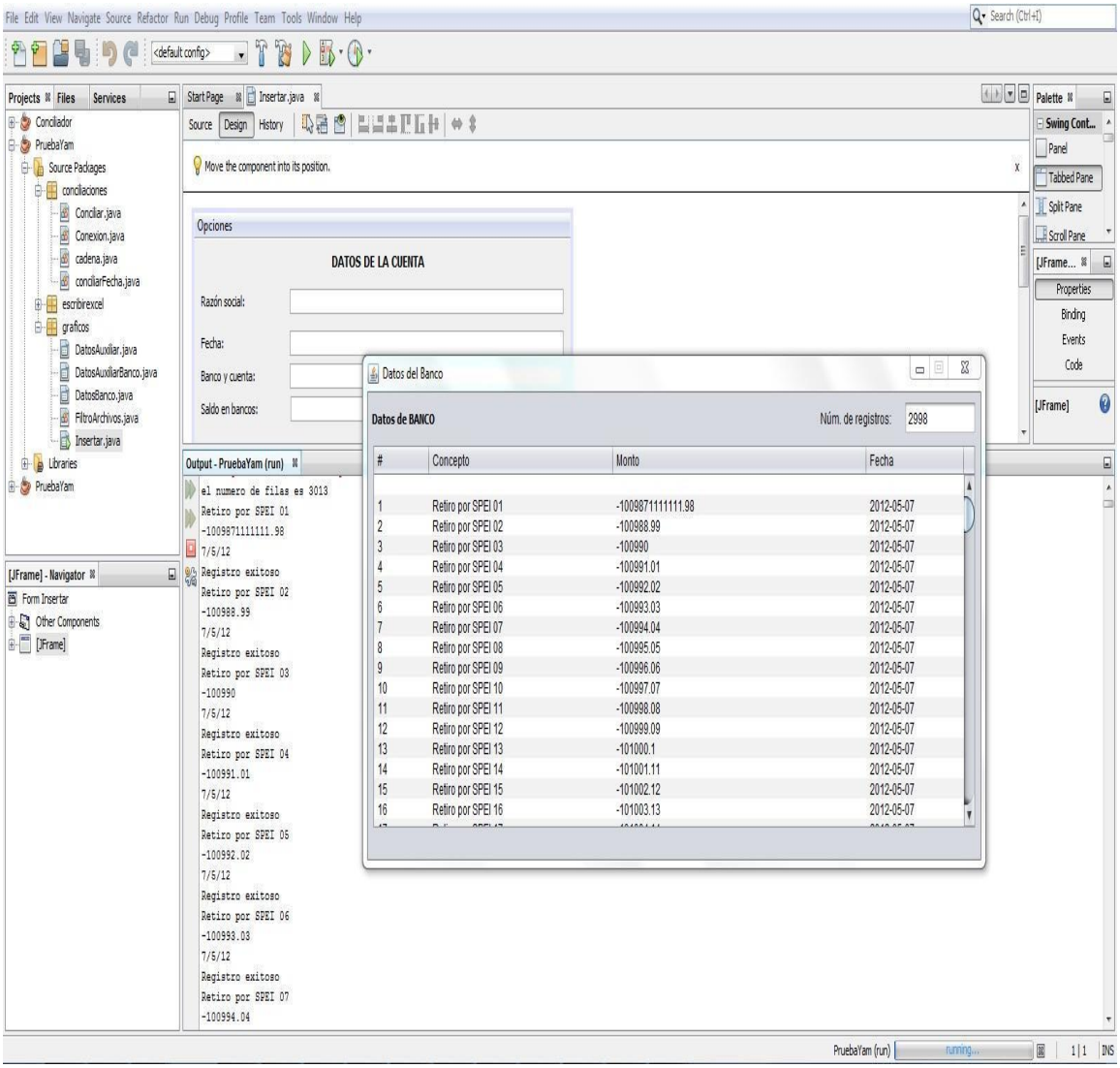


Imagen 5.0.12 Imagen que muestra la carga de datos del extracto bancario.

En la imagen 5.0.13 se pueden apreciar cantidades más grandes y con decimales, cabe resaltar que durante la inserción de datos con una longitud mayor a 4 cifras y con varios decimales fue la que más tiempo consumió durante las pruebas para estar seguros de que la cifra insertada era la correcta y así poder representar un reporte con las cantidades exactas.

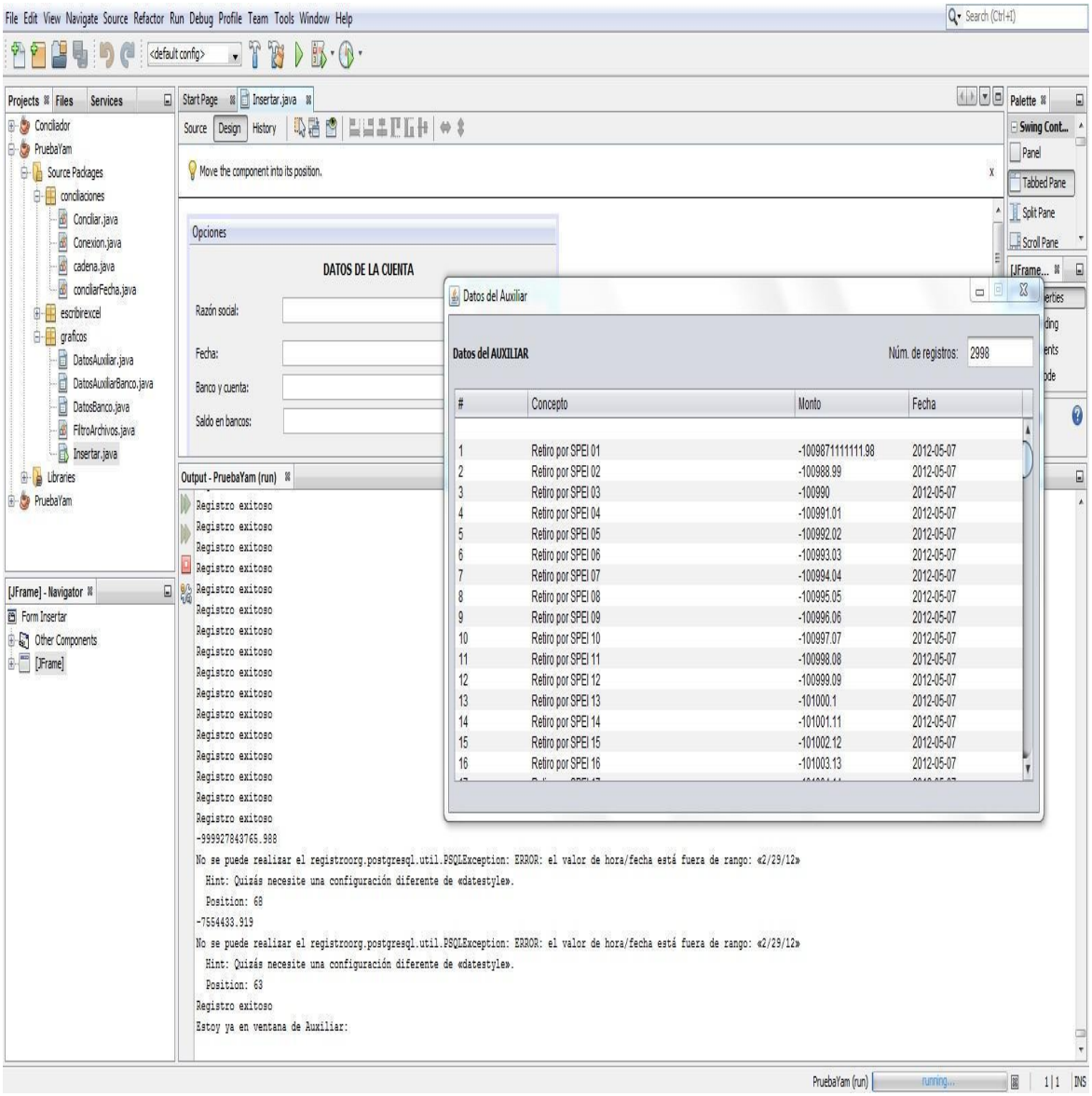


Imagen 5.0.13 La imagen muestra la carga de los datos del libro auxiliar con datos numéricos representados con muchos dígitos.

La imagen 5.0.14 muestra el procesamiento intermedio por el cual pasa la conciliación bancaria, en este ejemplo se muestra cuando las dos bases de datos han sido pobladas de Excel a la aplicación y se ha realizado el primer procesamiento conforme a las reglas del primer método, para ello se insertan los datos que faltan ser procesados en dos nuevas tablas y se continúa con el proceso.

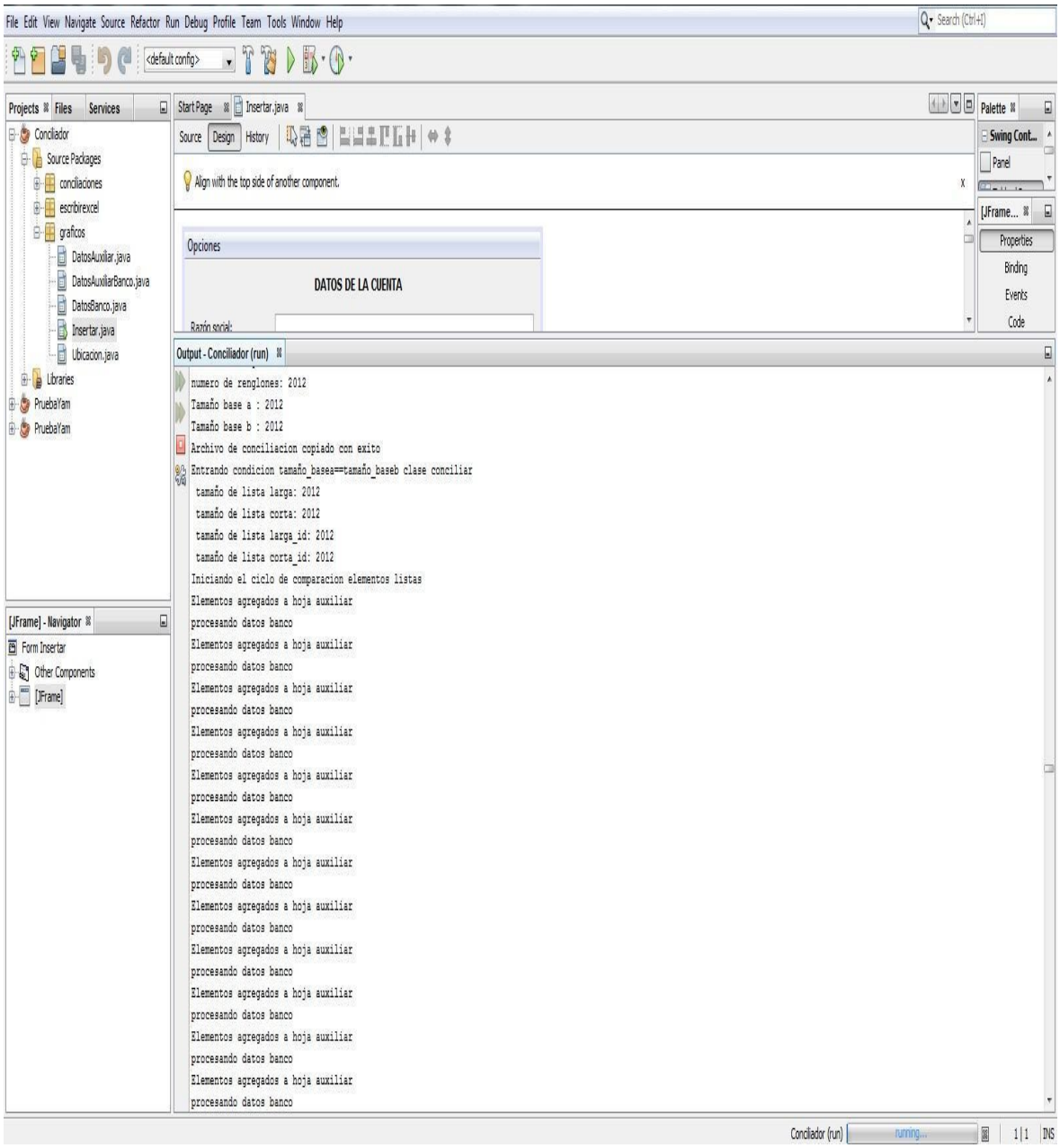


Imagen 5.0.14 Imagen que muestra en consola los registros del archivo auxiliar cuando han sido cargados a la base de datos.

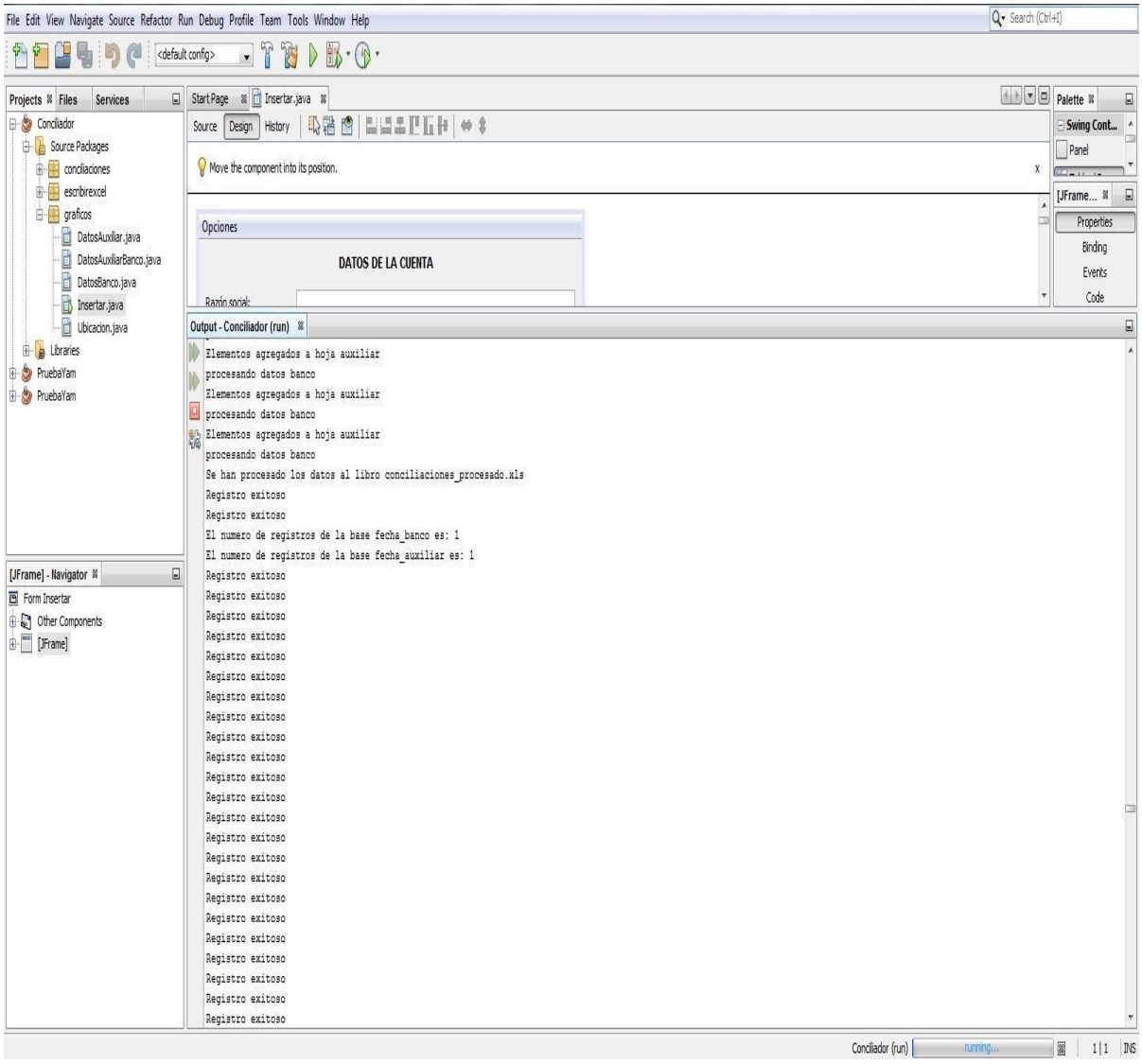


Imagen 5.0.15 Imagen que muestra el seguimiento del proceso de la conciliación.

En la imagen 5.0.16 se puede notar una prueba que se realizó para evaluar el número retornado con formato flotante.

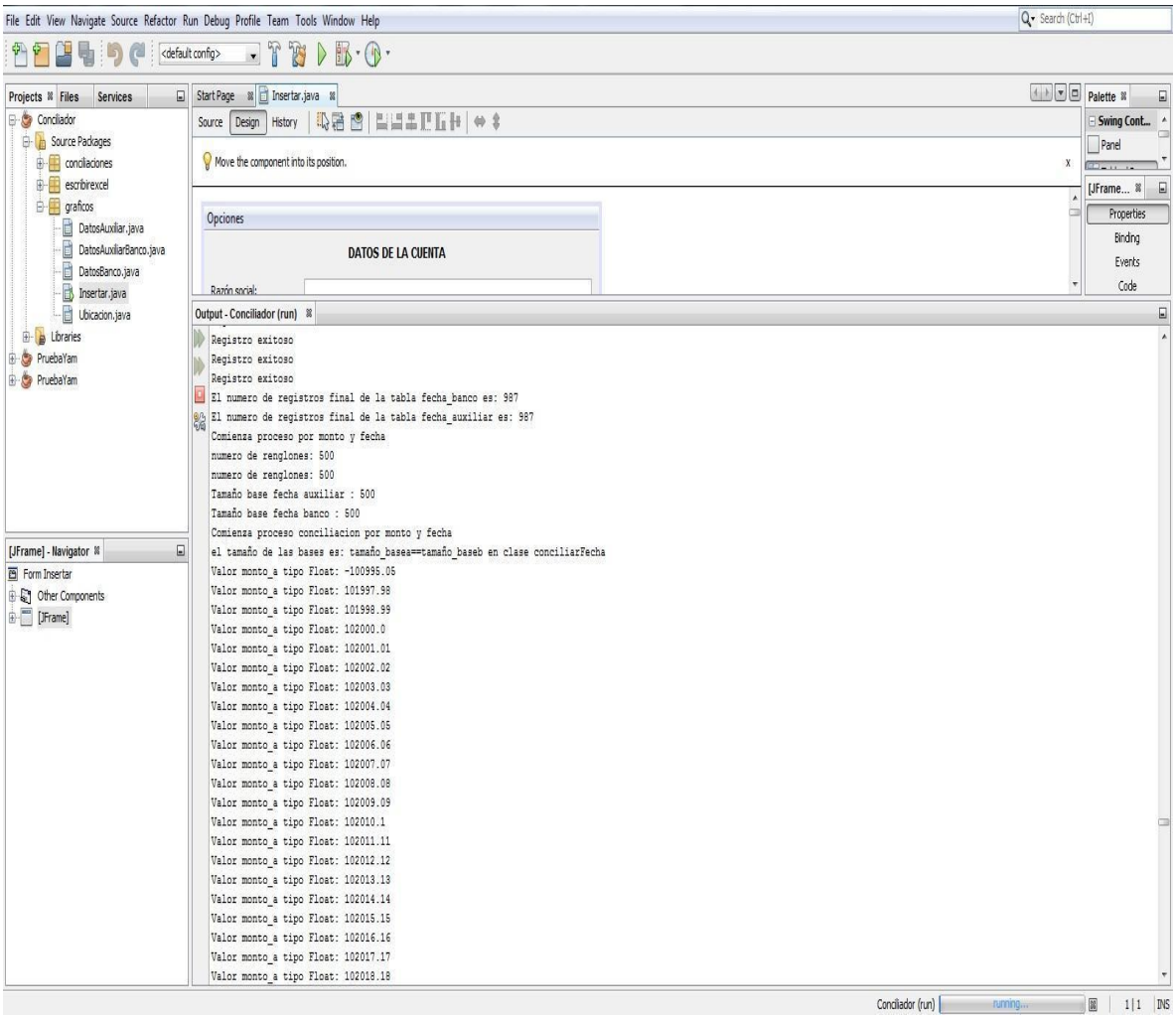


Imagen 5.0.16 Imagen que muestra el cambio para procesar los datos por monto y fecha.

En la imagen 5.0.17 se observa que el proceso sigue ejecutándose y se muestra un texto en la consola que nos permite evaluar cuáles son los datos que no pudieron conciliarse por monto y fecha, para poder conciliarlos por el tercer método.

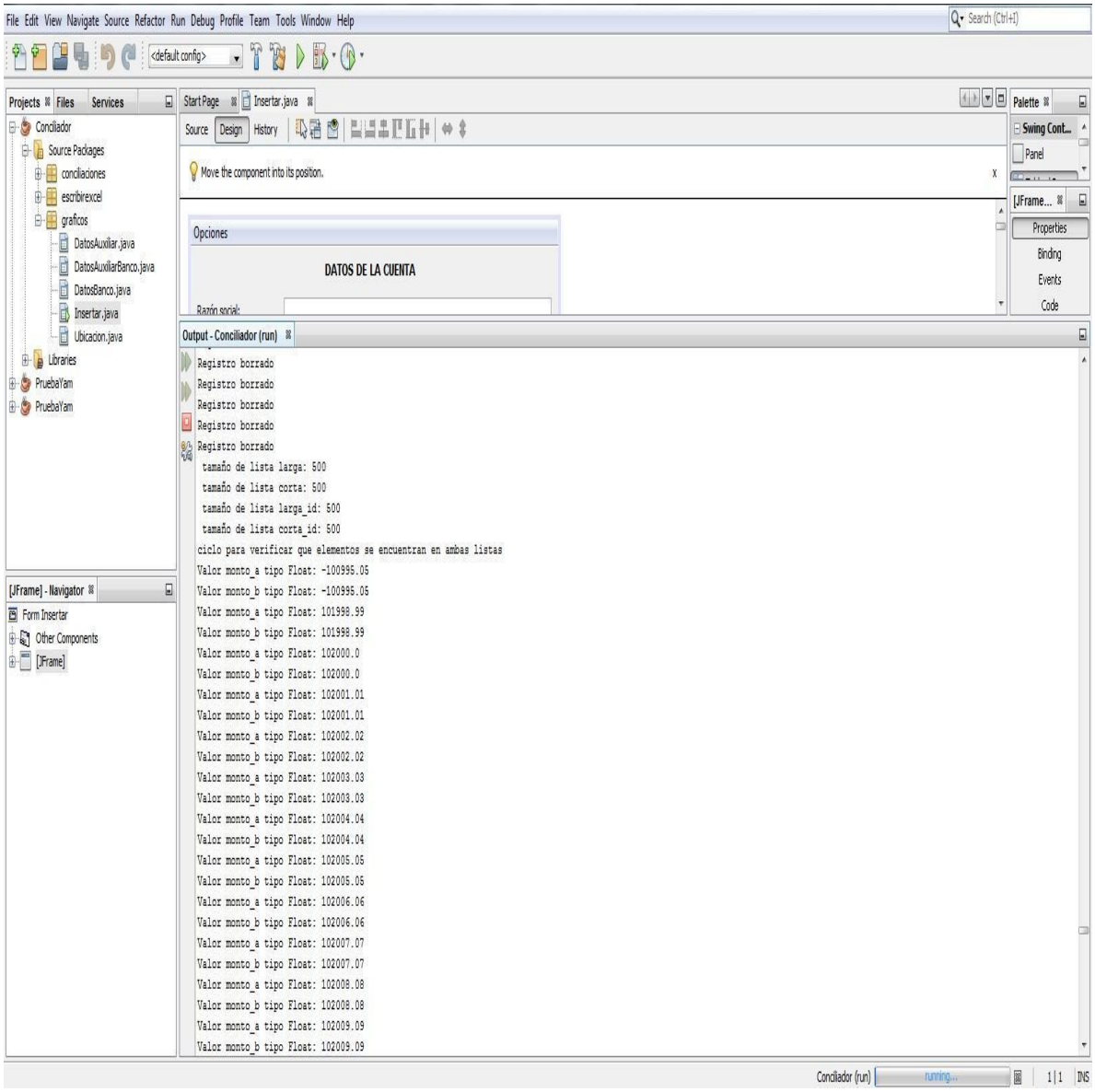


Imagen 5.0.17 Imagen que muestra los datos que no pudieron conciliarse por monto y fecha.

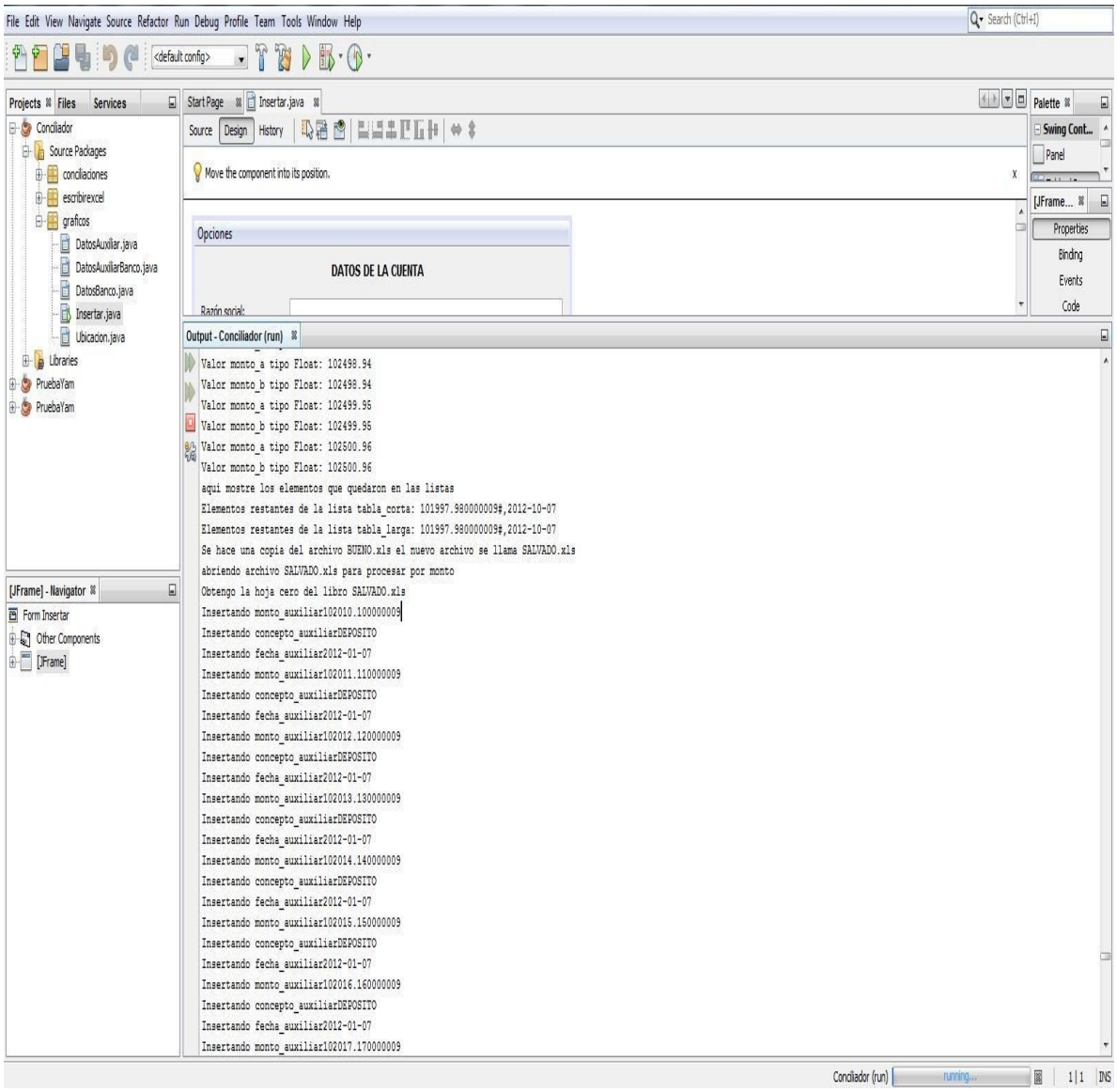


Imagen 5.0.18 Imagen que muestra los datos que empezaran a ser procesados por el tercer método (monto).

En la imagen 5.0.19 se muestra desde la consola de Netbeans cuando el proceso ha sido finalizado y en la interfaz gráfica ya se ha habilitado el botón de “ver resultados”, lo cual permite observar el reporte final desde Excel.

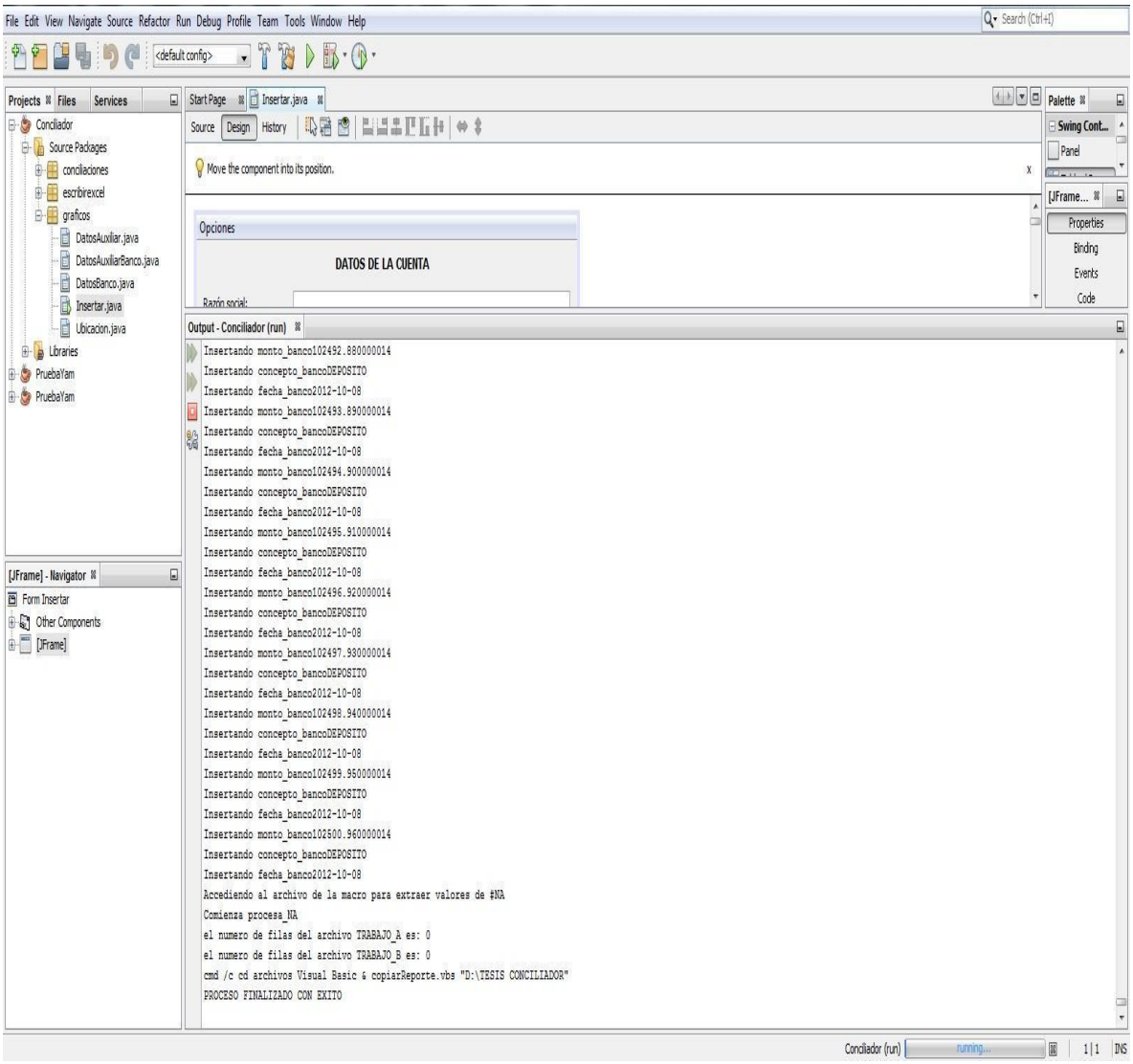
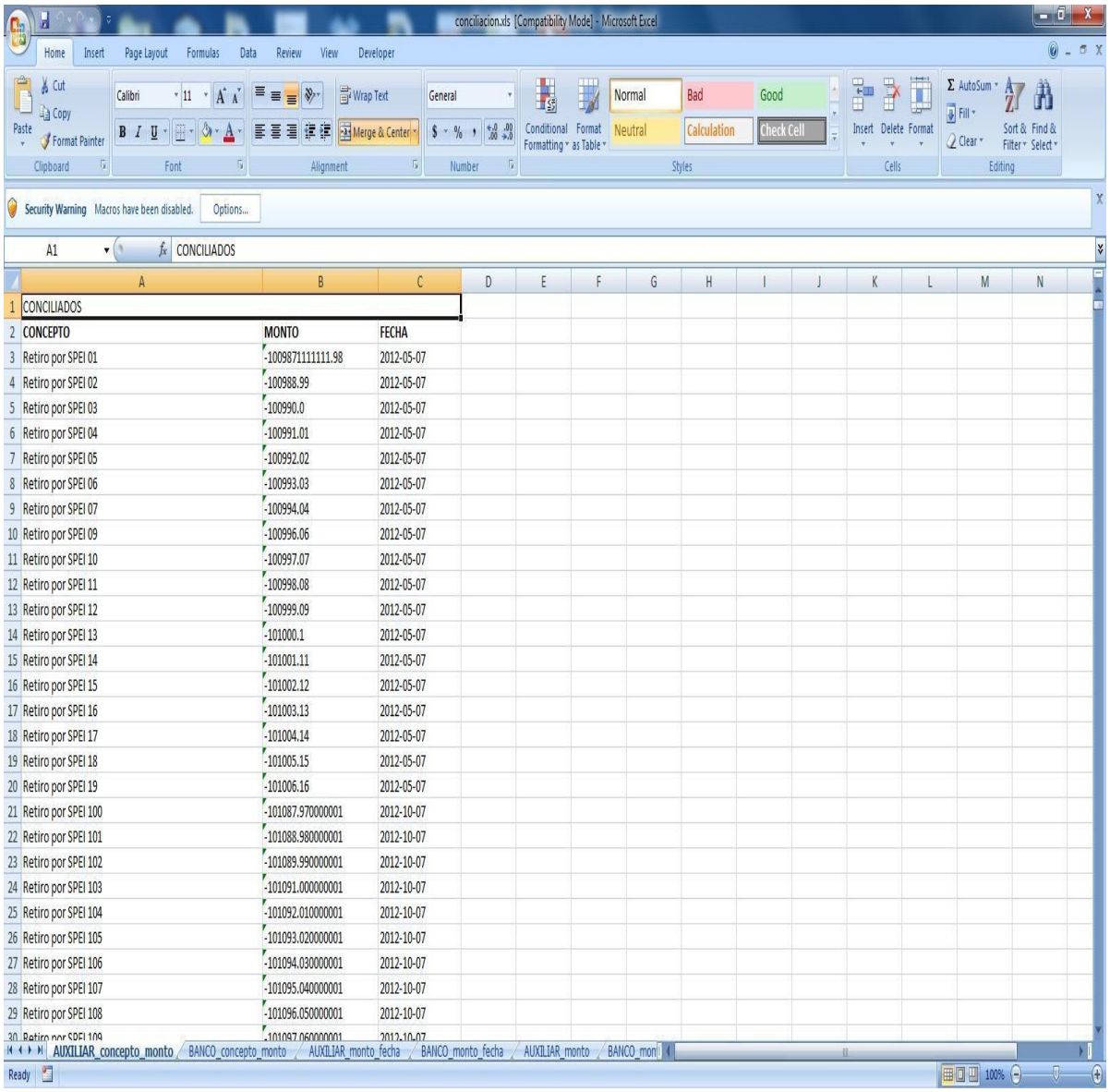


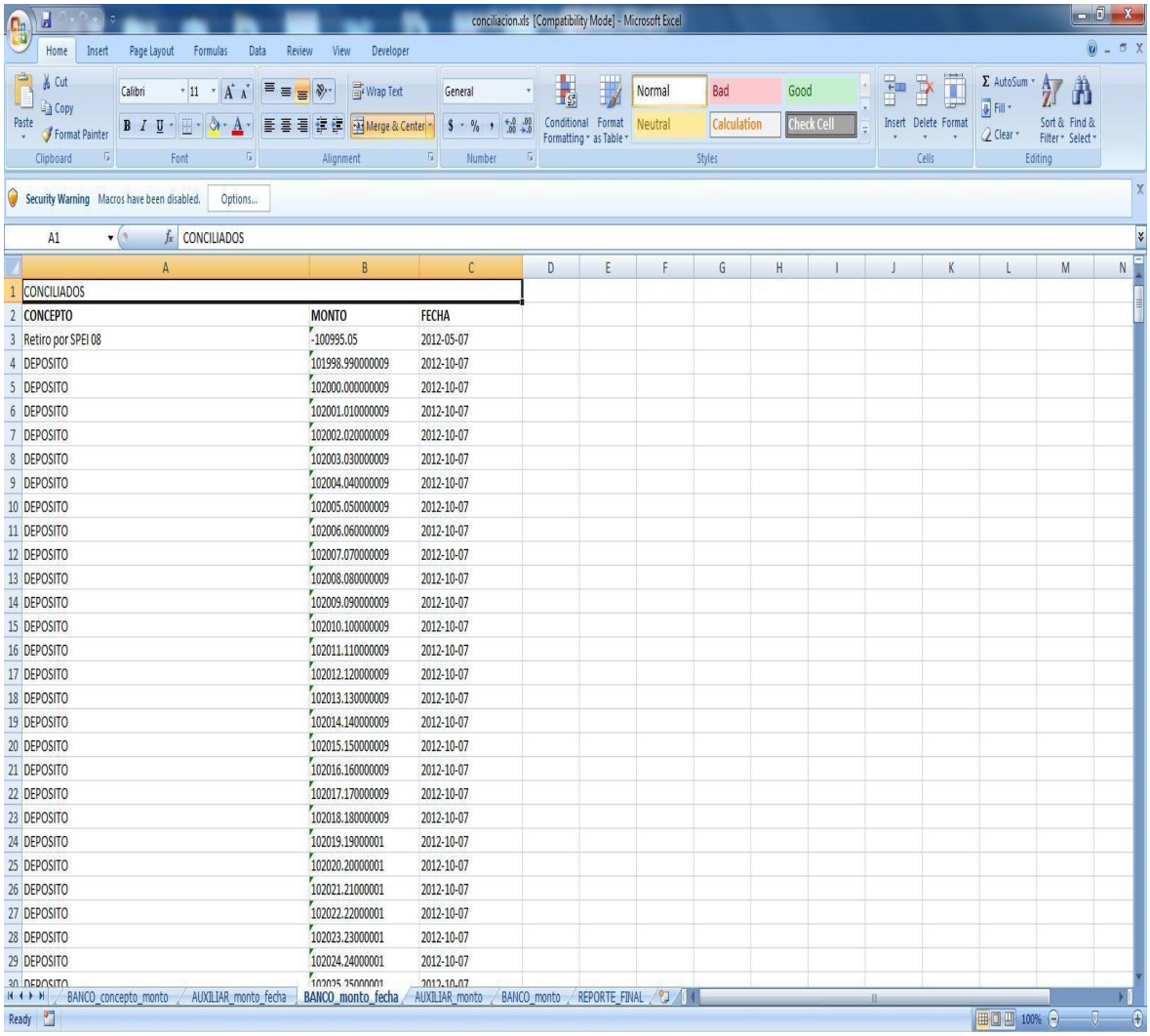
Imagen 5.0.19 Imagen que muestra desde la consola de Netbeans cuando el proceso ha finalizado.

En la Imagen 5.0.20 se muestra el archivo de Excel con formato para pruebas, debido a que imprimíamos todos los datos que habían sido procesados por cada método para poder compararlos con los resultados que nosotros habíamos obtenido tras la conciliación manual.



	A	B	C
1	CONCILIADOS		
2	CONCEPTO	MONTO	FECHA
3	Retiro por SPEI 01	-1009871111111.98	2012-05-07
4	Retiro por SPEI 02	-100988.99	2012-05-07
5	Retiro por SPEI 03	-100990.0	2012-05-07
6	Retiro por SPEI 04	-100991.01	2012-05-07
7	Retiro por SPEI 05	-100992.02	2012-05-07
8	Retiro por SPEI 06	-100993.03	2012-05-07
9	Retiro por SPEI 07	-100994.04	2012-05-07
10	Retiro por SPEI 09	-100996.06	2012-05-07
11	Retiro por SPEI 10	-100997.07	2012-05-07
12	Retiro por SPEI 11	-100998.08	2012-05-07
13	Retiro por SPEI 12	-100999.09	2012-05-07
14	Retiro por SPEI 13	-101000.1	2012-05-07
15	Retiro por SPEI 14	-101001.11	2012-05-07
16	Retiro por SPEI 15	-101002.12	2012-05-07
17	Retiro por SPEI 16	-101003.13	2012-05-07
18	Retiro por SPEI 17	-101004.14	2012-05-07
19	Retiro por SPEI 18	-101005.15	2012-05-07
20	Retiro por SPEI 19	-101006.16	2012-05-07
21	Retiro por SPEI 100	-101087.970000001	2012-10-07
22	Retiro por SPEI 101	-101088.980000001	2012-10-07
23	Retiro por SPEI 102	-101089.990000001	2012-10-07
24	Retiro por SPEI 103	-101091.000000001	2012-10-07
25	Retiro por SPEI 104	-101092.010000001	2012-10-07
26	Retiro por SPEI 105	-101093.020000001	2012-10-07
27	Retiro por SPEI 106	-101094.030000001	2012-10-07
28	Retiro por SPEI 107	-101095.040000001	2012-10-07
29	Retiro por SPEI 108	-101096.050000001	2012-10-07

Imagen 5.0.20 Imagen que muestra el archivo de Excel Auxiliar_concepto_monto con formato para pruebas.



	A	B	C
1	CONCILIADOS		
2	CONCEPTO	MONTO	FECHA
3	Retiro por SPEI 08	-100995.05	2012-05-07
4	DEPOSITO	101998.990000009	2012-10-07
5	DEPOSITO	102000.000000009	2012-10-07
6	DEPOSITO	102001.010000009	2012-10-07
7	DEPOSITO	102002.020000009	2012-10-07
8	DEPOSITO	102003.030000009	2012-10-07
9	DEPOSITO	102004.040000009	2012-10-07
10	DEPOSITO	102005.050000009	2012-10-07
11	DEPOSITO	102006.060000009	2012-10-07
12	DEPOSITO	102007.070000009	2012-10-07
13	DEPOSITO	102008.080000009	2012-10-07
14	DEPOSITO	102009.090000009	2012-10-07
15	DEPOSITO	102010.100000009	2012-10-07
16	DEPOSITO	102011.110000009	2012-10-07
17	DEPOSITO	102012.120000009	2012-10-07
18	DEPOSITO	102013.130000009	2012-10-07
19	DEPOSITO	102014.140000009	2012-10-07
20	DEPOSITO	102015.150000009	2012-10-07
21	DEPOSITO	102016.160000009	2012-10-07
22	DEPOSITO	102017.170000009	2012-10-07
23	DEPOSITO	102018.180000009	2012-10-07
24	DEPOSITO	102019.19000001	2012-10-07
25	DEPOSITO	102020.20000001	2012-10-07
26	DEPOSITO	102021.21000001	2012-10-07
27	DEPOSITO	102022.22000001	2012-10-07
28	DEPOSITO	102023.23000001	2012-10-07
29	DEPOSITO	102024.24000001	2012-10-07

Imagen 5.0.21 Imagen que muestra el libro de Excel BANCO_monto_fecha.

Durante el procesamiento de los datos en diversas ocasiones surgieron errores que podían irse presentando por formatos, por rutas que no eran encontradas por los scripts, o algunas veces por la cantidad de datos procesados.

En la imagen 5.0.22 se muestra un error en tiempo de ejecución el cual muchas veces fue resuelto mediante un debug dentro de Visual Basic y analizando la traza del proceso mostrado en la consola del Netbeans.

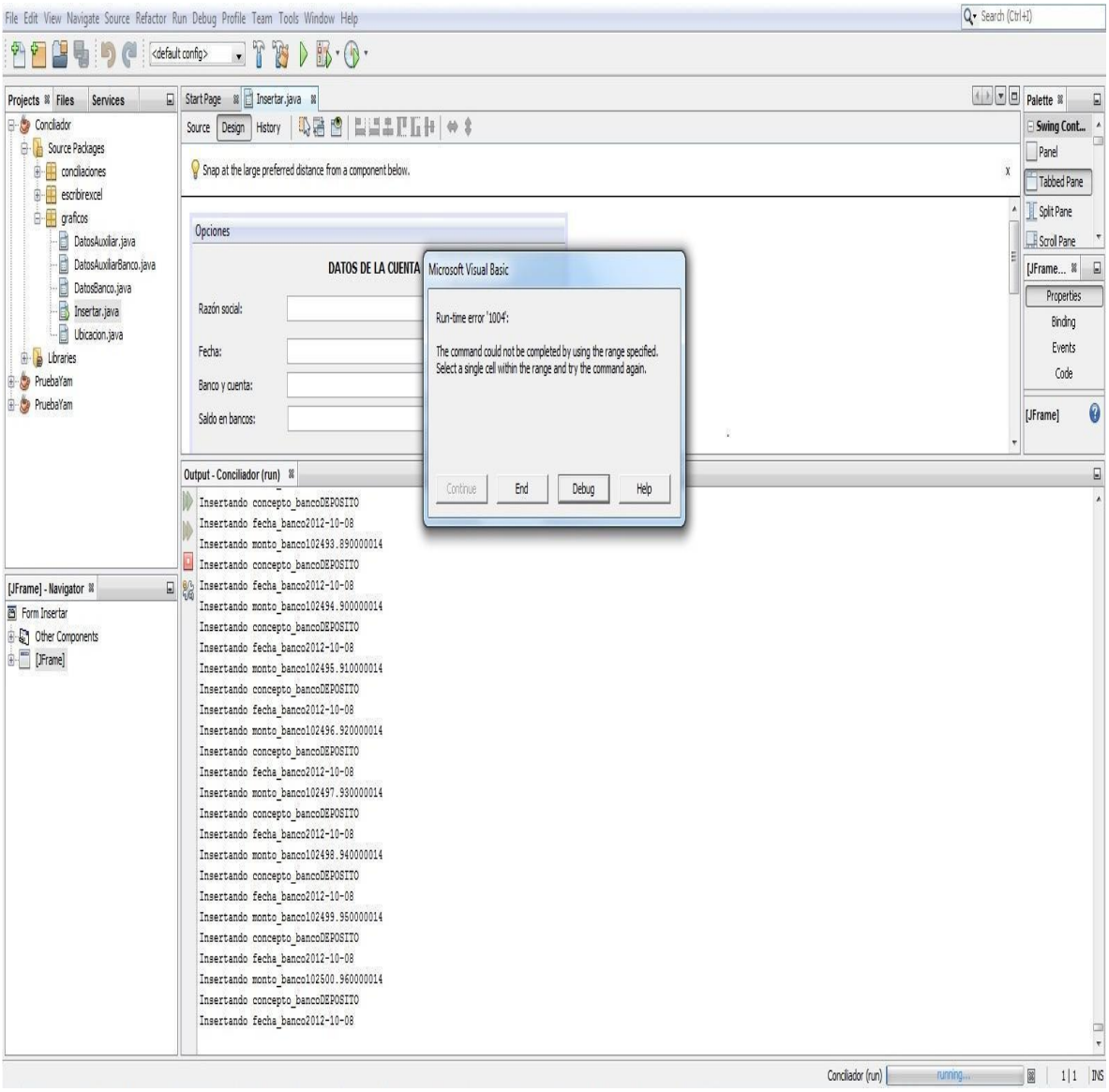


Imagen 5.0.22 Imagen que muestra un error en tiempo de ejecución.

En la imagen 5.0.23 se muestra un debug realizado por medio de Visual Basic, en el cual se realizaba un seguimiento para poder determinar cuál fue el error.

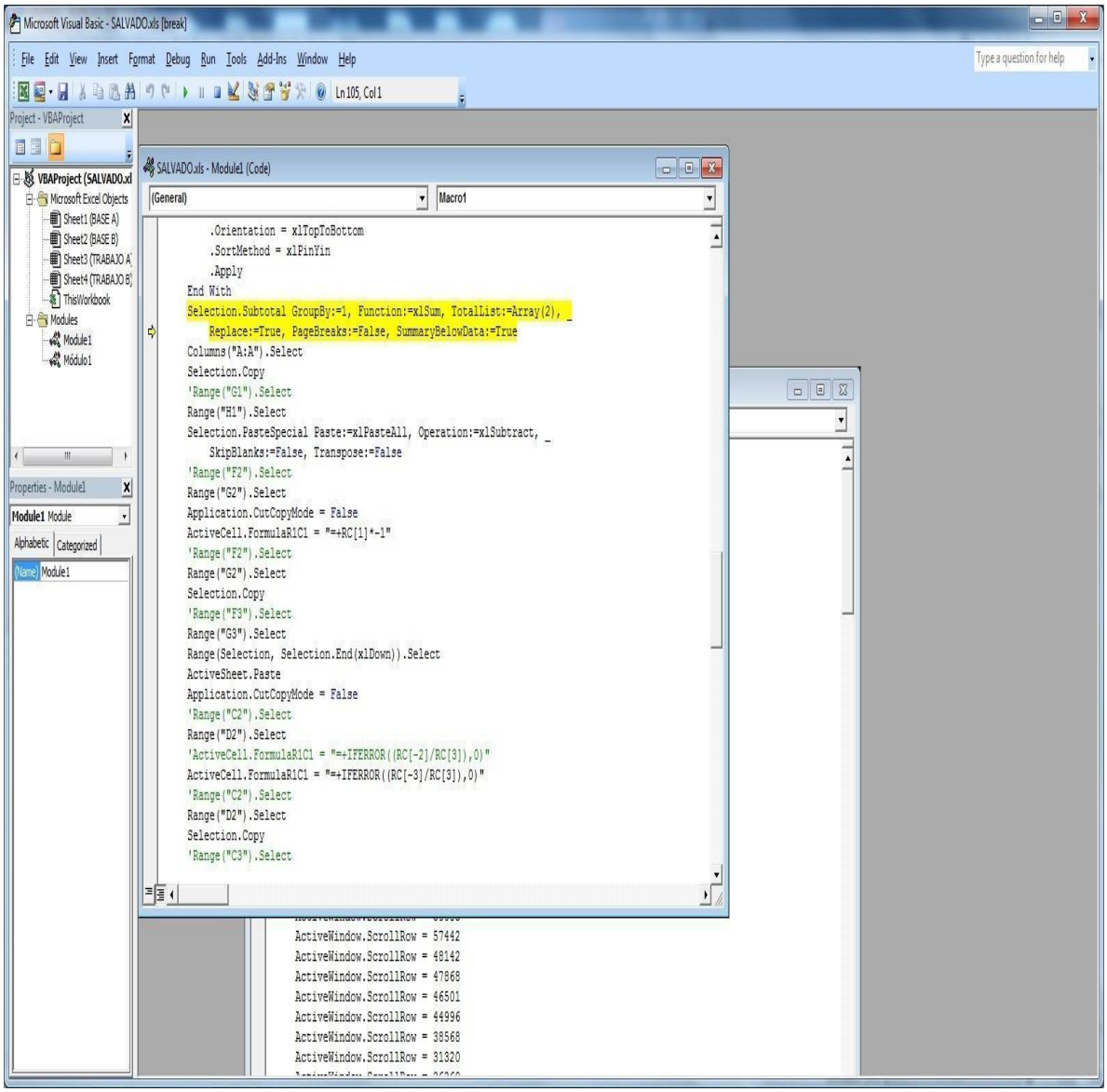


Imagen 5.0.23 Imagen que muestra las líneas de código dentro de Visual Basic donde el programa se había detenido.

6.0 Conclusiones

Actualmente el desarrollo y competencias empresariales fomentan la necesidad de adquirir habilidades, innovar productos y procesos y a su vez adaptarse al uso de nuevas herramientas tecnológicas capaces de favorecer el manejo eficiente y eficaz de los recursos que posee la empresa.

Por su lado, la tecnología para el desarrollo de sistemas ha evolucionado de tal manera que disponemos de la facilidad de desarrollar aplicaciones que demandan mayor complejidad y que logren cumplir con los requerimientos del cliente, tales como facilidad de uso, rapidez en el procesamiento y obtención de resultados, seguridad, entre otros.
Ahora, se abren las posibilidades para automatizar actividades que en la actualidad involucran un mayor esfuerzo y por tanto una mayor inversión en los recursos.

Debido a la interacción que tuvimos con el cliente se detectaron mejoras, así como características que en un principio no se habían contemplado ni visualizado, impactando en los tiempos de análisis y desarrollo, sin embargo gracias a esto se logró adecuar los avances con los requerimientos que iban surgiendo.

Un aspecto importante que se completó con éxito durante la fase de desarrollo surgió cuando el cliente planteo la idea de cargar toda la información de ambos libros a través de archivos de Excel, reduciendo el esfuerzo que implicaría el subir la información registro por registro, tal como se había planteado inicialmente y por consecuencia eliminando la posibilidad de capturar de forma errónea la información, como puede ocurrir con otras aplicaciones que ya existen en el mercado.

El usuario final tiene acceso a la aplicación y le da uso sin mayor complicación ya que se logró brindar una interfaz gráfica amigable, la cual es capaz de desplegar la información almacenada. En cuanto a los resultados que se obtienen, los usuarios pueden visualizar todos los registros que no pudieron ser conciliados a través de un archivo Excel en donde se indica si los registros provienen del libro auxiliar o del estado de cuenta facilitando al usuario el posterior análisis de todos estos registros. Brinda seguridad de la información al no requerir el acceso a la red por ser esta una aplicación de escritorio. Se tienen estas características tal como se especificó en los requerimientos.

La relevancia que tiene esta tesis radica en que es posible llevar a cabo el desarrollo de nuevas aplicaciones capaces de fomentar el uso de la automatización de procesos de uso habitual, no sólo en el ámbito empresarial, sino a la medida de las necesidades que surgen en la vida cotidiana. Esto abre un abanico de oportunidades, ya que en muchas empresas hoy en día existen procesos que se llevan a cabo de manera manual o algunos ya están automatizados, pero se pueden mejorar esto con los nuevos avances tecnológicos y conforme a las nuevas necesidades de las empresas.

De aquí que la propuesta de automatizar el proceso de las conciliaciones bancarias a través de la aplicación cumple satisfactoriamente con los resultados esperados ya que demostró ser capaz de reducir esencialmente el tiempo de ejecución, entre otros recursos, requeridos para realizar esta operación.

Gracias a este proyecto obtuvimos experiencia, conocimos un proceso contable que es llevado a cabo por la mayoría de las empresas y lo más importante fue el haber logrado concluir con el entregable, pese a todos los obstáculos que surgieron a lo largo de la implementación, los cuales los atacamos como equipo aplicando conocimientos, disciplina y orientándonos a los resultados junto con la retroalimentación por parte del cliente.

7.0 Bibliografía y mesografía

Sistemas Computacionales Avanzados SQL 2a. edición, CCPM, editorial McGraw-Hill
Páginas: 3 - 5.

Programación en Office, CCPM, editorial Limusa, 2006.
Páginas: 2 - 9.

Desarrollo de sistemas, CCPM, editorial Pearson Educación, México, 2006.
Páginas: 2 - 14.

Introducción a la Contabilidad, autor Kenneth W. Perry, editorial McGraw-Hill
Páginas: 158.

Contabilidad 2, autor Sergio Wats, Instituto Politécnico Nacional 2005
Páginas: 160.

Sistemas operativos y lenguajes de programación
Enrique Quero Catalinas

Estructuras de Datos y Algoritmos, Hernández, R.; Lázaro, J. C.; Dormido, R.; Ros, S., editorial PRENTICE HALL
Páginas: 40 - 43.

JAVA Cómo programar séptima edición
Páginas 19, 20, 379, 418

Procesamiento de bases de datos: fundamentos, diseño e implementación.
Octava edición
David M. Kroenke
Editorial Pearson Prentice Hall

Programador Certificado Java 2. Curso práctico
Tercera edición
Antonio J. Martín Sierra
Editorial RA-MA S.A. Editorial y Publicaciones

Contabilidad Gerencial. Fundamentos, principios e introducción a la contabilidad
Ismael Granados, Leovigildo Atorre y Elbar Ramírez
Universidad Nacional de Colombia

http://www.fdi.ucm.es/profesor/lgarmend/arcgisjava/temas/Tema%202%20El%20lenguaje%20Java.pdf

http://www.unav.es/SI/manuales/Java/indice.html#1.1.

http://www.cad.com.mx/historia_del_lenguaje_java.htm

http://books.google.com.mx/books?id=buM5rlZME-cC&pg=PA20&dq=manejador+de+base+de+datos&hl=es&sa=X&ei=idv4UJfrOKLc2AW75oHIDA&ved=0CCsQ6AEwAA#v=onepage&q=manejador%20de%20base%20de%20datos&f=false

http://www.conciliacionbancaria.com/

http://books.google.com.mx/books?id=ZFYSrjK_xOgC&pg=PA161&dq=conciliaci%C3%B3n+bancaria&hl=es&sa=X&ei=7VTFUN79Lo-HqwGk-4DQAQ&ved=0CCwQ6AEwAA#v=onepage&q&f=true

http://www.uazuay.edu.ec/analisis/El%20modelo%20relacional.pdf

http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf

http://es.scribd.com/doc/53804136/Libros-Auxiliares

http://translate.google.com.mx/translate?hl=es&langpair=en|es&u=http://smallbusiness.chron.com/benefits-bank-reconciliation-55857.html&ei=nj7vUKe-OeTQ2QWhx4D4CQ

http://www.uprb.edu/profesor/ntorres/base_de_datosventajasdesventajas.htm

http://www.gerencie.com/%C2%BFque-es-un-sobregiro-bancario.html

http://sofwarefreedom.wordpress.com/2010/10/21/licencias-bsd/

http://www.postgresql.org.es/sobre_postgresql