Trabajo Final de Graduación para optar por el título

Bachiller en Ingeniería en Computación

Informe Final

Migración Modulo Cuentas por Pagar

Yeudy Arias Alfaro

Carrera Ingeniería en Computación

Instituto Tecnológico de Costa Rica

Prof. Asesor: Oscar Víquez

Sede San Carlos

22 Noviembre de 2011

Resumen Ejecutivo

Business Advance es un sistema financiero contable, que fue desarrollado en el año 2004 por empleados de la compañía GBSYS, convirtiéndose así en uno de los principales productos de la empresa.

Durante los últimos años y tratando de satisfacer las necesidades de los clientes la empresa busca mejorar este sistema, migrando el producto a una plataforma más portable y que se pueda personalizar de acuerdo a los requerimientos de cada cliente.

Se puede mencionar que uno de los cambios más significativos que se le realizan al producto y que lo hace más atractivo a los ojos de los consumidores, es el hecho de ser adaptable a los distintos servidores de aplicaciones y plataformas de bases de datos.

Actualmente, el sistema se encuentra en una etapa final de migración correspondiente al flujo de compra, en la que se han desarrollado cuatro módulos: contabilidad, sistema, presupuesto, proveeduría, inventario, bancos y tesorería este último con dos sub modulo correspondiente a la emisión de pagos y las cuentas por pagar.

Para el desarrollo de este proyecto se cuenta con la participación del Director de Proyectos, el Ing. Luis Emilio Ramírez Espinoza, un Coordinador Técnico en Arquitectura, el Ing. Alberto Chávez y un Equipo de Desarrollo constituido por estudiantes de práctica del Instituto Tecnológico de Costa Rica y de la Universidad de Costa Rica.

Es de gran importancia mencionar la experiencia tan significativa que se obtuvo durante éste proceso, ya que el hecho de poner en práctica los conocimientos adquiridos beneficia en la formación profesional y personal de cada uno de los estudiantes que están relacionados estrechamente con la migración de este sistema.

Tabla de contenido

1. Contexto del proyecto

1.1 Descripción de la empresa

La empresa donde se desarrollará la práctica es Global Business System S.A (GBSYS). La razón de la empresa es ser una empresa especializada en Bases de Datos, y Desarrollo de Sistemas en ambientes ORACLE, Microsoft y Java. Adicionalmente se cuenta con una línea de distribución de productos de software especializados, incluyendo productos propios y otros ofrecidos por casas extranjeras reconocidas mundialmente.
También ofrecer servicios mediante la modalidad de Outsourcing ya sea en soporte técnico para administración de sistemas y bases de datos, como para desarrollo de aplicaciones.
La práctica supervisada se desarrollará en el departamento de gerencia de proyectos. La razón de ser del departamento es el de desarrollo, mantenimiento y consultoría de sistemas de información para el sector público y privado. Este departamento se centra en tecnologías Oracle, Java y Microsoft.
En cuanto a la estructura organizacional de la compañía se podría decir que es una organización de tipo funcional compuesta de la siguiente manera:

1.2. Visión de la empresa

La visión de la empresa es "ser reconocidos mundialmente como una corporación especializada en tecnología informática."

1.3. Misión de la empresa

La misión de la empresa es "ofrecer productos y servicios de la más alta calidad en tecnología informática que mejoren la eficacia, eficiencia y rentabilidad de nuestros clientes."

1.4. Objetivos de la empresa

Objetivo general

Ofrecer a los clientes de GBSYS, productos y servicios de alta calidad, que cumplan con sus necesidades y que se encuentren al alcance de su negocio.

Objetivos específicos

Entre los objetivos que se han establecido dentro de esta visión empresarial se destacan los siguientes:

- ✓ Desarrollar y dar mantenimiento a sistemas de cómputo de aplicación específica ajustados a los distintos niveles organizacionales y las condiciones particulares en que se desempeña cada usuario.

- ✓ Brindar entrenamiento en el manejo de equipo y paquetes de software en instituciones y empresas, por medio de seminarios, conferencias, charlas y cursos cortos.

- ✓ Brindar asesoría técnica en la selección y compra de equipo de cómputo.

- ✓ Desarrollar paquetes de software generalizados bajo estándares de calidad mundial.

- ✓ Distribuir software especializado de casas extranjeras de amplio reconocimiento en la industria.

1.5 Organigrama

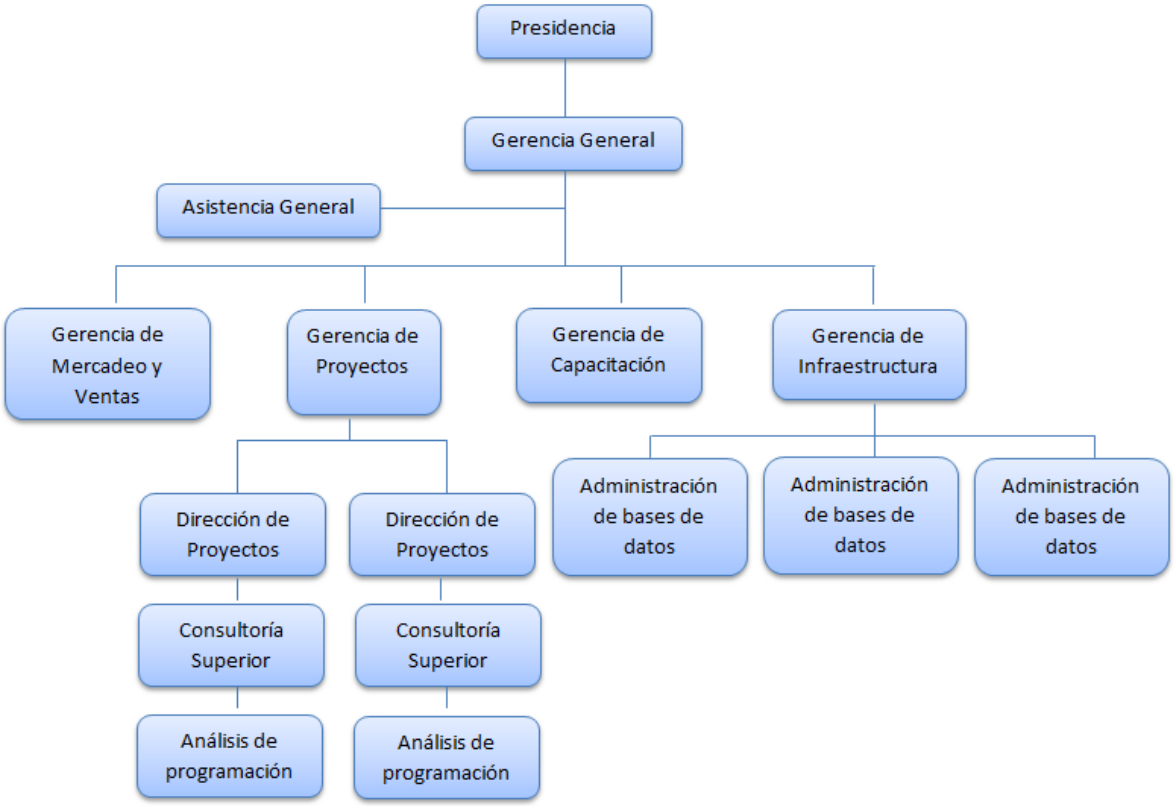

Imagen 1 - Organigrama de la empresa GBSYS

1.6 Estructura organizacional

En cuanto a la estructura organizacional de la compañía se podría decir que es una organización de tipo funcional compuesta de la siguiente manera:

- ✓ Presidente
- ✓ Gerente General

1.7 Departamentos

- ✓ Asistente Administrativa
- ✓ Gerente de Ventas
- ✓ Gerente de Proyectos
- ✓ Gerente de Infraestructura
- ✓ Gerente de Capacitación

1.8 Ubicación del departamento donde se realiza la práctica

1.8.1 Razón de ser del departamento

La práctica supervisada se desarrollará en el departamento de gerencia de proyectos. La razón de ser del departamento es el de desarrollo, mantenimiento y consultoría de sistemas de información para el sector público y privado. Este departamento se centra en tecnologías Oracle, Java y Microsoft.

1.8.2 Cantidad de recurso humano en el departamento

La cantidad de recursos humanos en el departamento de gerencia de proyectos es de 42 integrantes.

1.9 Antecedentes del Proyecto

1.9.1 Descripción general del proyecto

En la actualidad existe un producto (ERP) de la empresa denominado *Business Advance* (Sistema Administrativo Financiero) con 4 instalaciones en diferentes clientes. *Business Advance* es una solución financiero administrativo orientada a optimizar la gestión de las unidades de negocio de la organización.

Business Advance es un ERP que está compuesto e integrado por los siguientes módulos:

- ✓ Seguridad
- ✓ Administración
- ✓ Contabilidad
- ✓ Presupuesto
- ✓ Control Bancario
- ✓ Caja Chica
- ✓ Emisión de pagos
- ✓ Cuentas por Cobrar
- ✓ Transportes
- ✓ Cuentas por Pagar
- ✓ Activos Fijos
- ✓ Inventarios
- ✓ Proveeduría
- ✓ Custodia de Valores
- ✓ Inversiones
- ✓ Facturación

Los objetivos principales de *Business Advance* para la organización son:

- ✓ Manejo de la información detallada y consolidada con el propósito de poder medir y controlar cada unidad de negocio.
- ✓ Mantener y mejorar su competitividad ante el dinámico ambiente que genera la globalización y la apertura de mercados.
- ✓ Tomar decisiones en tiempo real en sus actividades de negocio con el propósito de consolidarse y fortalecerse.
- ✓ Planificar, ejecutar y controlar con eficiencia sus proyectos corporativos.

1.9.2 Antecedentes del proyecto

El actual producto *Business Advance,* fue desarrollado hace aproximadamente 6 años utilizando la herramienta Oracle Developer Suite 9i. Posteriormente fue migrado utilizando Oracle Developer Suite 10g, debido a las necesidades y cambios tecnológicos, así como también a factores económicos, y requisitos de los clientes. Actualmente los clientes han mostrado gran interés en tener la funcionalidad de este producto en plataforma WEB utilizando J2EE. Con esto se pretende lograr que la aplicación sea más portable a otros servidores de aplicaciones a parte del OAS.

2. Descripción del problema

2.1 Enunciado del Problema

El producto Business Advance se encuentra desarrollado bajo un servidor de aplicaciones ORACLE. Los productos ORACLE, a pesar de ser herramientas líderes a nivel mundial para la gestión de Bases de Datos, y de brindar gran robustez y fiabilidad en sus sistemas, requiere de un alto costo para su obtención y mantenimiento, siendo este uno de los principales inconvenientes del sistema actual. Los elevados precios para la adquisición de licencias, han llevado a muchas compañías a optar por aplicaciones que trabajen sobre plataformas de menor costo, y de esta forma optimizar más la funcionalidad de la aplicación.

2.2 Enunciado de la solución

Business Advance es una solución financiero administrativo orientada a optimizar la gestión de las unidades de negocio de la organización.
Se requiere que el producto sea migrado nuevamente a un servidor de aplicaciones más portable, utilizando tecnologías actuales, esto con el fin de independizar la lógica del negocio con el servidor de Base de Datos, lo que permite así, que el producto se pueda ofrecer en distintos precios dependiendo de sus características y reduciendo costos. Esto facilita además, que sean más las compañías, que puedan adquirir el producto

2.3 Productos esperados

Entre los productos esperados al finalizar la práctica se encuentran los siguientes:

- ✓ Documentación de la arquitectura a utilizar.
- ✓ Módulo de Cuentas por Pagar desarrollado en J2EE (fuentes y ejecutables).
- ✓ Pruebas de funcionalidad y rendimiento de los módulos migrados.
- ✓ Manual de usuario y técnico de ambos módulos.
- ✓ Capacitación en el uso de los módulos migrados.

2.4 Personal involucrado

Entre las personas involucradas en el proyecto se encuentran las siguientes:

- ✓ El Ing. Luis Emilio Ramírez Espinoza, Coordinador del Proyecto con conocimientos del producto que apoyará en las fases del proyecto, la planeación de las actividades y el control del cronograma.
- ✓ El Ing. Pablo Peraza Vargas coordinador técnico en arquitectura y desarrollo de aplicaciones WEB.
- ✓ Equipo de desarrollo conformado por tres estudiantes de práctica, Yeudy Arias, Víctor Guzmán, Claudia Leiva, y dos desarrolladores, Alberto Chávez y Cristina Rojas.

2.5 Necesidades y Expectativas

Necesidad	Problema que conlleva	Solución Actual	Solución que se propone
Independizar la lógica del negocio del sistema gestor de base de datos.	Migrar la totalidad de la lógica del negocio almacenada en procedimientos de base de datos y pasarlos al lenguaje al que se desea migrar.	Procedimientos almacenados encargados de gran parte de la lógica del negocio	Procedimientos elaborados en un lenguaje de alto nivel (java) encargados de la totalidad de la lógica del negocio
Migrar a tecnologías de desarrollo orientada a objetos como java, para agilizar el proceso de desarrollo del módulo actual y posterior.	Transcribir la lógica de los procedimientos en un lenguaje distinto		Utilizar un lenguaje de alto nivel, orientado a objetos como java, facilitando el desarrollo del mismo.
Reducir costos en adquisición de licencias, utilizando tecnologías de menor costo	Ajustar las características del producto actual, para que sea perfectamente adaptables a tecnologías de menor costo	El producto se encuentra elaborado bajo un servidor de aplicaciones ORACLE, utilizando además ORACLE Forms para la interfaz gráfica del cliente	Utilizar tecnologías como Java server faces, hibernate y java, que son tecnologías de código abierto, con soporte garantizando.

2.6 Requerimientos no funcionales

Escalabilidad

- ✓ El diseño debe contemplar el uso óptimo de recursos tales como conexiones a la base de datos, o Contemplar en el diseño la clara partición entre datos, recursos y aplicaciones para optimizar la escalabilidad del sistema.
- ✓ El sistema debe contemplar requerimientos de crecimiento para usuarios tanto internos como externos.

Disponibilidad

- ✓ El sistema debe contemplar requerimientos de confiabilidad y consistencia de los componentes de negocio ante recuperaciones. En caso de fallas de algún componente, no debe haber pérdida de información.
- ✓ La aplicación debe contemplar requerimientos de consistencia transaccional. Ante la falla del aplicativo, se debe contar con mecanismos que contemplen la interrupción de transacciones para que estas finalicen de manera correcta.

Mantenibilidad

- ✓ Se debe estructurar el código de una manera consistente y predecible.
- ✓ Para objetos que son frecuentemente manejados en la lógica del negocio, implementar las respectivas interfaces que aseguren su fácil implementación en el sistema.
- ✓ Asegurar que el diseño de las interfaces contemplen el que las propiedades públicas y los parámetros de los métodos sean de un tipo común (estandarizados).
- ✓ El sistema debe ser construido e implantado de tal manera que un cambio en los parámetros de negocio no obligue a la generación de una nueva versión del módulo.

Desempeño

- ✓ La aplicación debe ofrecer un buen desempeño del sistema ante una alta demanda acorde a los requerimientos funcionales y no funcionales de la solución.

2.7 Características generales.

2.7.1 Tecnologías a utilizar:

- ✓ Maven
- ✓ JAVA
- ✓ Framework de persistencia de datos Hibernate
- ✓ Framework de presentación JAVA SERVER FACES (JSF)
- ✓ Base de datos ORACLE 10g

3. Análisis de Riesgos

Riesgo	Categoría	Posible causa	Impacto	Probabilidad ocurrencia	Nuevos Cambios	Estrategia de mitigación	Estrategia de contingencia
Subestimación en las tareas que se establecen para el Spring semanal	Diseño	No se tienen requerimientos claros a la hora de establecer las tareas de la historia de usuario correspondiente	Atraso en el Spring semanal y por ende en el desarrollo del proyecto en general	Media	Actualmente se ha presentado este riesgo en ocasiones, ya que muchos de los requerimientos no son claros	Para la mitigación de este riesgo, se hace un análisis semanal del esfuerzo de tareas anteriores y se estima el esfuerzo de la nueva tarea	Si la tarea se ha subestimado se aplica más esfuerzo a dicha tarea para finalizar en el Spring acordado, esto quiere decir que la tarea que antes le correspondía a un solo desarrollador ahora pasaría a resolverse por dos desarrolladores
Perdida de la conexión del servidor de base de datos del sistema	Tecnológico	El servidor en el que se encuentra la base de datos del sistema se encuentra en otro departamento de la empresa que no es el departamento de desarrollo (soporte) y se utiliza para otra pruebas	Atraso en el Spring semanal y por ende en el desarrollo del proyecto en general	Media	Este se ha convertido en un riesgo de probabilidad media, ya que la maquina donde se encuentra la BD, también es trabajada por otro grupo de desarrollo y presenta caídas ocasionalmente	Para la mitigación de este riesgo, se deben de establecer tareas que no precisen de la base de datos del sistema, por lo tanto ante cualquier caída de la conexión se atacan este tipo de tareas.	En el caso de no contar con el servidor de base de datos, se atacan las tareas que no requieren de esta conexión, como la creación de interfaz de usuario, etc.
Tareas pendientes en el cierre del	Tecnológico	No se tienen requerimientos claros a	Atraso en el Spring semanal y por	Baja	Este riesgo no se ha presentado desde el	Para la mitigación de este riesgo, se hace	Se establece en el siguiente Spring las

Spring semanal		la hora de establecer las tareas de la historia de usuario correspondiente	ende en el desarrollo del proyecto en general		inicio del proceso de desarrollo	un análisis semanal del esfuerzo de tareas anteriores y se estima el esfuerzo de la nueva tarea	tareas que quedaron pendientes con una mayor prioridad ya que correspondían a tareas pasadas
Aparición de nuevas tareas no establecidas y que son requisito para la tarea que se está resolviendo	Tecnológico y diseño	No se tienen requerimientos claros a la hora de establecer las tareas de la	Atraso en el Spring semanal y por ende en el desarrollo del proyecto en	Baja	Este requisito cambia a probabilidad media, ya que ocasionalmente sucede debido a que los requisitos no son claro en su totalidad	Cada semana se hace un estudio de los requisito en general, para verificar la dependencia de nuevas tareas con otras posibles a futuro	Se establece en el siguiente Spring las tareas de este tipo con una mayor prioridad ya que muchas dependen de estas
Atraso en la reuniones de capacitación con el coordinador del proyecto	Personas		Atraso en el desarrollo del producto y en el establecimiento de requisitos	Media	No se han tenido atrasos en este tipo de reuniones, por lo tanto este riesgo cambia a una probabilidad baja		Se atacan los requisitos e historias de usuario que se tienen claras al momento y se dejan pendientes las que necesiten capacitación por parte del coordinador

4. Objetivos y alcances del Sistema

4.1Objetivo General

- ✓ Migrar el Módulo de Cuentas por Pagar del producto Business Advance de la empresa GBSYS a una plataforma WEB utilizando J2EE.

4.2 Objetivos específicos

- ✓ Capacitar al equipo desarrollador en las herramientas que se utilizarán en el desarrollo del sistema por medio de tutoriales, libros y otros medios para el uso correcto de las mismas.

- ✓ Capacitar al equipo desarrollador en la funcionalidad actual del Módulo de Cuentas por Pagar de Business Advance con el propósito de que conozcan el producto.

- ✓ Migrar el módulo de Cuentas por Pagar de Business Advance que están actualmente desarrollado con la herramienta Oracle Developer Suite 10g a plataforma WEB utilizando J2EE.

4.2 Alcances del proyecto

4.2.1 Cuentas por pagar

- ✓ El sistema permitirá el registro de nuevos formularios de cuentas por pagar para el próximo año en proceso, además de permitir aprobar y aplicar los mismos.
- ✓ El sistema permitirá el ingreso de documentos de cuentas por pagar, que modificarán las cuentas dentro del año en proceso.
- ✓ El sistema podrá generar movimientos bancarios a partir de documentos y formularios aplicados, y que se verán reflejados en las diferentes cuentas.
- ✓ El sistema permitirá realizar modificaciones de cuentas, modificando las cuentas por pagar.

5. Solución Implementada

5.1 Arquitectura conceptual de la solución

El nuevo producto Business Advance estará construido bajo una plataforma de J2EE [7]., la cual es una plataforma de programación, el cual es parte de la Plataforma Java [2], para desarrollar y ejecutar software de aplicaciones en Lenguaje de programación Java con arquitectura de N capas distribuidas y que se apoya ampliamente en componentes de software modulares, ejecutándose sobre un servidor de aplicaciones.

El hecho de estar desarrollando bajo esta plataforma genera una gran ventaja para el producto, ya que Java EE incluye varias especificaciones de API, tales como: JDBC[11], RMI[12], e-mail, JMS[15], Servicios Web[13], XML[14] que pueden tomarse como punto departida para el desarrollo del proyecto además de permitir una escalabilidad del producto.

El proyecto se encuentra dividido en 3 capas:

- ✓ Capa de acceso a datos
- ✓ Lógica del negocio
- ✓ Capa de Presentación

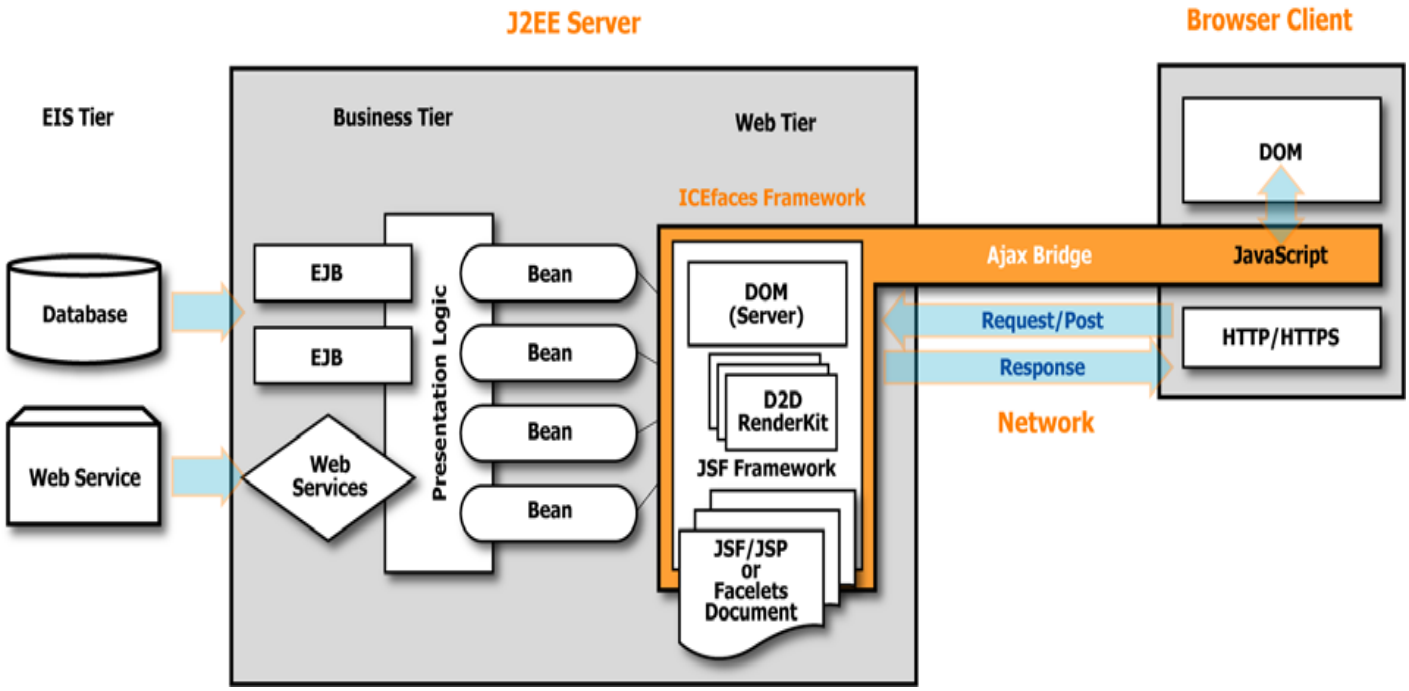

Figura 2 Diagrama General del Proyecto

5.1.1 Capa de Acceso a Datos

La capa de acceso a datos para Business Advance, a diferencia de otros productos de la empresa GBSYS, no utiliza “JDBC” para el acceso a base de datos. Se optó por implementar Hibernate [3], una nueva opción que la compañía está utilizando en algunos proyectos y que ha resultado exitoso para nuevos productos que se están desarrollando actualmente.

Hibernate es una herramienta de Mapeo objeto-relacional(ORM) para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones.

En otras palabras, el modelo de base de datos es representado por el modelo de objetos; únicamente el desarrollador se limita a indicar cuál será la relación de los objetos entity java e Hibernate se encargará de interpretarlos e integrarlos a la base de datos, asignando las relaciones entre tablas de una manera automática.

La clase ControlGenerico se encarga de llamar a un archivo de configuración “persistance” el cual le indica el datasource de la base de datos a la que se desea conectar. Es ahí cuando las clases Entity definidas por el desarrollador son interpretadas por Hibernate y éste las configurará como una tabla dentro de la base de datos indicada.

Ejemplo de una Clase Entity:

```
/**
 *
 * @author Yeudy Arias
 */

@NamedQueries({
@NamedQuery(name = "obtenerNegocios", query = "select f from MontosNegocio f where f.cod_cia = ?")})
@Entity
public class MontosNegocio implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String cod_negocio;
    private String des_negocio;
    private String nom_negocio;
    private String cod_cia;
    @OneToMany(mappedBy = "cod_negocio")
    private List<NegociosActivos> negociosActivoss;

    public MontosNegocio() {
    }

    public MontosNegocio(String cod_negocio, String des_negocio, String nom_negocio, String cod_cia) {
        this.cod_negocio = cod_negocio;
        this.des_negocio = des_negocio;
        this.nom_negocio = nom_negocio;
        this.cod_cia = cod_cia;
    }
```

C1

C2

Figura 3. Ejemplo de Entity

C1: El usuario una vez comprendido la lógica del sistema, establece las relaciones que se va a mantener entre las entidades, ya sea: 1:N, N:1, N:N, 1:1

C2: Corresponde al objeto al cual se va a realizar la relación, entre la entidad actual, con cualquier otra entidad u objeto del sistema.

5.1.2 Lógica del negocio

La lógica del negocio se encuentra desarrollada dentro de los paquetes de vistas del proyecto. Los llamados "Beans" [4, JSF] quienes son los que contendrán la lógica y algoritmos del proyecto. Estos "Beans" (archivos .java) son sumamente sencillos de implementar y conectar con la capa de presentación. El objetivo de la utilización de Java Server Faces, es permitir esa facilidad de enviar y recibir mensajes captados por la interfaz de usuario, y agilizar el proceso de desarrollo para las empresas.

Esta parte del proyecto, es la que le da inteligencia al sistema ya que dentro de cada Beans se encuentran almacenados métodos y variables que posteriormente van a ser usados y llamados desde el mismo Beans o ya sea desde los archivos .xhtml. Cada parámetro que es declarado se le tiene que asignar su get y set correspondientes, ya que esto es una ventaja para el programador a la hora de asignar y recibir datos.

Otra característica importante que presentan los Beans, es que dentro de estos se declaran métodos con gran funcionalidad para la aplicación, el principal y el cual es usado cada vez que una pantalla es cargada es el método de inicializar, que permite inicializar cada uno de los parámetros que fueron declarados al inicio del Beans, también es importante mencionar que existen métodos que permiten llamar los query que fueron declarados en diferentes entidades, enviándoles los parámetros que estos necesitan para realizar la consulta de datos.

Dentro del proyecto existen diferentes paquetes correspondientes a cada módulo y es en estos paquetes donde se van a guardar los Beans que son declarados para representar la lógico del sistema.

```java
import com.gbsys.baee.controlador.ControlGenerico;
import com.gbsys.baee.proveeduria.entidades.Bancos;
import com.gbsys.baee.utils.Mensaje;
import com.gbsys.baee.utils.Util;
import com.gbsys.baee.vista.genericos.VistaSesion;
import java.util.List;
/**
 *
 * @author Yeudy Arias
 */
public class VistaBancos {

    private Bancos catalogo_bancos;
    private List<Bancos> bancos;
    private VistaSesion sesion;
    String nom_corto_banco;
    private String fecha_registro;
    private String fecha_modificado;
    List<Bancos> bancosx;

    public VistaBancos() {
        inicializar();
    }

    public void inicializar() {

        this.sesion = (VistaSesion) Util.getBeanDeContexto("VistaSesion");
        cargarBancos(this.sesion.getCod_cia());
    }
```

C1

Figura 4. Ejemplo Beans

C1: Método que inicializa el Beans, donde se encuentran almacenados los parámetros que fueron declarados y que necesitan que sean inicializados al inicio del Beans.

5.1.3 Capa de presentación

Por último tenemos la capa de presentación que estará constituida por archivos XHTML y el framework Java Server Faces para la implementación de la Interfaz de Usuario, junto con componentes como “CSS” [9], Javascript[8] y JQuery [10] para efectos y eventos del diseño de interfaz.

5.2 Los modelos de subsistemas

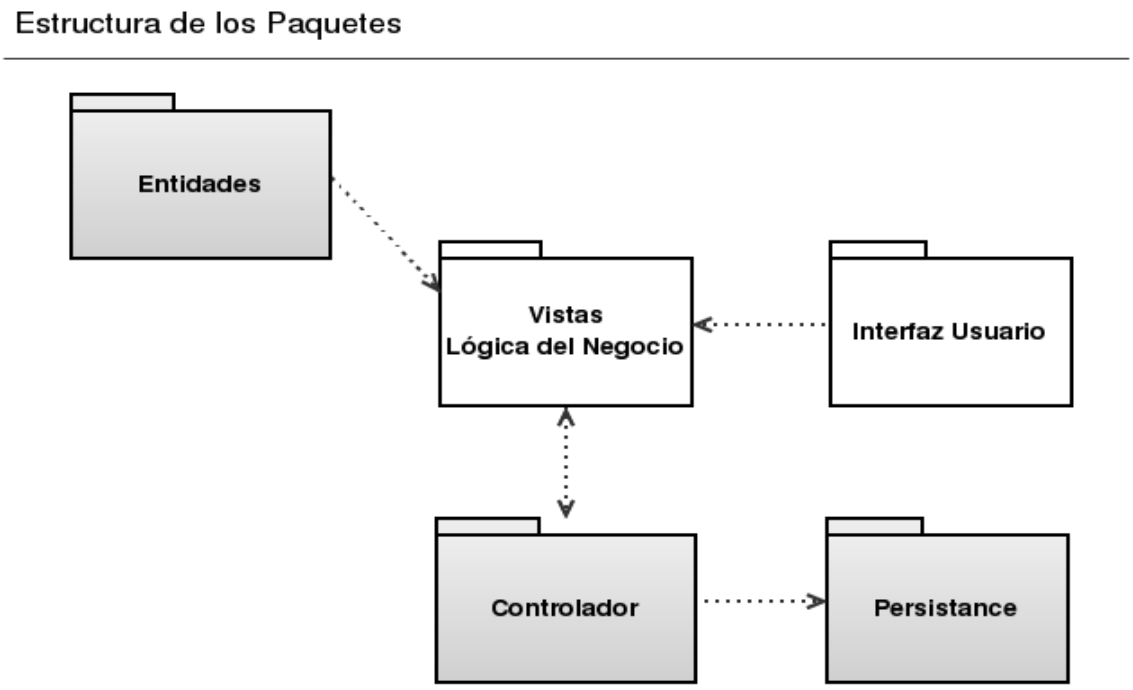

Figura 5. Estructura de los Paquetes

Entidades
Son el modelo de Objetos que una vez que son creados, serán representados en el servidor de base de datos con sus respectivas relaciones entre ellas.

Vistas
Clases en .Java encargada de procesar la información proveniente de la capa de presentación y de enviar una respuesta. Las vistas hacen llamadas a las entidades para operar con ellas y la capa de presentación hace llamadas a estas vistas.

Controlador
El controlador es el encargado de llamar al Persistance para todos los procesos que tengan que ver con llamadas a base de datos.

Persistance
Archivo XML encargado de amarrar la aplicación con la base de datos a través de Hibernate.

Interfaz de Usuario
Plantillas XHTML (Interfaz de Usuario) de la capa de presentación.

5.3 Diagrama de Clases

5.3.1 Diagrama de clases para Cuentas por Pagar

A continuación se muestra el diagrama de clase de cuentas por pagar, mostrando una serie de entidades con sus respectivas relaciones, necesarias para el flujo de compra del sistema, que permite definir la solución de diseño que se va a tener.
Cada una de estas entidades se encuentra relacionada por medio de un código de compañía.

CatalogoCuenta: Representa el catálogo general de cuentas contables para una empresa. Esta entidad es usada en el módulo de cuentas por pagar como también en otros módulos del flujo de compras.

DetalleFactura: Se encuentra estrechamente relacionada con la entidad "Factura" ya que presentan una relación bidireccional. "DetalleFactura" tiene la funcionalidad de almacenar cada una de las líneas por las cuales está creada una factura, describiendo detalladamente las características del artículo que se está registrando, como es la cantidad de items, descripción del ítem, montos, impuestos, etc.

Factura: Se incluyen todos los datos de la empresa, como también los datos necesarios para registrar una factura, como el monto total de esta, además contiene parámetros que permite llevar control de fechas como también de las personas que realizan movimientos de facturas. La entidad "Factura" presenta relación bidireccional con las entidades "Detallefactura" y "RebajoFactura".

PresupuestoFactura: Esta entidad presenta relación unidireccional con las entidades: "Proveedor", "OrdenCompra", "CentroCosto", "CatalogoCuenta" y relación bidireccional con "Factura". Tiene la funcionalidad de almacenar el monto de pago de la factura.

ParametrosProveeduria: Los parámetros de configuración de proveeduría y los parámetros de cuentas por pagar son representados de la misma forma en el sistema. Corresponde a parámetros de configuración para un control más detallado.

TiposDocumentos: No presenta relación con ninguna entidad, tiene la funcionalidad de describir los tipos de documentos que existen en el flujo de compras, almacenando características ligadas a los documentos de las cuentas por pagar.

TiposDeducciones: No presenta relación con ninguna entidad, contiene información de los tipos de deducciones que existen en el sistema, está estrechamente relacionado con la entidad “RebajoFacturas” pero no presenta ningún tipo de relación con esta.

RebajoFacturas: Presenta una relación bidireccional con la entidad “Factura” y una relación unidireccional con la entidad “CuentaContable”. Esta entidad fue creada con el fin de almacenar el descuento que se le va aplicar a una factura. Contiene parámetros que permiten tener un control de la deducción como es el código, descripción, el monto, etc.

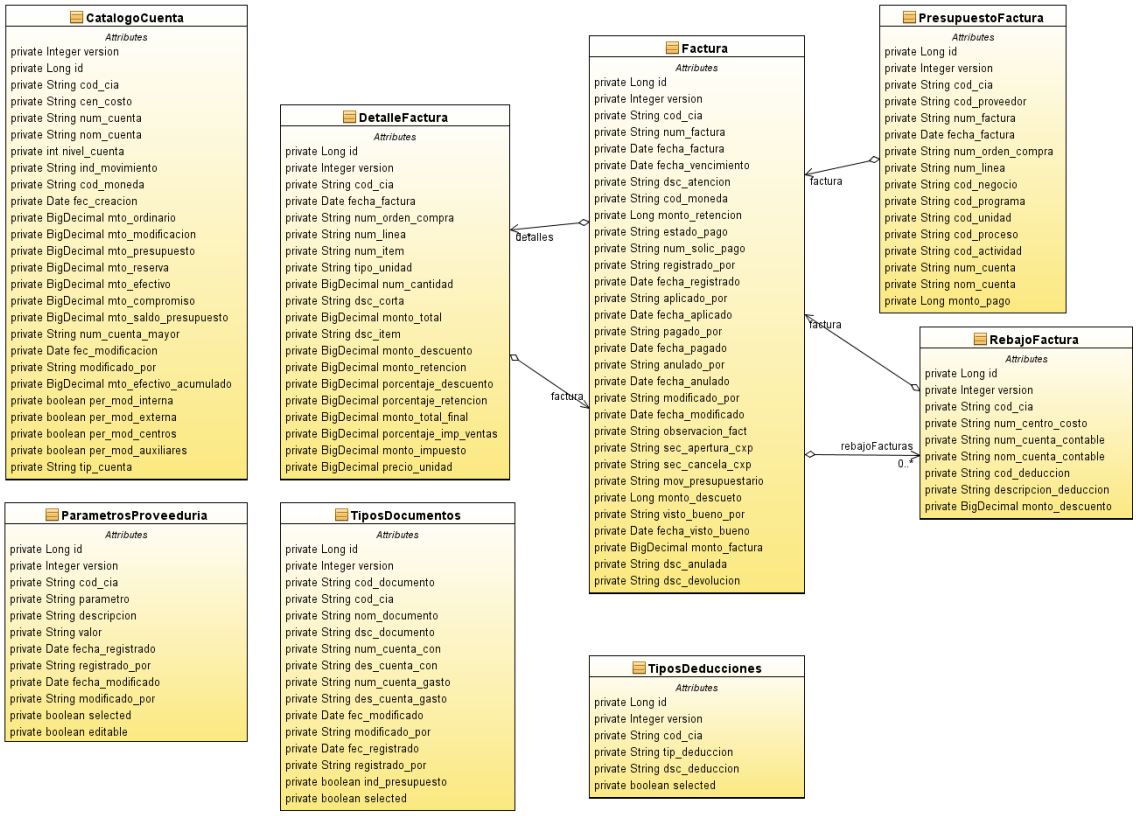

Figura 6. Diagrama de Clases. Cuentas por pagar

5.4 Interfaces de usuario

Java Server Faces (JSF) es una tecnología y framework para aplicaciones Java basadas en WEB que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa Java Server Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas [4]

Todas las plantillas de interfaz estarán desarrolladas bajo este Framework, ya que aporta gran cantidad de componentes y facilidades para los desarrolladores, además de ser sencillo y estructurado.

El diseño de la interfaz está básicamente constituido por una plantilla base, que a su vez contiene los encabezados que debe llevar todas las páginas, el menú respectivo de opciones que presta el producto y una área de despliegue.

Cada página hereda la plantilla base, con su encabezado y menú.

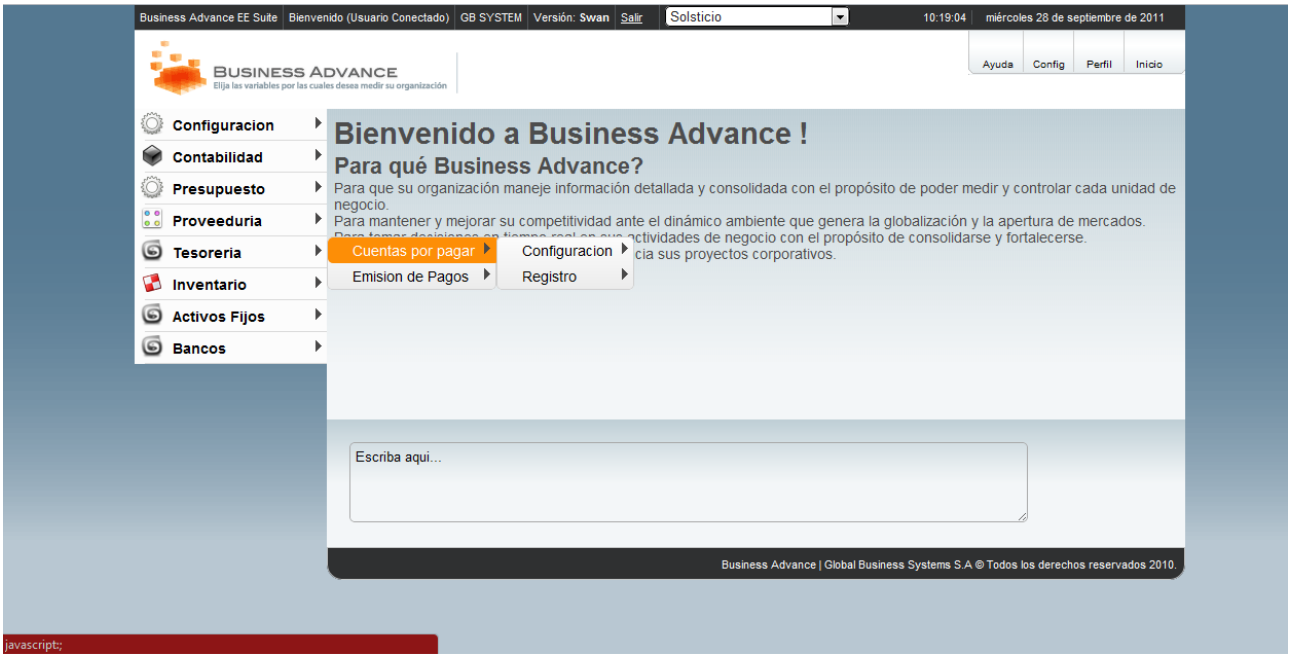

Figura 7. Plantilla Base

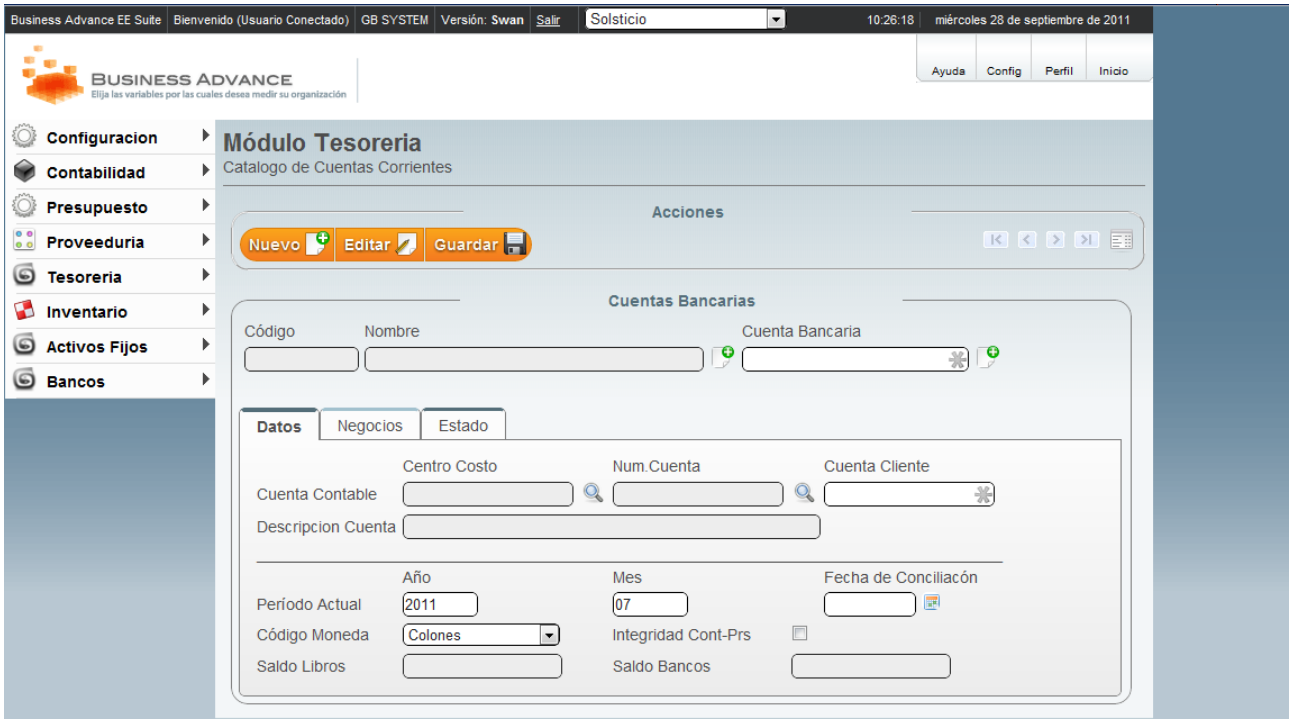

Figura 8. Página Estándar

Todas las páginas son archivos .XHTML que únicamente se mostrarán en el área de despliegue de la plantilla, con un formato especifico. El usuario no sentirá una recarga de la página ya que solamente el área indicada será la que cambie.

Nota: El diseño global de la interfaz actual, solo es una versión previa, ya que el diseño final (estilos, CSS) son aportados por un diseñador gráfico, aparte cuando todos los módulos del producto se encuentren implementados. El diseño actual es únicamente una base para el desarrollo del producto.

5.5 Componentes y servicios

Maven

Maven una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002. Es similar en funcionalidad a Apache Ant (y en menor medida a PEAR de PHPy CPAN de Perl) pero tiene un modelo de configuración de construcción más simple, basado en un formato XML [1]. Maven utiliza un Project Object Model (POM) para describir el proyecto de software a construir, sus dependencias de otros módulos, componentes externos y el orden de construcción de los elementos. Viene con objetivos predefinidos para realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado [1].

Hibernate

Es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones.

Hibernate es software libre, distribuido bajo los términos de la licencia GNU LGPL.

Java Server Faces

Es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa Java Server Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías como XUL. [4]

Apache Tomcat

Tomcat es un servidor web con soporte de servlets y JSPs. Tomcat no es un servidor de aplicaciones, como JBoss o JOnAS. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache [5].

Base de Datos ORACLE

Oracle es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation [6].

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando:

- ✓ Soporte de transacciones
- ✓ Estabilidad
- ✓ Escalabilidad
- ✓ Soporte multiplataforma.

CSS

Las hojas de estilo en cascada (en inglés *Cascading Style Sheets*), CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores [9].
La idea que se encuentra detrás del desarrollo de CSS es separar la *estructura* de un documento de su *presentación*

Las ventajas de utilizar CSS (u otro lenguaje de estilo) son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los navegadores permiten a los usuarios especificar su propia hoja de estilo local, que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o, incluso, a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil o ser "leída" por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño (siempre y cuando no se utilice estilo en línea).

JQuery

jQuery es una biblioteca o framework de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de enero de 2006 en el BarCamp NYC [10].

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

5.6 Diseño de base de datos

Hibernate se encarga de convertir el modelo de objetos en un modelo de base de datos. Por lo tanto, será el encargado de asignar las llaves necesarias entre las tablas de una manera automática según las relaciones que existen entre los objetos Entidad.

El objetivo de la utilización de este servicio es independizar en su totalidad la lógica del negocio con la base de datos. Al contener el diseño en el modelo de objetos no es imprescindible un tipo específico de servidor de base de datos, ya que Hibernate se adapta fácilmente a la gran mayoría de gestores de bases de datos comerciales actualmente: MySql, Postgres, SQL, Oracle, otros.

La siguiente imagen representa cada una de las tablas con sus respectivas relaciones del módulo de cuentas por pagar, que a su vez son utilizadas en otros módulos del flujo de compras.

Para obtener datos de cada una de las tablas, se crean los querys en las entidades según la necesidad del programador, estos son ejecutados en los beans y los datos son almacenados en cualquier tipo de parámetro, según el valor a retornar por el query. Cada una de las tablas contiene un ID que es generado autogenerado por el sistema y es el que permite realizar la identificación de objetos.

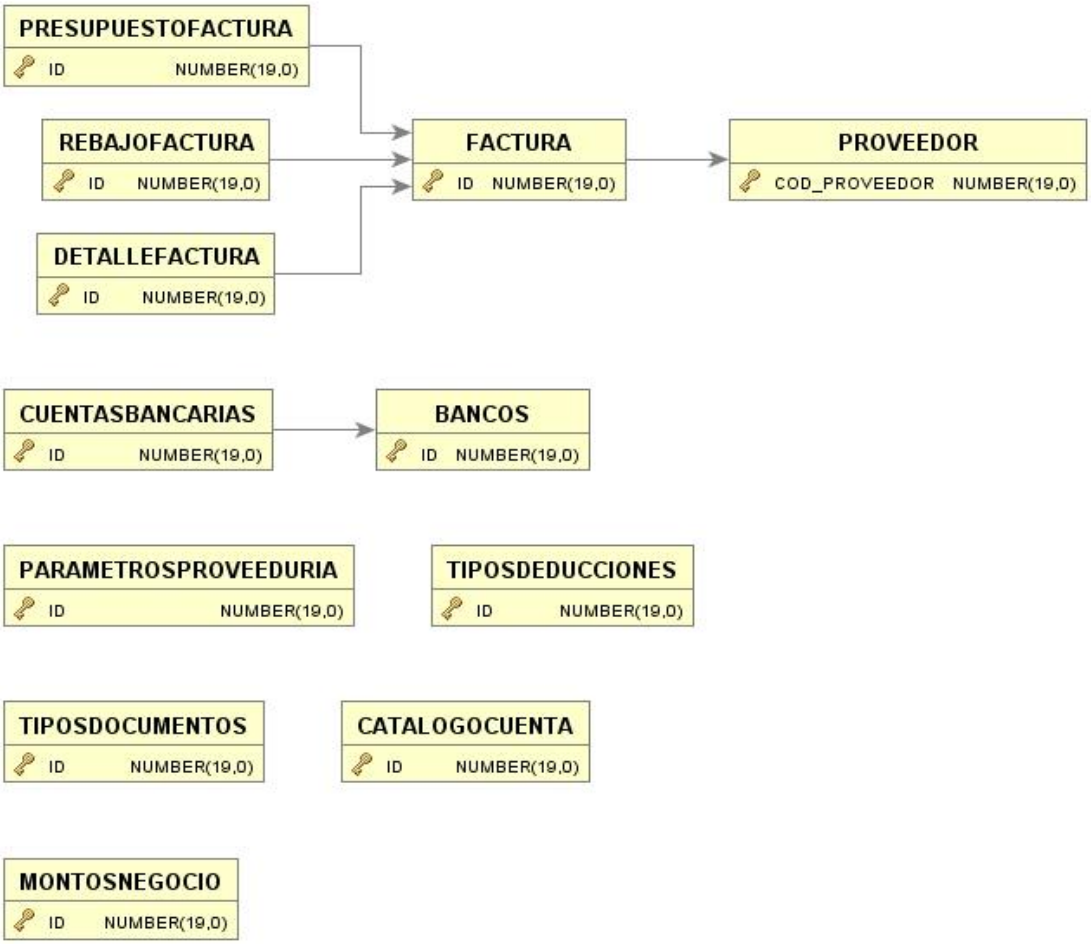

Figura 9. Diseño de la base de datos

6. Conclusiones y comentarios

Módulo Cuentas por Pagar

El proyecto correspondiente a la práctica de especialidad, específicamente del módulo de cuentas por pagar, se logró implementar con toda la funcionalidad requerida para la administración de los formularios correspondientes a una parte del flujo de compras. Este flujo compras, es una mezcla entre varios módulos, el cual la empresa requiere que se encuentre en un perfecto funcionamiento en un lapso corto de tiempo, esto nos obliga a desarrollar una migración casi perfecta, por lo que la conexión entre estos módulos hizo que existiera una comunicación adecuada entre los desarrolladores.

El módulo de cuentas por pagar es capaz de registrar una factura, darle el visto bueno y crear una cuenta por pagar, así mismo permite pagar una factura, generando una solicitud de pago, procesos a los que se les tiene que prestar mucha atención, ya que tiene relación con bancos, cuentas bancarias, montos, etc. Por lo que se tiene que tener mucho cuidado y entender a fondo todo lo que correspondiente a cuentas por pagar.

Módulo Emisión de Pagos

El módulo de emisión de pagos corresponde a la parte final del flujo de compras, el cual genera solicitudes de pago. Al inicio de la práctica este módulo junto con el módulo de bancos no se encontraban en los planes de desarrollo, pero debido a la relación que existe con el módulo de cuentas por pagar y los movimientos que se generan, se tuvo que realizar una implementación en paralelo de estos tres módulos.

Lo correspondiente a emisión de pagos se logró implementar en su totalidad, únicamente se encuentra pendiente las pruebas del flujo de compras para dar por concluido el flujo de compras de Avance.

Los objetivos logrados dentro de los módulos de emisión de pagos y bancos se encuentran:

- ✓ La capacidad de aplicar y entregar cheque
- ✓ La aplicación de una transferencia.
- ✓ Generación del archivo de transferencia.
- ✓ Consulta de movimientos en libros.
- ✓ El modulo cuenta con la característica de mostrar en detalle la emisión de pago, ya sea por cheque o transferencia.
- ✓ La capacidad de registrar, modificar y eliminar bancos.
- ✓ La capacidad de registrar nuevas cuentas bancarias.

Como se detalló anteriormente, todos los objetivos planteados para el módulo de cuentas por pagar durante el inicio de la práctica de especialidad, se concluyeron de manera satisfactoria.

Por último, uno de los objetivos más importantes que se logró durante el tiempo de trabajo en la práctica de especialidad, fue la experiencia y el conocimiento obtenido tanto en las tecnologías que se utilizaron, como también en el aprendizaje sobre el manejo del cuentas por pagar y solicitudes de pago de una compañía.

Cabe recalcar que el uso de las tecnologías ayudo a la simplificación del trabajo, un ejemplo es la utilización de Hibernate ya que por medio de esta herramienta el programador se desentiende de un de gran parte del uso de la base de datos, debido a que es una herramienta capas de manipular de una manera eficiente el uso de tablas, relaciones y todo lo relacionado con bases de datos, también se debe mencionar el uso de IceFaces, ya es una herramienta grafica que ayuda a disminuir la cantidad de código que se digita, debido a que solamente una porción de la pantalla es la que se va a estar alterando por medio de componentes AJAX que se encuentran ligados a IceFaces. En cuanto a la metodología de trabajo se utilizó SCRUM que permitió tener un control adecuado de las tareas con el fin de medir el tiempo y estimar las tareas que fueron asignadas de una manera correcta.

La metodología utilizada durante el proceso de desarrollo llamada SCRUM, permitió conocer y aprender sobre el trabajo de equipo en una compañía y la importancia que tiene para con la misma. Esta metodología permitió además, obtener las bases sobre la administración de proyectos y la autogestión en un proceso de desarrollo. Así mismo, reflejó la importancia de una buena organización en un equipo de trabajo.

7. Referencias

[1] Maven. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/Maven

[2] ¿Qué es la tecnología Java y por qué lo necesito? Consultado el 27 de Setiembre de 2011. Disponible en: http://www.java.com/es/download/faq/whatis_java.xml

[3] Hibernate. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/Hibernate

[4] JSF. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/JavaServer_Faces

[5] Tomcat. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/Tomcat

[6] Base de Datos Oracle. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en http://es.wikipedia.org/wiki/Oracle

[7] Java EE. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/Java_EE

[8] JavaScript. Wikipedia, consultado el 27 de Setiembre. Disponible en: http://en.wikipedia.org/wiki/JavaScript

[9] Hojas de estilo en cascada. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada

[10] JQuery. Wikipedia, consultado el 27 de Setiembre de 2011. Disponible en: http://es.wikipedia.org/wiki/JQuery

[11] JDBC. Wikipedia, consultado el 09 de Octubre de 2011. Disponible en: http://es.wikipedia.org/wiki/Java_Database_Connectivity

[12] RMI. Wikipedia, consultado el 09 de Octubre de 2011. Disponible en: http://es.wikipedia.org/wiki/Java_Remote_Method_Invocation

[13] Servicios Web. Wikipedia, consultado el 09 de Octubre de 2011. Disponible en: http://es.wikipedia.org/wiki/Servicio_web

[14] XML. Monografías, consultado el 09 de Octubre de 2011. Disponible en: http://www.monografias.com/trabajos7/xml/xml.shtml