Instituto Tecnológico de Costa Rica
Sede Regional San Carlos

Escuela de Ingeniería en Computación

"Desarrollo de Funcionalidades para el Sistema de Intermediación de Valores para Fondos de Inversión y Puestos de Bolsa"

Segundo Informe de Proyecto de Graduación para optar por el grado de Bachiller en Ingeniería en Computación

Rafael Ángel Sánchez Saborío.

San José, Abril 2008.

Resumen Ejecutivo.

El objetivo principal que tiene el presente documento es mostrar el Informe Final del curso de Práctica de Especialidad desarrollado para el Instituto Tecnológico de Costa Rica en la empresa Lidersoft, la cual consistió en generar un nuevo modelo del sistema de acceso a datos utilizando tecnologías de vanguardia para así brindar un manejo más eficiente del sistema completo, de manera que se mejoren sus procesos y brindar mejores servicios. En los siguientes apartados del documento se darán a conocer algunos de los detalles del origen y la razón de ser del proyecto, así como del proceso de desarrollo tomados en cuenta para las tareas de práctica, habrán una serie de conclusiones, comentarios y recomendaciones que surgieron durante el proceso de práctica de especialidad.

Este documento está dividido en las siguientes secciones:

- Descripción del Problema: Abarca temas actualizados de los puntos desarrollados en el primer informe de práctica que se presentó. Entre los puntos que abarca se encuentran el contexto en el que fue desarrollado el proyecto, la descripción general de sus principales características, los objetivos y alcances que se desean lograr con él, así como los riesgos que se pueden llegar a presentar durante el desarrollo del mismo.

- Solución Implementada: En esta sección se presenta una recopilación actualizada de la información sobre el diseño del proyecto presentada en el segundo informe de práctica. En él se hace referencia a puntos como el diseño arquitectónico del sistema, clara descripción de los componentes, además de una descripción básica del diseño y estructura que poseen las clases y sus principales métodos y procedimientos.

- Conclusiones y Comentarios: En esta última sección se hace referencia de una forma general al trabajo realizado durante toda la práctica de especialidad desde el punto de vista de la experiencia laboral que se logró.

Lidersoft
Soluciones y Servicios.

Tabla de Contenido

Tabla de Imágenes

Capítulo 1

DESCRIPCIÓN DEL PROBLEMA.

1.1 Contexto del Proyecto.

1.1.1 Antecedentes del Proyecto

El proyecto inició en el año 2005 como una iniciativa de la compañía en conjunto con varias instituciones financieras para ofrecer una solución informática a los Puestos de Bolsa y las Sociedades de Fondos de Inversión de estos grupos financieros. Desde entonces, el proyecto ha ido creciendo según la necesidad de los clientes y se ha convertido en uno de los proyectos más importantes que posee la empresa, por lo cual se busca cambiar el producto principal de la empresa hacia tecnologías más eficientes y novedosas.

1.2 Descripción del Problema

1.2.1 Enunciado del problema:

El problema básicamente consiste en que se desea cambiar la arquitectura del producto principal de la empresa, llamado BROKER, hacia tecnologías de punta. Este proyecto tiene como objetivo principal conformar y consolidar el producto BROKER para así lograr brindar a los clientes un mejor producto que cumpla con sus expectativas.

1.2.2 Enunciado de la solución:

Como solución, se propone migrar la aplicación existente hacia una nueva tecnología llamada EJB 3.0.

De manera que el propósito principal de este proyecto es generar una investigación exhaustiva sobre como migrar el BROKER a esta arquitectura, para lo cual se van a presentar aplicaciones que ejemplifiquen y muestren las utilidades de EJB 3.0, así como la migración completa de uno de los módulos actuales del sistema, con el propósito de analizar la factibilidad del cambio.

1.2.3 Descripción de los participantes:

Los participantes en la resolución del proyecto son:

- El Estudiante: Rafael Ángel Sánchez Saborío.
- Arquitecto de Lidersoft: José David Gonzáles.
- Administrador de Proyectos: Christopher Barboza Herrera.
- Profesor Asesor: Ing. Abel Méndez.

1.3 Análisis de los Riesgos

Puesto que el presente proyecto es solamente una fase de investigación para generar una propuesta de cambio al sistema principal de la empresa, no existen riesgos específicos que afecten directamente el producto desde las perspectiva del cliente. Esto porque la propuesta se basará en tecnologías de punta, para las cuales se necesitará generar la investigación correspondiente, además de pruebas tangibles de lo investigado.

Dadas las circunstancias, los riesgos detectados son los siguientes:

- El esfuerzo es mayor que el estimado
- Necesidad de más tiempo para adaptarse a las nuevas herramientas.

Nombre del riesgo	El esfuerzo es mayor que el estimado
Categoría del riesgo	Alta
Posible causa del riesgo	Mal diseño del plan de trabajo
El impacto que tiene el riesgo para el proyecto	2 semanas
La probabilidad de ocurrencia	60%
La exposición que se tendrá ante el riesgo	1.5%
La estrategia de evasión	Revisar semanalmente el cronograma estimado para aplicar cambios convenientes y reestructurar el tiempo disponible entre las tareas.
La estrategia de mitigación	Desarrollo y revisión de calidad
Estrategia de contingencia	Continuar con el proyecto y evitar devolverse a fases anteriores

Nombre del riesgo	El personal de desarrollo necesita tiempo para adaptarse a las nuevas herramientas
Categoría del riesgo	Alta
Posible causa del riesgo	Desconocimiento de ciertos estándares usados por la empresa
El impacto que tiene el riesgo para el proyecto	3 semanas
La probabilidad de ocurrencia	50%
La exposición que se tendrá ante el riesgo	1.%
La estrategia de evasión	Capacitarse antes de iniciar la fase de implementación
La estrategia de mitigación	Buscar ayuda en manuales técnicos y en personas capacitadas en estos estándares
Estrategia de contingencia	Investigar más a fondo sobre los estándares de desarrollos utilizados por la empresa

1.4 Objetivos y Alcances del Sistema

1.4.1 Objetivo General:

- Realizar una nueva propuesta tecnológica de desarrollo para el Sistema de Intermediación de Valores para Fondos de Inversión y Puestos de Bolsa, que involucre tecnologías innovadoras para lograr un proceso de desarrollo más simple y eficiente en todo el sistema.

1.4.2 Objetivos Específicos:

- Iniciar un proceso de investigación sobre las capacidades de las nuevas tecnologías, JPA y EJB 3.0.
- Generar la propuesta del uso de JPA y EJB 3.0, utilizando el Módulo de Seguridad para realizar las pruebas de integración de estas tecnologías.
- Proponer una visión de la factibilidad de acuerdo a la propuesta generada para la posible migración hacia estas nuevas tecnologías en todos los módulos y el cambio en la forma de desarrollar a lo interno en Lidersoft.

1.4.3 Alcances:

- Generación de una nueva capa de acceso a datos utilizando la tecnología EJB 3.0 para ser usada en cada uno de los módulos del Sistema Broker.
- Acoplar el Módulo de Seguridad actual a la nueva capa de acceso a datos, con sus operaciones básicas de inserción, borrado, modificado y consulta de datos.
- Cambiar la especificación de las clases del Módulo de Seguridad de EJB 2.x a EJB 3.0.
- Utilizar como proveedor de persistencia la tecnología Hibernate.
- Usar al Servidor de Aplicaciones WebSphere 6.1 de IBM.

Capítulo 2

PRODUCTOS GENERADOS EN LA FASE DE INVESTIGACIÓN

2.1 Descripción de las Tecnologías.

A continuación se brinda una descripción de las tecnologías que se utilizan en este proyecto, tanto las actuales como las que formarán parte de la base del producto final creado. Estas tecnologías son:

EJB 2.x:

JPA.

Hibernate.

2.1.1 EJB 2.x.

Los Enterprise JavaBeans (EJB) son una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE de Sun Microsystems (ahora JEE 5.0). Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son, precisamente, los EJBs.

Los EJBs proporcionan un modelo de componentes distribuido estándar del lado del servidor. El objetivo de los EJBs es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables.

En su especificación EJB 2.x, la cual es la base actual de la arquitectura del sistema, se muestra la información de los siguientes puntos:

- Terminología.
- Tipos de Enterprise Bean.
- Forma en que se instancian los beans.
- Funcionamiento de los EJB.
- Interface Home.
- Interface Remota.

Terminología:

- ENTERPRICE BEAN:
  Objeto distribuido que provee servicios remotos y está albergado en un EJB Container. Tiene dos interfaces: home y local/remote, y una implementación.

- EJB API:
  Modelo de programación constituido por un conjunto de clases y convenciones.

- EJB CONTAINER:
  Alberga y maneja los enterprice beans, incluye acceso remoto, seguridad, persistencia, transacciones, concurrencia, acceso y pooling de recursos. El enterprice bean depende del container para todo y utiliza tres métodos para comunicarse con él:

  a. Callback Methods
     Permiten al bean hacer algo antes y después de un evento.

  b. EJB Context
     Permite al bean acceder información sobre el ambiente, como identidad del cliente, status de una transacción o accesos remotos a sí mismo.

  c. JNDI
     Es la extensión estándar de la plataforma de Java para acceder sistemas de nombres. Cada bean tiene acceso a un sistema de nombres especial (ENC) que es manejado por el container y accedido por el JNDI.

Tipos de Enterprise Beans:

Existen tres tipos de EJBs:

- EJBs de Entidad (Entity EJBs): su objetivo es encapsular los objetos del lado del servidor que almacena los datos. Los EJBs de entidad presentan la característica fundamental de la persistencia:
    - Persistencia gestionada por el contenedor (CMP): el contenedor se encarga de almacenar y recuperar los datos del objeto de entidad mediante el mapeo de una tabla de la base de datos. Es más difícil de mantener por el EJB server. Las herramientas del vendedor son las que se usan para realizar la correspondencia entre los campos de la base de datos y los atributos del bean. No todos los atributos son container-managed. Los atributos container-managed se definen en el descriptor de deployment y deben tener tipos correspondientes en la base de datos, estos pueden ser tipos primitivos o objetos serializables. Las clases primarias compuestas deben ser representadas por una clase especial definida por el desarrollador. Los métodos de búsqueda se utilizan para localizar beans específicos y cada vendedor los implementa a su gusto.
    - Persistencia gestionada por el bean (BMP): el propio objeto entidad se encarga, mediante una base de datos u otro mecanismo, de almacenar y recuperar los datos a los que se refiere, por lo cual, la responsabilidad de implementar los mecanismos de persistencia es del programador. Las conexiones de la base de datos se obtienen del container usando el JNDI Context (ENC) que provee acceso transaccional, conexiones JDBC y pool por medio de una fábrica de conexiones estándar.

- EJBs de Sesión (Session EJBs): gestionan el flujo de la información en el servidor. Generalmente sirven a los clientes como una fachada de los servicios proporcionados por otros componentes disponibles en el servidor. Puede haber dos tipos:
    - Con estado (stateful). Los beans de sesión con estado son objetos distribuidos que poseen un estado. El estado no es persistente, pero el acceso al bean se limita a un solo cliente. Contienen lógica de negocios y el estado específico de un cliente entre invocaciones. Mantienen un estado de negocios que puede ser compartido por los métodos del mismo bean. Para conservar recursos pueden pasar a un estado pasivo, lo que significa escribir su estado en disco. Al activarse lee su estado nuevamente.
    - Sin estado (stateless). Los beans de sesión sin estado son objetos distribuidos que carecen de estado asociado permitiendo por tanto que se los acceda concurrentemente. No se garantiza que los contenidos de las variables de instancia se conserven entre llamadas al método. Contienen métodos de negocios que se comportan como procedimientos. Operan solamente con los argumentos que se les pasan. No tienen métodos de búsqueda y no pueden ser inicializados con ningún argumento. Utilizan el JNDI ENC para obtener referencia a otro bean y para el casting de objetos deben utilizar el PortableRemoteObject.narrow()

- EJBs dirigidos por mensajes (Message-driven EJBs): son los únicos beans con funcionamiento asíncrono. Usando el Java Messaging System (JMS), se suscriben a un tema (topic) o a una cola (queue) y se activan al recibir un mensaje dirigido a dicho tema o cola. No requieren de su instanciación por parte del cliente.

Forma de Instanciar un Bean:

Un bean sea un objeto distribuido, esto significa que vive y es instanciado en el container, pero puede ser accedido por otras aplicaciones que viven en espacios de dirección diferentes.

La instancia se envuelve en un objeto especial llamado SKELETON que tiene una conexión de red con otro objeto llamado STUB que se encuentra en la aplicación cliente. El STUB mantiene un socket con el SKELETON de forma que cuando se hace una llamada a un método el STUB envía el mensaje al SKELETON, el SKELETON lo envía a la instancia del bean que lo ejecuta y retorna el resulta al SKELETON y este, finalmente, lo pasa al STUB. El SKELETON para las interfaces Home y Local/Remote es implementado por el container. El servidor de EJBs soporta JRMP o IIOP para comunicación, pero el bean siempre usa el mismo API, el JAVA RMI API. Los servidores de EJB no tienen que usar IIOP pero tienen que respetar sus restricciones.

Funcionamiento EJB:

Los EJBs se disponen en un contenedor EJB dentro del servidor de aplicaciones. La especificación describe cómo el EJB interactúa con su contenedor y cómo el código cliente interactúa con la combinación del EJB y el contenedor.

Cada EJB debe facilitar una clase de implementación Java y dos interfaces Java. El contenedor EJB creará instancias de la clase de implementación Java para facilitar la implementación EJB. Las interfaces Java son utilizadas por el código cliente del EJB. Las dos interfaces, conocidas como interfaz "home" e interfaz remoto, especifican las firmas de los métodos remotos del EJB. Los métodos remotos se dividen en dos grupos:

- Métodos que no están ligados a una instancia específica, por ejemplo aquellos utilizados para crear una instancia EJB o para encontrar una entidad EJB existente. Estos métodos se declaran en la interfaz "home".
- Métodos ligados a una instancia específica. Se ubican en la interfaz remota.

Dado que se trata simplemente de interfaces Java y no de clases concretas, el contenedor EJB genera clases para esas interfaces que actuarán como un proxy en el cliente. El cliente invoca un método en los proxies generados que a su vez sitúa los argumentos método en un mensaje y envía dicho mensaje al servidor EJB. Los proxies usan RMI-IIOP para comunicarse con el servidor EJB.

El servidor llamará a un método correspondiente a una instancia de la clase de implementación Java para manejar la llamada del método remoto.

Interface Home:

Como indicamos anteriormente, la interfaz "home" permite al código cliente manipular ciertos métodos de clase del EJB, esto es, métodos que no están asociados a ninguna instancia particular. La especificación EJB 1.1 establece el tipo de métodos de clase que se pueden definir como métodos que crean un EJB o para encontrar un EJB existente si es un "bean" de entidad. La especificación EJB 2.0 permite a los desarrolladores de aplicaciones definir nuevos métodos de clase sin limitarse a su sola creación, borrado y búsqueda.

Interface Remota:

La interfaz remota especifica los métodos de instancia públicos encargados de realizar las operaciones. Una sesión bean puede implementar múltiples interfaces, con cada interfaz apuntada por un tipo de cliente diferente. Interfaz local es para aquellos clientes que corren en la misma máquina virtual que el contenedor EJB. Interfaz remote es para clientes remotos. Frente a una consulta del cliente, el contenedor retorna un stub serializado del objeto que implementa la interfaz remote. El stub conoce como pasar llamadas a procedimientos remotos (RPCs) al servidor. Este tipo de interfaz es también un POJI.

2.1.2 JPA.

Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sí pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

2.1.3 Hibernate.

Hibernate es una herramienta de Mapeo objeto-relacional 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) que permiten establecer estas relaciones.

Como todas las herramientas de su tipo, Hibernate busca solucionar el problema de la diferencia entre los dos modelos usados hoy en día para organizar y manipular datos: El usado en la memoria de la computadora (orientación a objetos) y el usado en las bases de datos (modelo relacional). Para lograr esto permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen. Con esta información Hibernate le permite a la aplicación manipular los datos de la base operando sobre objetos, con todas las características de la OOP. Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL. Hibernate genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todas las bases de datos con un ligero incremento en el tiempo de ejecución.

Hibernate está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente. También tiene la funcionalidad de crear la base de datos a partir de la información disponible.

2.2 Ventajas de Uso de EJB 3.0 sobre EJB 2.x.

- Los descriptores de despliegue ya no son necesarios, todo puede ser integrado utilizando las anotaciones.
- El CMP (Container Managed Persistence) se simplifica mucho; este ahora tiene una estructura más al estilo Hibernate o JDO.
- Se incorporan los valores por defecto programáticos. Por ejemplo, el modelo de transacción es definido REQUIRED por default. El valor necesita ser definido solo si un valor específico diferente al default es deseado.
- El uso del chequeo de excepciones es reducido; el RemoteException no es necesario en cada método del negocio remoto.
- Ahora se permite la herencia; antes, los beans podían ser extendidos solamente del código base.
- Son soportadas las consultas nativas SQL.

2.3 Ejemplo de Aplicación en EJB 2.x.

El ejemplo consiste en una simple aplicación “Hello World”, la cual desarrolla todas las interfaces y clases requeridas por una aplicación EJB 2.0, desarrollada en NetBeans. El desarrollo de este ejemplo se subdivide en:

2.3.1 Interface Remota:

La Interface Remota soporta todos los métodos del negocio que expone el bean. Esta interface se extiende de javax.ejb.EJBObject, lo que quiere decir que el EJB object contenido generado, el cual implementa la interface remota, contendrá todos los métodos que la interface javax.ejb.EJBObject contiene. El ejemplo de código es el siguiente:

```
public interface Hello extends javax.ejb.EJBObject
{

  /**
   * El método hello() devuelve el string al cliente.
   */

  public String hello() throws java.rmi.RemoteException;
}
```

2.3.2 Interface Local:

El cliente local utiliza la Interface Local para llamar a los métodos del bean. Dentro de la Interface Remota, se tienen un método hello() el cual retorna un String “Hello World” al cliente. Es necesario implementar este método en la Clase Enterprise Bean. Existen diferencia marcadas entre la Interface Local y la Interface Remota. En este tipo se extiende de javax.ejb.EJBLocalObject, además que no se necesitan lanzar las excepciones remotas.

```
public interface HelloLocal extends javax.ejb.EJBLocalObject
{

  /**
   * The El método hello() devuelve el string al cliente.
   */

  public String hello();
}
```

2.3.3 Interface Home:

La Interface Home contiene métodos para crear y destruir EJB Objects. La implementación de esta interface es el Home Object, el cual es generado por el container.

```
public interface HelloHome extends javax.ejb.EJBHome
{

    /*
     * Este método crea el EJB Object.
     *
     * @return El nuevo EJB Object creado.
     */

    Hello create() throws java.rmi.RemoteException,
        javax.ejb.CreateException;
}
```

El cliente usa el método create() para obtener una referencia a un EJB Object. Este método también es utilizado para inicializar un bean.

La Interface Home se extiende de javax.ejb.EJBHome. EJBHome define un camino para destruir un EJB Object, por lo que no es necesario escribirlo.

2.3.4 Interface Local Home:

Es la Interface utilizada por los clientes locales que da mayor rendiemiento.

```
public interface HelloLocalHome extends javax.ejb.EJBLocalHome
{
    /*
     * Este método crea un EJB Object.
     *
     * @return El nuevo EJB Object creado.
     */

    HelloLocal create() throws javax.ejb.CreateException;
}
```

La Interface Local Home se extiende de EJBLocalHome, para esta interface la implementación no será un objeto remoto.

2.3.5 Clase Bean:

El código de la Clase Bean es el siguiente:

```
public class HelloBean implements javax.ejb.SessionBean
{
    // Los métodos EJB requeridos
    //
    public void ejbCreate()
    {
        System.out.println("ejbCreate()");
    }

    public void ejbRemove()
    {
        System.out.println("ejbRemove()");
    }

    public void ejbActivate()
    {
        System.out.println("ejbActivate()");
    }

    public void ejbPassivate()
    {
        System.out.println("ejbPassivate()");
    }

    public void setSessionContext(SessionContext ctx)
    {
        System.out.println("setSessionContext()");
    }

    //
    // Métodos del negocio
    //
    public String hello()
    {
        System.out.println("hello()");
        return "Hello, World!";
    }
}
```

El bean contiene el método ejbCreate() que corresponde con el método Create() del Home Objects, el cual no tiene parámetros.

Se tiene un método del negocio, hello(). Este método retorna “Hello World” al cliente.

Los métodos ejbActivate() y ejbPassivate() no aplican para un bean de tipo Stateless.

Cuando se destruye el bean, no es necesario limpiar, por lo que solo se tiene el método simple remove().

2.3.6 Clase Cliente:

El contenido de la clase cliente es el siguiente:

```
public class HelloClient {

        public static void main(String[] args) throws Exception {
                /*
                 * Inicia la configuración para el JNDI.
                 *
                 */
                Properties props = System.getProperties();

                /*
                 * Obtiene el contexto inicial JNDI.
                 *
                 * El contexto inicial es un punto de inicio para
                 * la conexión con el JNDI. Se elije el JNDI driver
                 * propio, la localización del servidor en la red, etc
                 * pasándolas por las propiedades del ambiente.
                */
                Context ctx = new InitialContext(props);

                /*
                 * Se obtiene una referencia al Home Object - el
                 * factory para los Hello EJB Objects
                 */
                Object obj = ctx.lookup("HelloHome");


                /*
                 * Los Home objects son objetos RMI-IIOP, y estos
                 * deben ser emitidos detro de objetos RMI-IIOP
                 * usando un convertir especial de RMI-IIOP.
                 */
                HelloHome home = (HelloHome)
                        javax.rmi.PortableRemoteObject.narrow(
                                obj, HelloHome.class);

                /*
                 * Se usa el factory para crear el Hello EJB Object
                 */
                Hello hello = home.create();

                /*
                 * Se llama al método hello() en el EJB object.  El
```

```
                * EJB object delegará la llamada al bean,
                * recibe los resultados y los retorna.
                *
                * Se imprime el resultado.
                */
                System.out.println(hello.hello());

                /*
                * Cuando se termina con el EJB Object, se elimina.
                * El container destruirá el EJB object.
                */
                hello.remove();
        }
}
```

2.3.7 Archivo ejb-jar.xml:

Este archivo contiene lo siguiente:

```
<ejb-jar>
  <enterprise-beans>
    <session>
      <ejb-name>HelloWorld</ejb-name>
      <home>examples.HelloHome</home>
      <remote>examples.Hello</remote>
      <local-home>examples.HelloLocalHome</local-home>
      <local>examples.HelloLocal</local>
      <ejb-class>examples.HelloBean</ejb-class>
      <session-type>Stateless</session-type>
    </session>
  </enterprise-beans>
</ejb-jar>
```

- <ejb-jar>: Contiene el nickname del bean en particular. Puede ser usado después en el descriptor de despliegue para referirse a las configuraciones de este bean.
- <home>: El nombre de la Interface Home.
- <remote>: El nombre de la Interface Remota.
- <local-home>: El nombre de Interface Local Home.
- <local>: El nombre de la Interface Local.
- <ejb-class>: El nombre de la Clase Bean.
- <session-type>: Si el Session Bean es Stateless o Stateful .

2.4 Ejemplo de Aplicación en EJB 3.0.

Esta aplicación consiste en un Frame que contiene los campos necesarios para insertar los datos, eliminarlos o hacer la consulta.

2.4.1 Clase Cliente.

Este procedimiento agrega un registro a la base de datos

```
private void createData() {
    try {

        Date dat = new Date();
        dat.setDate(Integer.valueOf(jTextField7.getText()));
        dat.setMonth(Integer.valueOf(jTextField6.getText()));
        dat.setYear(Integer.valueOf(jTextField5.getText()) - 1900);
        request.addBenchMark(
                Integer.valueOf(jTextField1.getText()),
                jTextField2.getText(),
                dat,
                dat,
                jTextField3.getText(),
                dat,
                jTextField4.getText());
    } catch (Exception ex) {
        System.err.println("Caught an exception in createData():");
        ex.printStackTrace();
    }
}
```

Este procedimiento elimina un registro de la base de datos de acuerdo a su llave primaria.

```
private void removeData() {
    try {
        request.removeBenchMark(Integer.valueOf(jTextField1.getText()));
    } catch (Exception ex) {
        System.err.println("Caught an exception in createData():");
        ex.printStackTrace();
    }
}
```

Este procedimiento realiza una consulta por medio de la llave primaria y retorna simplemente el nombre del BenchMark encontrado. Por cuestiones de tiempo, para hoy solo se realizo esta simple consulta, para mañana se espera extenderla para que retorne un objeto que contenga todos los datos del registro encontrado.

```
private void findNom() {
    try {
        String nombre = request.findConBench(Integer.valueOf(jTextField1.getText()));
        jTextField2.setText(nombre);
    } catch (Exception ex) {
        System.err.println("Caught an exception in createData():");
        ex.printStackTrace();
    }
}
```

Es necesario agregar todos los imports necesarios para que funcione la aplicación, además de una variable de instancia a la clase de gestión Request:

```
@EJB
   private static Request request;
```

Esta anotación se agrega a la clase al inicio de su declaración.

2.4.2 Clase Entity Bean.

Esta es la misma clase utilizada para la presentación de la aplicación la semana pasada, su código es el siguiente:

```
@Entity
@Table (name = "adpBenchMark", schema="devcustodia")
public class BenchMark implements Serializable {

   private int conBenchMark;
   private String nomBenchMark;
   private Date fechaVigencia;
   private Date fechaIngreso;
   private String usuarioIngreso;
   private Date fechaModificacion;
   private String usuarioModificacion;

   @Id
   public int getConBenchMark() {
      return conBenchMark;
   }

   public void setConBenchMark(int conBenchMark) {
      this.conBenchMark = conBenchMark;
   }

   @Column
   public String getNomBenchMark() {
      return nomBenchMark;
   }

   public void setNomBenchMark(String nomBenchMark) {
      this.nomBenchMark = nomBenchMark;
   }

   @Column (name ="fecVigencia")
   @Temporal(javax.persistence.TemporalType.DATE)
   public Date getFechaVigencia() {
      return fechaVigencia;
   }

   public void setFechaVigencia(Date fechaVigencia) {
      this.fechaVigencia = fechaVigencia;
   }

   @Column (name ="fecIngreso")
   @Temporal(javax.persistence.TemporalType.DATE)
   public Date getFechaIngreso() {
      return fechaIngreso;
   }

   public void setFechaIngreso(Date fechaIngreso) {
```

```
        this.fechaIngreso = fechaIngreso;
    }

    @Column (name ="usrIngreso")
    public String getUsuarioIngreso() {
        return usuarioIngreso;
    }

    public void setUsuarioIngreso(String usuarioIngreso) {
        this.usuarioIngreso = usuarioIngreso;
    }

    @Column (name ="fecModifico")
    @Temporal(javax.persistence.TemporalType.DATE)
    public Date getFechaModificacion() {
        return fechaModificacion;
    }

    public void setFechaModificacion(Date fechaModificacion) {
        this.fechaModificacion = fechaModificacion;
    }

    @Column (name ="usrModifico")
    public String getUsuarioModificacion() {
        return usuarioModificacion;
    }

    public void setUsuarioModificacion(String usuarioModificacion) {
        this.usuarioModificacion = usuarioModificacion;
    }

}
```

En el código anterior simplemente se encuentra la declaración de la entidad, sus propiedades y los métodos get y set para cada una de estas propiedades.

2.4.3 Clase Request.

Esta clase funciona como la interface remota, simplemente se debe agregar la anotación @Remote a su inicio para lograr esto. Contiene los procedimientos utilizados para la gestión de los datos en la aplicación:

```
@Remote
public interface Request {

    void addBenchMark(
        Integer conBencMark,
        String nomBenchMark,
        Date vigenia,
        Date ingreso,
        String usrIngreso,
        Date modifico,
        String usrModifico);

    void removeBenchMark(Integer conBencMark);

    String findConBench(int con);

}
```

2.4.4 RequestBean.

Aquí es donde básicamente se implementan los procedimientos para realizar las inserciones, borrados y consultas a la base de datos. Su código es el siguiente:

```
public class RequestBean implements Request {
    @PersistenceContext
    private EntityManager em;

   public void addBenchMark(Integer conBencMark, String nomBenchMark, Date vigenia, Date ingreso, String usrIngreso,
Date modifico, String usrModifico) {
        try {
            BenchMark bench = new BenchMark ();
            bench.setConBenchMark(conBencMark);
            bench.setNomBenchMark(nomBenchMark);
            bench.setFechaVigencia(vigenia);
            bench.setFechaIngreso(ingreso);
            bench.setUsuarioIngreso(usrIngreso);
            bench.setFechaModificacion(modifico);
            bench.setUsuarioModificacion(usrModifico);

            em.persist(bench);


        } catch (Exception e) {
            throw new EJBException(e.getMessage());
        }
    }
    public void removeBenchMark(Integer conBencMark) {
         try {
            BenchMark bench = em.find(BenchMark.class, conBencMark);
            em.remove(bench);
        } catch (Exception e) {
            throw new EJBException(e.getMessage());
        }
    }

    public String findConBench(int con) {
        String nombre;
        try {
            BenchMark bench = em.find(BenchMark.class, con);
            nombre = bench.getNomBenchMark();
        } catch (Exception e) {
            throw new EJBException(e.getMessage());
        }
        return nombre;
    }

}
```

Los tres procedimientos que contiene realizan la inserción de un registro, el borrado, y una consulta por llave primaria respectivamente. La forma en que se crea en que se crea el contexto de persistencia es mucho más sencilla que utilizando los EJB 2.x:

```
@PersistenceContext
    private EntityManager em;
```

2.4.5 Archivo de la Unidad de Persistencia.

Este archivo, como su nombre lo indica, contiene la configuración necesaria para que la aplicación obtenga la persistencia para sus objetos, en este caso, se utiliza Hibernate.

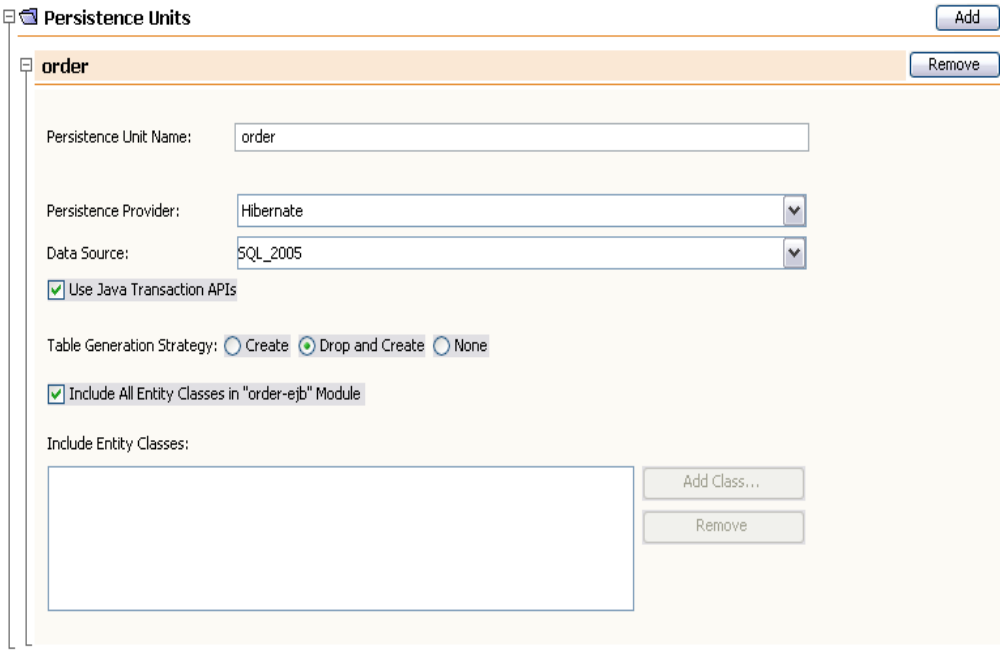

Figura #1: Unidad de Persistencia

En la imagen se muestra la forma en que se debe configurar la unidad de persistencia, primero se le indica un nombre, después se selecciona Hibernate como proveedor, ya que por defecto viene con TopLink.

Después se debe indicar la fuente de datos, la cual se puede predefinir mediante la selección de nueva fuente; aquí se le indica un nombre, y se selecciona la configuración de la conexión a la base de datos. Si esta no existe, se crea una indicando el nombre, que en este caso sería Microsoft SQL Server 2005, después se indica el URL de la base de datos, introduciendo el nombre del servidor, el puerto utilizado y la base de datos. Finalmente el usuario y la contraseña.

Finalmente, se selecciona la opción correspondiente la estrategia de generación de la tabla y la aplicación queda lista para probarse.

Capítulo 3

SOLUCIÓN IMPLEMENTADA.

3.1 Diseño Arquitectónico.

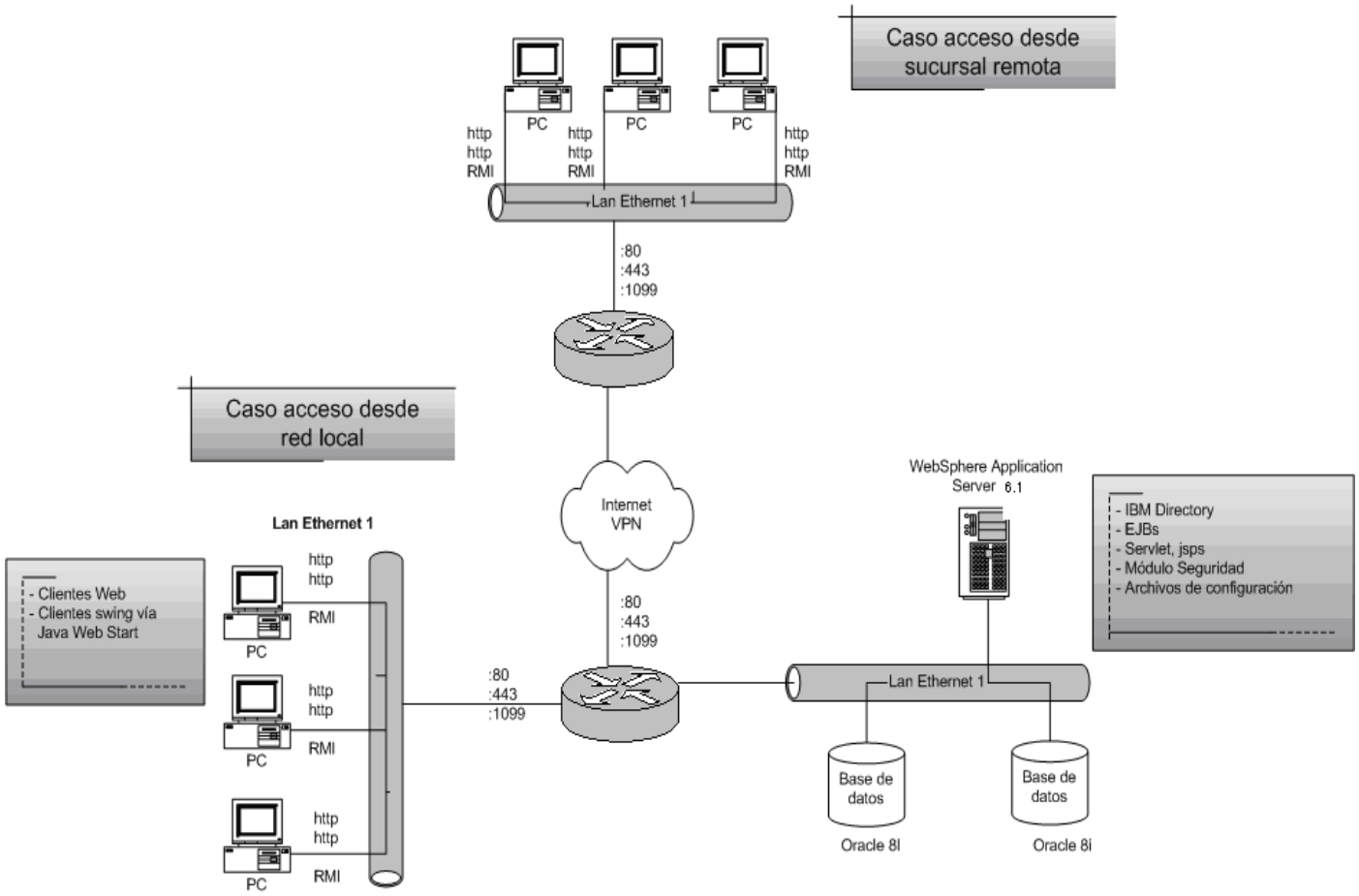

Figura #2: Diseño Arquitectónico

3.2 Componentes.

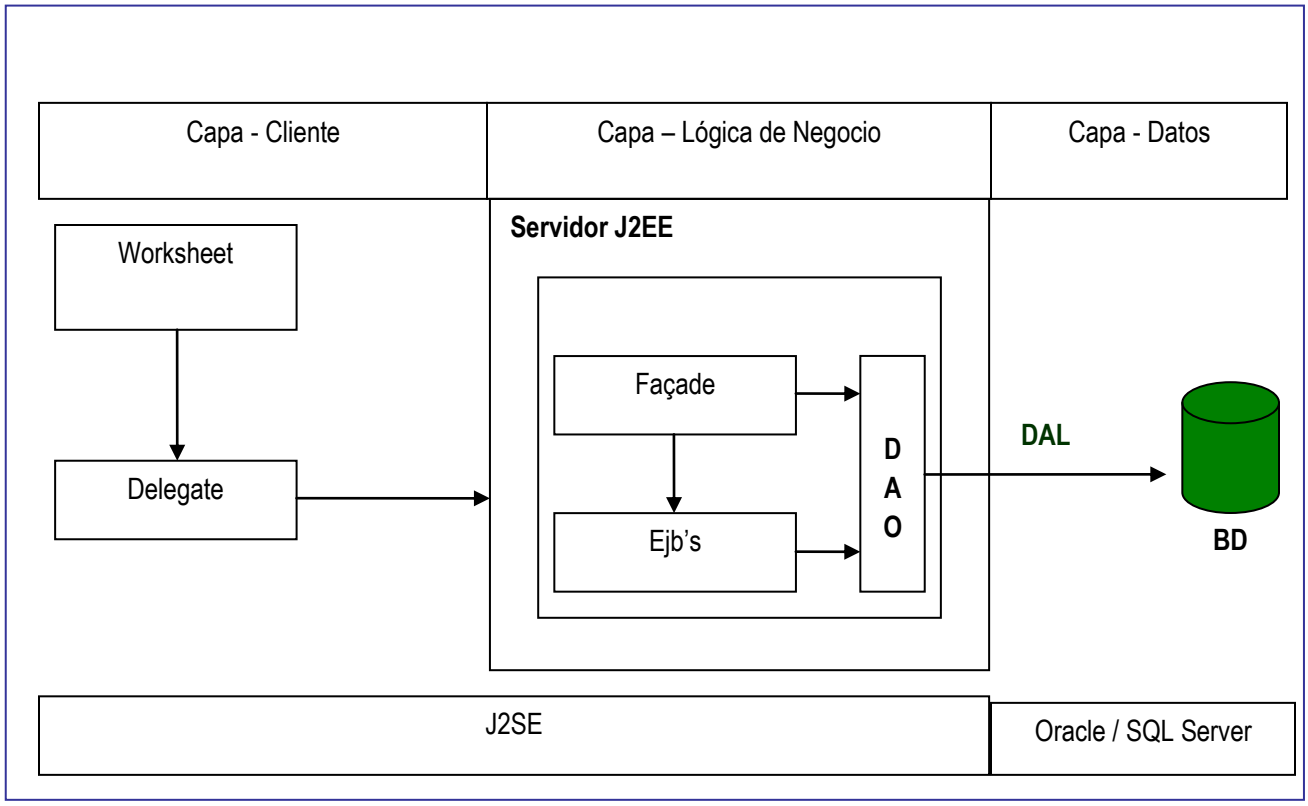

Figura #3: Diseño de Componentes del Sistema

El modelado del sistema se encuentra organizado tal y como se muestra en la figura anterior, se dividen en tres capas según sus funcionalidades. A continuación se explica el funcionamiento de cada una de las capas mostradas, sin embargo, la base del proyecto de práctica estará enfocada la forma en que se manejan los Session Beans, los Entitys Beans y el Dao (Data Access Object) en la capa de lógica de negocios, sin que esto interfiera en ningún caso con la lógica del negocio pura. Además, el trabajo también se centra extensivamente en la capa de datos.

3.2.1 Capa Cliente

En esta capa se encuentran todas las funcionalidades que van a afectar directamente al usuario de la aplicación, esto va a incluir temas como creación de interfaz, ciertos cálculos sencillos, extraer y 'setear' datos en la interfaz, entre otros.

Estas clases se diferencian físicamente de las demás por la terminación Worksheet, junto con el nombre específico que se le de a la clase.

Las clases encargadas de la comunicación de la capa cliente con la capa lógica del negocio se diferencian por la terminación Delegate. En esta se encuentran todos los métodos que deben acceder al servidor o a la base de datos, los cuales se van a comunicar por medio de una instancia que hace referencia al Facade.

3.2.2 Capa Lógica del Negocio

Esta capa se maneja directamente desde el servidor de la aplicación, cuenta con 3 importantes grupos de clases, los Session Beans, los Entitys Beans y el Dao. Esta capa intermedia tiene como función principal comunicar la parte del cliente con la información de la base de datos, ya que es aquí donde se procesan los datos que vienen del cliente para ser almacenados así como posee los métodos que extraen los datos que necesita el cliente para ser mostrados.

El funcionamiento de los Session Beans y los Entitys Beans fue mostrado en el Primer Informe de este proyecto, tanto su funcionamiento sobre EJB 2.1, como en EJB 3.0, tecnología que se busca implementar.

El Dao, es el archivo que se comunica directamente con el Dal de cada clase, en el se encuentran todas las referencias a cada una de las consultas que posea junto con los parámetros que corresponda.

3.2.3 Capa de Datos

El DAL es un archivo XML que posee en código SQLServer todas las consultas que se desean hacer en una clase en específica. La base fundamental de las clases de esta capa se muestra en el apartado 2.3 de este documento, donde se muestra el diagrama de clases completo de la capa.

En el apartado 2.4, se da la descripción específica de cada una de las clases que conforman esta capa.

3.3 Diagrama de Clases.

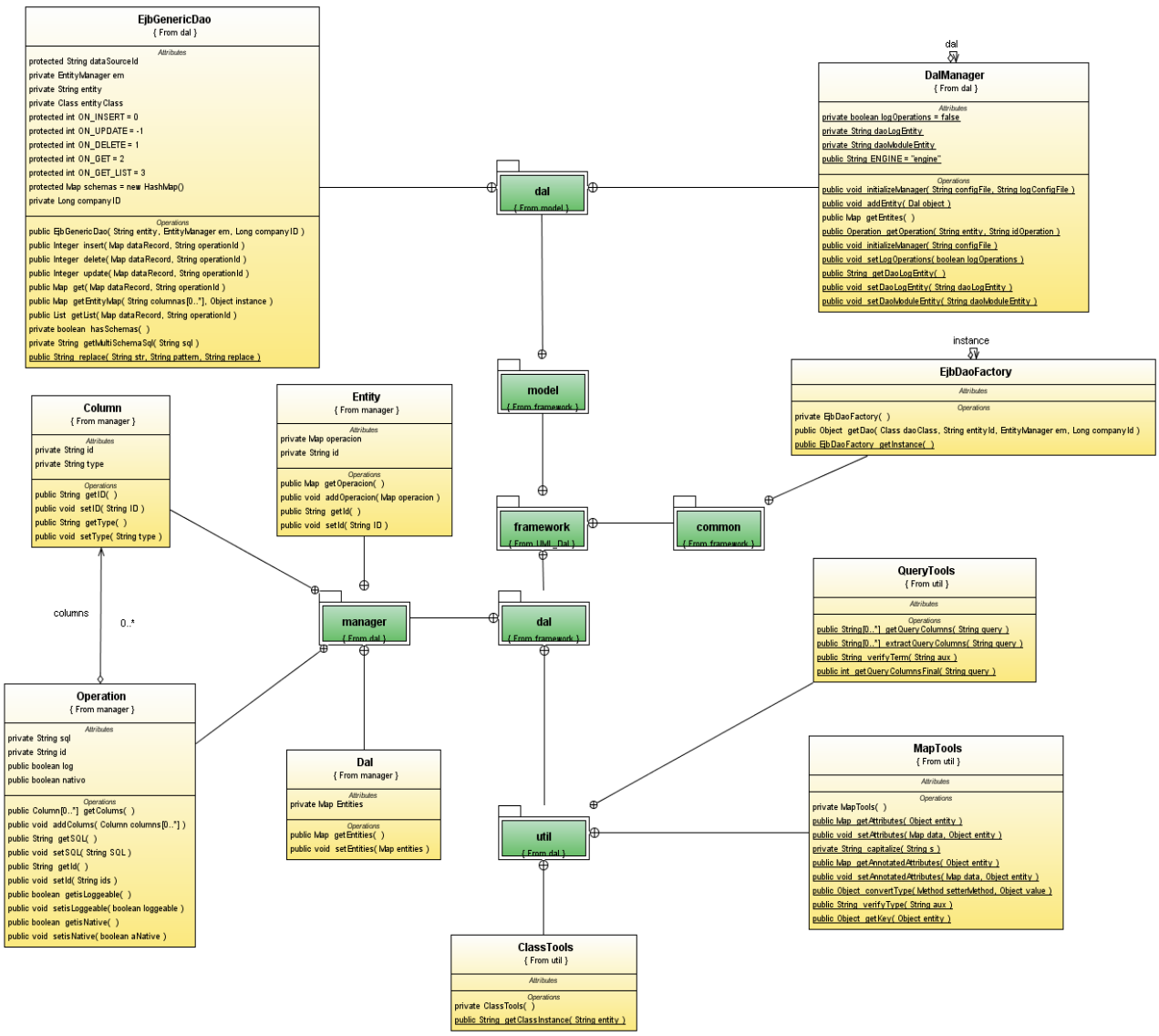

Figura #4: Diagrama de Clases

3.4 Diseño de Clases.

Para simplificar la forma en que se muestran cada una de las descripciones de las clases, se van a subdividir estas de acuerdo a la paquetería en que se encuentran dentro del proyecto.

3.4.1 Paquete framework.common

Clase	**EjbDaoFactory**
Descripción	Esta clase tiene la función de crear instancias para cada uno de los dao (Objetos de Acceso a Datos) de acuerdo al entity correspondiente, para de esta manera, una vez obtenido el dao, poder realizar las operaciones correspondientes al entity relacionado con una tabla en la base de datos.
Atributos	
Métodos	• **private EjbDaoFactory( ):** Constructor de la clase. • **public Object getDao(Class daoClass, String entityId, EntityManager em, Long companyId):** Obtiene un dao de acuerdo a la clase y lo inicializa con los argumentos recibidos. • **public EjbDaoFactory getInstance( ):** para crear una instancia de la clase.

3.4.2 Paquete framework.model.dal

Clase	**DalManager**
Descripción	Cumple la función de cargar cada uno de los archivos DAL xml en memoria que contienen las operaciones sobre cada entity relacionado a una tabla en la base de datos. Además de esto, la obtención de una operación específica.
Atributos	
Métodos	• **public void initializeManager(String configFile, String logConfigFile):** Carga el archivo .xml del DAL de acuerdo al nombre pasado por argumento • **public void addEntity(Dal object):** Agrega el entity pasado por argumento a la variable global • **public Map getEntites( ):** Obtiene el mapa de entities • **public Operation getOperation(String entity, String idOperation):** Obtiene el string de una operacion determinada de un entity, de acuerdo a la clase del entity y al id de la operacion • **public void initializeManager(String configFile):** en caso de que no se pase el archive logConfigFile

Clase	EjbGenericDao
Descripción	Contiene las operaciones genéricas de acceso a datos, como los son las inserciones, modificaciones, borrados o obtenciones de registros.
Atributos	• **private EntityManager em:** variable encargada de las persistencia de los datos. • **private String entity:** string que contiene el nombre de cada entity a cargar. • **private Class entityClass:** variable tipo clase que representa la clase de cada uno de los entibies que serán cargados. • **protected int ON_INSERT = 0:** representa el retorno para la operación de inserción. • **protected int ON_UPDATE = -1:** representa el retorno para la operación de actualización. • **protected int ON_DELETE = 1:** representa el retorno para la operación de borrado. • **protected int ON_GET = 2:** representa el retorno para la operación de obtención de un solo registro. • **protected int ON_GET_LIST = 3:** representa el retorno para la operación de obtención de una lista de registros. • **protected Map schemas = new HashMap():** mapa que contiene los esquemas utilizados por el sistema. • **private Long companyID:** id de la compañía.
Métodos	• **public EjbGenericDao(String entity, EntityManager em, Long companyID):** constructor de la clase.

	• **public Integer insert(Map dataRecord, String operationId):** inserta un objeto de acuerdo al entity relacionado, asigna los parámetros que recibe en el mapa, para este caso el "operationId" no se utiliza. • **public Integer delete(Map dataRecord, String operationId):** elimina un objeto del entity relacionado, de acuerdo a los parámetros que recibe en el mapa, para este caso el "operationId" no se utiliza. • **public Integer update(Map dataRecord, String operationId):** actualiza un objeto, de acuerdo a los parámetros que recibe en el mapa, para este caso el "operationId" no se utiliza. • **public Integer get(Map dataRecord, String operationId):** obtiene un registro del entity de acuerdo a los parámetros tomados del mapa y la operación específica. • **public List getList(Map dataRecord, String operationId):** obtiene una lista de registros del entity de acuerdo a los parámetros tomados del mapa y la operación específica. • **private boolean hasSchemas( ):** verifica si los esquemas del sistema ya han sido cargados. • **private String getMultiSchemaSql(String sql):** obtiene el esquema del string de la consulta SQL.

3.4.3 Paquete framework.dal.manager

Clase	Dal
Descripción	Clase para contener cada entity cargado de los archivos DAL
Atributos	• **private Map Entities:** mapa que contiene los entities del dal correspondiente.
Métodos	• **public Map getEntities( ):** obtiene el mapa de entities del dal correspondiente. • **public void setEntities(Map entities):** "setae" los entities recibidos en el mapa al dal.

Clase	Entity
Descripción	Clase que representa cada entity cargado de un determinado archivo DAL
Atributos	• **private Map operación:** mapa de las operaciones que se realizan sobre ese entity. • **private String id:** identificador del entity.
Métodos	• **public Map getOperacion( ):** obtiene el mapa de las operaciones del entity. • **public void addOperacion(Map operacion):** agrega una operación al mapa de operaciones del entity en cuestión. • **public String getId( ):** obtiene el identificador del entity. • **public void setId(String ID):** "setea" el identificador al entity.

Clase	Operation
Descripción	Clase que representa las operaciones que contiene cada entity
Atributos	• **private String sql:** string que contiene el quero de la consulta SQL de la operación. • **private String id:** identificador de la operación. • **public boolean log:** indica si la operación es loggeable o no. • **public boolean nativo:** indica si la consulta es nativa o no.
Métodos	• **public Column[0..*] getColums( ):** obtiene cada una de las columnas que contiene la operación. • **public void addColums(Column columns[0..*]):** agrega columnas a la operación. • **public String getSQL( ):** obtiene la sentencia SQL. • **public void setSQL(String SQL):** "setea" la sentencia SQL • **public String getId( ):** obtiene el identificador de la operación • **public void setId(String ids):** "setea" el identificador de la operación • **public boolean getisLoggeable( ):** si la operación el loggeable • **public void setisLoggeable(boolean loggeable):** "setea" si la operación va a ser loggeable • **public boolean getisNative( ):** si la operación es nativa. • **public void setisNative(boolean aNative):** "setea" si la

	operación va a ser nativa

Clase	**Column**
Descripción	Case para las columnas de cada operación de cada entity
Atributos	• **private String id:** identificador de la columna • **private String type:** tipo de dato de la columna
Métodos	• **public String getID( ):** obtiene el identificador de la columna • **public void setID(String ID):** “setea” el identificador de la columna. • **public String getType( ):** obtiene el tipo de la columna. • **public void setType(String type):** “setea” el tipo de la columna.

3.4.4 Paquete framework.dal.util

Clase	ClassTools
Descripción	Obtiene la dirección de un entity dentro de la paquetería del proyecto de acuerdo a una llave en un archivo .properties
Atributos	
Métodos	• **private ClassTools( ):** constructor de la clase. • **public String getClassInstance(String entity): o**btiene la dirección del entity dentro de la paquetería del proyecto de acuerdo a la llave que recibe por parámetro.

Clase	MapTools
Descripción	Realiza la conversión entre mapas y objetos de tipo entity, tanto para copiar los archivos de un mapa a una instancia de un entity como viceversa
Atributos	
Métodos	• **private MapTools( ):** constructor de la clase. • **public Map getAnnotatedAttributes(Object entity):** retorna los valores de las propiedades del entity enviado en el argumento, cada atributo debe tener el método setter correspondiente, siguiente el estándar de nombres, el mapa retornado contiene tantos valores como atributos tenga el entity y la llave de cada valor será igual al nombre del atributo en mayúsculas, los tipos de los valores serán iguales a los atributos del entity.

	• **public void setAnnotatedAttributes(Map data, Object entity):** inicializa el entity con los valores contenidos en data, si el mapa no contiene un valor llamado de la misma forma que el atributo del entity en mayusculas, este lo inicializa en nulo. • **public Object convertType(Method setterMethod, Object value):** verifica que el tipo de dato de "value" y el tipo que debe recivir el método "setterMethod" sean compatibles, si no los son, hace la conversión de "value" • **public Object getKey(Object entity):** obtiene la llave primaria del entity.

Clase	**QueryTools**
Descripción	Obtiene las columnas que se solicitan en una consulta desde el string de la consulta.
Atributos	
Métodos	• **public String[0..*] getQueryColumns(String query):** obtiene del query recibido la zona donde se ubican las columnas de la consulta. • **public String[0..*] extractQueryColumns(String query):** separa cada una de las columnas de la zona y las retorna en un arreglo de string.

	• **public String verifyTerm(String aux):** valida la separación de cada columna. • **public int getQueryColumnsFinal(String query):** recibe un string con la sentencia de la consulta.

Capítulo 4

CONCLUSIONES Y COMENTARIOS.

4.1 Objetivos Propuestos

Cada uno de los objetivos pensados y propuestos por los que se iba a trabajar en este proyecto se llevaron acabo con el desarrollo de la práctica de manera satisfactoria, en donde cada uno de los puntos citados se finalizaron de manera exitosa cumpliendo así con el principal objetivo de la práctica de ayudar al desarrollo del proyecto del Sistema de Intermediación de Valores mediante mejoras en la arquitectura del producto.

4.2 Productos

Al estar este proyecto de práctica de especialidad orientado a la estructura específica de la arquitectura del producto base de la empresa, no existe un producto finalizado como tal que haya sido presentado, aún así, según las tareas realizadas podemos encontrar una subdivisión de 2 subproductos que se llevaron acabo durante la elaboración del proyecto, los cuales son:

- Creación de una nueva capa de acceso a datos que funciona de manera más transparente y simplificada, otorgando así al sistema una mejora tangible en cuanto a la implementación del acceso a datos. Para este punto, la nueva capa ha sido probada solamente con el Módulo de Seguridad, sin embargo, el propósito es continuar con un plan bien definido para continuar con los demás módulos.
- Funcionamiento completo del Módulo de seguridad, el cual ha servido de referencia para observar los puntos relevantes de utilizar estas nuevas tecnologías, ya aplicadas al producto como tal.

4.3 Experiencia y Conclusiones

- Como uno de los puntos más importantes durante la realización del presiente proyecto, se encuentra el hecho de que la práctica se basó completamente en aspectos estructurales del sistema de la empresa en el lenguaje Java, por lo que la lógica del negocio prácticamente no fue utilizada hasta este punto. Es por esto, que la fase de integración que el mercado laboral de la empresa aún no ha sido iniciada.
- Es interesante la seguridad que una aplicación de este tipo demanda, en donde se tiene aún más responsabilidad sobre el buen funcionamiento de la misma por el tema empresarial que maneja, de manera que el uso del Módulo de Seguridad agrega un esfuerzo extra en cuanto a la eficiencia requerida.
- Para lograr los objetivos propuestos, se tuvo que trabajar muy estrechamente tanto con la J2EE de Java (Especificación actual utilizada por la empresa), como con la nueva JEE5 (nueva especificación que contiene EJB 3.0), lo cuál provocó mucho aprendizaje en este lenguaje, punto muy favorable debido al crecimiento en el mercado del uso de Java.
- Adaptarse a una estructura organizacional bien definida dentro de una empresa con tanta trayectoria como LiderSoft ha proporcionado también mucho enriquecimiento, el cual es necesario cuando se inicia una carrera profesional ya en el mercado.
- Al ser un proyecto tan grande y dividido en tantos recursos a la vez, es interesante ver como el trabajo en grupo viene a ser de gran importancia, ya que su coordinación y rendimiento debe de estar muy bien medido, para así lograr el cumplimiento de los objetivos propuestos en los tiempos e integridad requeridos.
- Por último esta experiencia de práctica de especialidad brinda una amplia visión de lo que es el trabajo real en una empresa, otorgando formación de una manera muy profesional para desempeñar las labores de manera exitosa como profesionales, cumpliendo con las normas del mercado.

Capítulo 5

ANEXOS.

5.1 Informes Semanales

Informe Semanal de Avance # 01

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 10/12/2007 al 14/12/2007_____

A) Actividades planeadas para esta semana

1. Comprensión y lectura de documentos introductorios, como:

- Estándares de Programación de LiderSoft.
- Tecnologías utilizadas en el proceso de desarrollo de software Java.
- Arquitectura general de la empresa.
- Manuales de aplicaciones propias.

B) Actividades realizadas para esta semana según lo planeado

1. Comprensión y lectura de documentos introductorios, como:

- Estándares de Programación de LiderSoft.
- Tecnologías utilizadas en el proceso de desarrollo de software Java.
- Arquitectura general de la empresa.
- Manuales de aplicaciones propias.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Comprensión y lectura de documentos introductorios al sistema de negocios de la empresa, como:

- Algunos conceptos bursátiles.
- Introducción a la bolsa.

- Puestos de bolsa.
- Forma de operación del servicio básico de custodia.
- Reglamento de custodia.
- Implementación del reglamento de custodia.

Informe Semanal de Avance # 02

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 17/12/2007_ al 21/12/2007_____

A) Actividades planeadas para esta semana

1. Comprensión y lectura de documentos introductorios al sistema de negocios de la empresa, como:

- Algunos conceptos bursátiles.
- Introducción a la bolsa.
- Puestos de bolsa.
- Forma de operación del servicio básico de custodia.
- Reglamento de custodia.
- Implementación del reglamento de custodia.

B) Actividades realizadas para esta semana según lo planeado

1. Comprensión y lectura de documentos introductorios al sistema de negocios de la empresa, como:

- Algunos conceptos bursátiles.
- Introducción a la bolsa.
- Puestos de bolsa.
- Forma de operación del servicio básico de custodia.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

- Reglamento de custodia.
- Implementación del reglamento de custodia.

E) Actividades por hacer la próxima semana

1. Continuar las tareas pendientes.

Informe Semanal de Avance # 03

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 02/01/2008_ al 04/01/2008_____

A) Actividades planeadas para esta semana

1. Continuar las tareas pendientes, las cuales son: comprensión y lectura de documentos introductorios al sistema de negocios de la empresa.

- Reglamento de custodia.
- Implementación del reglamento de custodia.

B) Actividades realizadas para esta semana según lo planeado

1. Continuar las tareas pendientes, las cuales son: comprensión y lectura de documentos introductorios al sistema de negocios de la empresa.

- Reglamento de custodia.
- Implementación del reglamento de custodia.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Introducción a los objetivos del Proyecto, explicación de este y tareas a realizar.

- Información básica sobre Hibernate y descarga de aplicaciones necesarias.
- Creación de un ejemplo para probar las aplicaciones.
- Información sobre el uso de la Persistencia por JPA.
- Información sobre el Hibernate EntityManager y Hibernate Annotations.
- Compilación de la información recolectada para realizar un documento de avance.

Informe Semanal de Avance # 04

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 07/01/2008_ al 11/01/2008_____

A) Actividades planeadas para esta semana

1. Introducción a los objetivos del Proyecto, explicación de este y tareas a realizar.

- Información básica sobre Hibernate y descarga de aplicaciones necesarias.
- Creación de un ejemplo para probar las aplicaciones.
- Información sobre el uso de la Persistencia por JPA.
- Información sobre el Hibernate EntityManager y Hibernate Annotations.
- Compilación de la información recolectada para realizar un documento de avance.

B) Actividades realizadas para esta semana según lo planeado

1. Introducción a los objetivos del Proyecto, explicación de este y tareas a realizar.

- Información básica sobre Hibernate y descarga de aplicaciones necesarias.
- Creación de un ejemplo para probar las aplicaciones.
- Información sobre el uso de la Persistencia por JPA.
- Información sobre el Hibernate EntityManager y Hibernate Annotations.
- Compilación de la información recolectada para realizar un documento de avance.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Recolectar más información sobre:

- Anotaciones.
- Recopilación de información sobre Hibernate Validator.
- Resolución de un ejemplo.

Informe Semanal de Avance # 05

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 14/01/2008_ al 18/01/2008_____

A) Actividades planeadas para esta semana

1. Recolectar más información sobre:

- Anotaciones.
- Recopilación de información sobre Hibernate Validator.
- Resolución de un ejemplo básico.

B) Actividades realizadas para esta semana según lo planeado

1. Recolectar más información sobre:

- Anotaciones.
- Recopilación de información sobre Hibernate Validator.
- Resolución de un ejemplo básico.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Resolución de un ejemplo que integre Hibernate desde JPA, con operaciones básicas.

- Configuración de Aplicaciones.
- Uso de anotaciones y validaciones.

2. Análisis y comprensión de la estructura del módulo actual de la empresa en el que se desarrollará el proyecto.

Informe Semanal de Avance # 06

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 21/01/2008_ al 25/01/2008_____

A) Actividades planeadas para esta semana

1. Resolución de un ejemplo que integre Hibernate desde JPA, con operaciones básicas.

- Configuración de Aplicaciones.
- Uso de anotaciones y validaciones.

2. Análisis y comprensión de la estructura del módulo actual de la empresa en el que se desarrollará el proyecto.

B) Actividades realizadas para esta semana según lo planeado

1. Resolución de un ejemplo que integre Hibernate desde JPA, con operaciones básicas.

- Configuración de Aplicaciones.
- Uso de anotaciones y validaciones.
- 2. Análisis y comprensión de la estructura del módulo actual de la empresa en el que se desarrollará el proyecto.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Integración de JPA y EntityManager desde EJB 2.0:

- Ejemplo de una aplicación en EJB 2.0.
- Se concluyó que la integración con EJB 2.0 no es posible.

2. Plan de migración de EJB 2.x a EJB 3.0.

- Ejemplo de una aplicación en EJB 3.0.

Informe Semanal de Avance # 07

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 28/01/2008_ al 01/02/2008_____

A) Actividades planeadas para esta semana

1. Integración de JPA y EntityManager desde EJB 2.0:

- Ejemplo de una aplicación en EJB 2.0.
- Se concluyó que la integración con EJB 2.0 no es posible.

2. Plan de migración de EJB 2.x a EJB 3.0.

- Ejemplo de una aplicación en EJB 3.0.

B) Actividades realizadas para esta semana según lo planeado

1. Integración de JPA y EntityManager desde EJB 2.0:

- Ejemplo de una aplicación en EJB 2.0.
- Se concluyó que la integración con EJB 2.0 no es posible.

2. Plan de migración de EJB 2.x a EJB 3.0.

- Ejemplo de una aplicación en EJB 3.0.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Aplicación utilizando EJB 3.0, JPA y WebSphere 6.1 como servidor de aplicaciones.

- Implementación de la aplicación en Rational Application Developer 7.0.
- Se concluyó que no existe soporte para EJB 3.0 en el ambiente de desarrollo mencionado.

2. Implementación de la aplicación en Rational Application Developer 7.5.

- Descarga de la versión.
- Realización de la aplicación.

Informe Semanal de Avance # 08

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 04/02/2008_ al 08/02/2008_____

A) Actividades planeadas para esta semana

1. Aplicación utilizando EJB 3.0, JPA y WebSphere 6.1 como servidor de aplicaciones.

- Implementación de la aplicación en Rational Application Developer 7.0.
- Se concluyó que no existe soporte para EJB 3.0 en el ambiente de desarrollo mencionado.

2. Implementación de la aplicación en Rational Application Developer 7.5.

- Descarga de la versión.
- Realización de la aplicación.

B) Actividades realizadas para esta semana según lo planeado

1. Aplicación utilizando EJB 3.0, JPA y WebSphere 6.1 como servidor de aplicaciones.

- Implementación de la aplicación en Rational Application Developer 7.0.
- Se concluyó que no existe soporte para EJB 3.0 en el ambiente de desarrollo mencionado.

2. Implementación de la aplicación en Rational Application Developer 7.5.

- Descarga de la versión.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

2. Implementación de la aplicación en Rational Application Developer 7.5.

- Realización de la aplicación.

E) Actividades por hacer la próxima semana

1. Integración de un módulo actual del sistema para realizar la prueba de migración a EJB 3.0

2. Modificación del esquema actual del módulo para su funcionamiento con EJB 3.0.

Informe Semanal de Avance # 09

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 11/02/2008_ al 15/02/2008_____

A) Actividades planeadas para esta semana

1. Integración de un módulo actual del sistema para realizar la prueba de migración a EJB 3.0

2. Modificación del esquema actual del módulo para su funcionamiento con EJB 3.0.

B) Actividades realizadas para esta semana según lo planeado

1. Integración de un módulo actual del sistema para realizar la prueba de migración a EJB 3.0

2. Modificación del esquema actual del módulo para su funcionamiento con EJB 3.0.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Inicio de la migración del módulo seleccionado:

- Migración de los Entity Beans.
- Migración de los Session Beans.

2. Reestructuración de la capa de acceso a datos.

Informe Semanal de Avance # 10

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 18/02/2008_ al 22/02/2008_____

A) Actividades planeadas para esta semana

1. Inicio de la migración del módulo seleccionado:

- Migración de los Entity Beans.
- Migración de los Session Beans.

2. Reestructuración de la capa de acceso a datos.

B) Actividades realizadas para esta semana según lo planeado

1. Inicio de la migración del módulo seleccionado:

- Migración de los Entity Beans.
- Migración de los Session Beans.

2. Reestructuración de la capa de acceso a datos (en proceso).

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

2. Reestructuración de la capa de acceso a datos (en proceso).

E) Actividades por hacer la próxima semana

1. Finalizar la reestructuración de la capa de acceso a datos.

2. Realización de pruebas de la capa de acceso a datos.

Informe Semanal de Avance # 11

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 25/02/2008_ al 29/02/2008_____

A) Actividades planeadas para esta semana

1. Finalizar la reestructuración de la capa de acceso a datos.

2. Realización de pruebas de la capa de acceso a datos.

B) Actividades realizadas para esta semana según lo planeado

1. Finalizar la reestructuración de la capa de acceso a datos.

2. Realización de pruebas de la capa de acceso a datos.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

E) Actividades por hacer la próxima semana

1. Pruebas y corrección de errores sobre operaciones básicas de acceso a datos.

Informe Semanal de Avance # 12

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 03/03/2008_ al 07/03/2008_____

A) Actividades planeadas para esta semana

1. Pruebas y corrección de errores sobre operaciones básicas de acceso a datos.

B) Actividades realizadas para esta semana según lo planeado

1. Pruebas y corrección de errores sobre operaciones básicas de acceso a datos (aún en proceso).

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

1. Finalización de pruebas y corrección de errores sobre operaciones básicas de acceso a datos.

E) Actividades por hacer la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

Informe Semanal de Avance # 13

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 10/03/2008_ al 14/03/2008_____

A) Actividades planeadas para esta semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

B) Actividades realizadas para esta semana según lo planeado

C) Actividades realizadas durante esta semana, que no estaban planeadas

1. Corrección de errores propios del módulo de seguridad a la hora de integrarlo con la nueva tecnología de EJB 3.0 y JPA.

D) Actividades que quedaron pendientes para la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

E) Actividades por hacer la próxima semana

1. Continuación de la corrección de errores propios del módulo de seguridad a la hora de integrarlo con la nueva tecnología de EJB 3.0 y JPA.

2. Integración de la paginación en las consultas.

3. Integración de consultas dinámicas.

Informe Semanal de Avance # 14

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 17/03/2008_ al 21/03/2008_____

A) Actividades planeadas para esta semana

1. Continuación de la corrección de errores propios del módulo de seguridad a la hora de integrarlo con la nueva tecnología de EJB 3.0 y JPA.

2. Integración de la paginación en las consultas.

3. Integración de consultas dinámicas.

B) Actividades realizadas para esta semana según lo planeado

1. Continuación de la corrección de errores propios del módulo de seguridad a la hora de integrarlo con la nueva tecnología de EJB 3.0 y JPA.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

E) Actividades por hacer la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

Informe Semanal de Avance # 15

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 24/03/2008_ al 28/03/2008_____

A) Actividades planeadas para esta semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

B) Actividades realizadas para esta semana según lo planeado

C) Actividades realizadas durante esta semana, que no estaban planeadas

1. Realización de pruebas de inserción, borrado, modificado y consultas.

2. Corrección de errores encontrados en las pruebas.

D) Actividades que quedaron pendientes para la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

E) Actividades por hacer la próxima semana

1. Continuación de las pruebas y corrección de errores.

Informe Semanal de Avance # 16

Nombre: Rafael Ángel Sánchez Saborío.____
Semana del: 31/03/2008_ al 04/04/2008_____

A) Actividades planeadas para esta semana

1. Continuación de las pruebas y corrección de errores.

B) Actividades realizadas para esta semana según lo planeado

1. Continuación de las pruebas y corrección de errores.

C) Actividades realizadas durante esta semana, que no estaban planeadas

D) Actividades que quedaron pendientes para la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

E) Actividades por hacer la próxima semana

1. Integración de la paginación en las consultas.

2. Integración de consultas dinámicas.

Lidersoft
Soluciones y Servicios.

5.2 Plan de Trabajo.

		Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Proyecto Practica de Especialidad**	85 days?	**12/10/07 8:00 AM**	**4/4/08 5:00 PM**		
2	✓	**Organización del Proyecto**	54.875 days	**12/10/07 2:00 PM**	**2/25/08 1:00 PM**		
3	✓	Reunión Inicial (Definición del proyecto)	0.25 days	1/2/08 8:00 AM	1/2/08 10:00 AM		Cristopher Barboza;Rafael Sanchez;Jose David Gonzales
4	✓	Definición del Cronograma	0.25 days	1/2/08 10:00 AM	1/2/08 1:00 PM	3	Cristopher Barboza;Rafael Sanchez;Jose David Gonzales
5	✓	Primera Reunión con el Profesor	0.125 days	2/25/08 9:00 AM	2/25/08 10:00 AM		Cristopher Barboza;Profesor ITCR - Abel Mendez;Rafael Sanchez
6	✓	Definción de contenido del primer informe	0.25 days	2/25/08 10:00 AM	2/25/08 1:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
7	✓	Revisión del Borrador del Primer Informe	0 days	12/10/07 2:00 PM	12/10/07 2:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
8	✓	Definición de contenido del segundo informe	0 days	12/10/07 2:00 PM	12/10/07 2:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
9	✓	Revisión del Borrador del Segundo Informe	0 days	12/10/07 2:00 PM	12/10/07 2:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
10		Definición de Informe Final	0 days	12/10/07 2:00 PM	12/10/07 2:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
11		Revisión del Borrador del Informe Final	0 days	12/10/07 2:00 PM	12/10/07 2:00 PM		Profesor ITCR - Abel Mendez;Rafael Sanchez
12		**Programación**	70 days?	**12/10/07 8:00 AM**	**3/14/08 5:00 PM**		
13		**Tareas de Desarrollo**	70 days?	**12/10/07 8:00 AM**	**3/14/08 5:00 PM**		
14	✓	Comprensión y lectura de documentos introductorios	5 days	12/10/07 8:00 AM	12/14/07 5:00 PM		Rafael Sanchez
15	✓	Documentos introductorios al sistema de negocios de la empresa	5 days	12/17/07 8:00 AM	12/21/07 5:00 PM		Rafael Sanchez
16	✓	Introducción a los objetivos del Proyecto	0.75 days	1/2/08 10:00 AM	1/2/08 5:00 PM	3	Cristopher Barboza;Rafael Sanchez;Jose David Gonzales
17	✓	Recolección de información básica sobre Hibernate	7 days	1/3/08 8:00 AM	1/11/08 5:00 PM		Rafael Sanchez
18	✓	Recopilación de información sobre Hibernate EntityManager	1 day?	1/14/08 8:00 PM	1/15/08 5:00 PM		Rafael Sanchez
19	✓	Recopilación de Información sobre Hibernate Anotaciones	1 day?	1/15/08 8:00 AM	1/15/08 5:00 PM		Rafael Sanchez
20	✓	Recopilación de Información sobre Hibernate Validator	1 day?	1/16/08 8:00 AM	1/16/08 5:00 PM		Rafael Sanchez
21	✓	Resolución de un ejemplo básico	2 days	1/17/08 8:00 AM	1/18/08 5:00 PM		Rafael Sanchez
22	✓	Resolución de un ejemplo que integre Hibernate desde JPA, con operaciones básicas	5 days	1/21/08 8:00 AM	1/25/08 5:00 PM		Rafael Sanchez
23	✓	Integración de JPA y EntityManager desde EJB 2.0	3 days	1/28/08 8:00 AM	1/30/08 5:00 PM		Rafael Sanchez
24	✓	Ejemplo de Migración de EJB 2.x a EJB 3.0	2 days	1/31/08 8:00 AM	2/1/08 5:00 PM		Rafael Sanchez
25	✓	Aplicación utilizando EJB 3.0, JPA y WebSphere 6.1	10 days	2/4/08 8:00 AM	2/15/08 5:00 PM		Rafael Sanchez;Jose David Gonzales[50%]
26	✓	Integración de un módulo actual del sistema para realizar la prueba de migración a EJB 3.0	2 days	2/11/08 8:00 AM	2/12/08 5:00 PM		Rafael Sanchez;Jose David Gonzales
27	✓	Modificación del esquema actual del módulo para su funcionamiento con EJB 3.0	1.5 days	2/13/08 8:00 AM	2/14/08 1:00 PM	26	Rafael Sanchez;Jose David Gonzales[50%]
28	✓	Migración de Entity Beans del módulo seleccionado	2 days	2/18/08 8:00 AM	2/19/08 5:00 PM	27	Rafael Sanchez
29	✓	Migración de los Session Beans del módulo seleccionado	3 days	2/20/08 8:00 AM	2/22/08 5:00 PM	28	Rafael Sanchez
30	✓	Reestructuración de la capa de acceso a datos	20 days	2/13/08 8:00 AM	3/11/08 5:00 PM	26	Rafael Sanchez;Jose David Gonzales[50%]
31	✓	Inicio de pruebas sobre operaciones básicas	3 days	2/27/08 8:00 AM	2/29/08 5:00 PM	27	Rafael Sanchez;Jose David Gonzales[50%]
32		Integración de la paginación al módulo	2 days	3/12/08 8:00 AM	3/13/08 5:00 PM	30	Rafael Sanchez
33		Realización de consultas dinámicas	3 days	3/12/08 8:00 AM	3/14/08 5:00 PM	30	Rafael Sanchez
34		**Pruebas de Aplicación**	3 days	**3/10/08 8:00 AM**	**4/1/08 3:24 PM**		
35	✓	Pruebas de Funcionalidad	14 days	3/10/08 8:00 AM	3/20/08 11:12 AM		Rafael Sanchez;Jose David Gonzales[50%]
36	✓	Correcciones y Ajustes	7 days	3/19/08 8:00 AM	4/1/08 3:24 PM	35	Rafael Sanchez;Jose David Gonzales[50%]
37		**Documentación**	7 days	**3/28/08 8:00 AM**	**4/8/08 3:24 PM**		
38	✓	Documentación de producto	5 days	3/28/08 8:00 AM	4/8/08 3:24 PM	36	Rafael Sanchez
39		**Entrega**	5 days	**3/4/08 8:00 AM**	**4/9/08 3:24 PM**		
40		Entrega	1 day	4/4/08 8:00 AM	4/9/08 3:24 PM	38	Rafael Sanchez

5.3 Hoja de Información.

Información del Estudiante:

Nombre: Rafael Ángel Sánchez Saborío.

Cédula o No. Pasaporte: 6 0339 0972.

Carné ITCR: 200227864

Dirección de su residencia en época lectiva: Contiguo al Bar El Coquito, Santa Clara de Florecia de San Carlos, Alajuela.

Dirección de su residencia en época no lectiva: Contiguo a Bloquera Abangares, Barrio Palo Hueco, Las Juntas de Abangares, Guanacaste.

Teléfono en época lectiva: 8851 1878

Teléfono época no lectiva: 8851 1878 – 2662 1960

Email: rssaborio@hotmail.com

Fax:

Información del Proyecto:

Nombre del Proyecto: Desarrollo de Funcionalidades para el Sistema de Intermediación de Valores para Fondos de Inversión y Puestos de Bolsa

Profesor Asesor: Abel Méndez

Horario de trabajo del estudiante: De Lunes a Viernes de 8 a.m. a 5:30 p.m.

Información de la Empresa:

Nombre: Lidersoft Internacional

Zona: San José Centro

Dirección: 25 mts Sur de la Iglesia del Carmen

Teléfono: 2258 3220

Actividad Principal: Desarrollo de Software para entidades financieras.