Análisis, diseño e Implementación de una aplicación J2EE

Proyecto de reparaciones y control de ventas
Naoru Project

Alumno: **David Ferreiro Vilar**
E.T. Informática de Gestión

Consultor: **Jose Juan Rodriguez**
17 de Enero de 2011

Introducción

Este trabajo de fin de carrera pretende sintetizar los conocimientos adquiridos a lo largo de la carrera para ponerlos en práctica en este proyecto e intentar aportar un proyecto totalmente diferente a los que hay actualmente.

El objetivo principal de este trabajo es desarrollar un sistema de consultas de reparación online y además aprovechar para tener un historial de compras y reparaciones tanto para los clientes como los operarios de la tienda o los técnicos.

Para la implementación se ha utilizado la tecnología que nos proporciona el framework Struts2, el cual nos permitirá interactuar i crear una aplicación con JSP's de una manera profesional i sencilla.

Objetivos del TFC

El objetivo principal del TFC consiste en un llevar a cabo un proyecto de principio a fin, des del análisis del mismo, pasando por el diseño y finalmente el desarrollo e implementación.

Como objetivo inherente al citado, aprenderemos y profundizaremos en el uso de las tecnologías J2EE, Java i SQL.

Y como objetivo relacionado con el proyecto elegido, se ha desarrollado una aplicación Web, con la cual, tanto clientes como operarios y técnicos (en nuestro caso de una tienda de informática aunque con múltiples proyecciones empresariales) puedan gestionar las ventas (no contabilidad), las reparaciones y los clientes que frecuentan la tienda.

Hacer todo lo anterior de la manera más ordenada, accesible y útil tanto para futuras modificaciones como para el diseño de cara los usuarios.

Resumen Funcional

El proyecto esta dividido en diferentes funcionalidades.

-Gestión de Usuarios: Se dan de alta y de baja los usuarios en el sistema, asimismo se pueden modificar los datos. Al mismo tiempo se pueden buscar los datos de un cliente por su Nif.

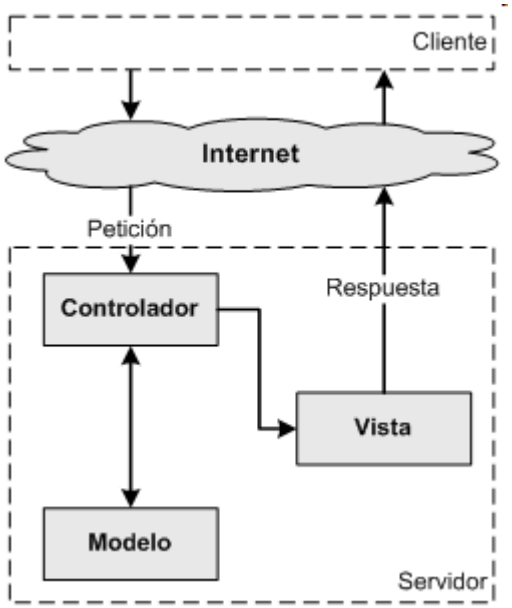
-Gestión de Compras: Se dan de alta y se borran las compras si es necesario, también se pueden modificar si se hace alguna devolución o si se ha detectado algún error. Al mismo tiempo se puede ver un histórico de compras por cliente.

-Gestión de Reparaciones: Por último y no menos importante, la opción de dar de alta y baja de reparaciones, modificar sus datos, entre los cuales se da relevancia a las modificaciones en “tiempo real” que sufran los aparatos llevados, así en todo momento se sabe el Estado y en el apartado de comentarios se puede ver que falla, cual es la solución y posible presupuesto.

Arquitectura

Una de las características mas importantes sin tener en cuenta lo que es el proyecto en si es la implementación de un patrón MVC (Modelo-Vista-Controlador).

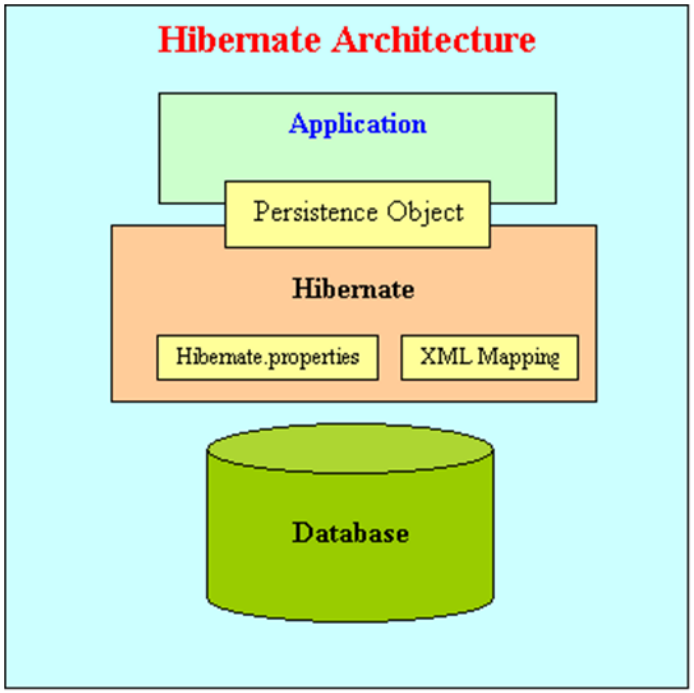
Este sistema separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Además nos permite acercarnos más a una programación orientada a las pruebas ya que se pueden establecer clases unitarias de pruebas



- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista transforma el modelo en una página Web que permite al usuario interactuar con ella.
- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

Persistencia I

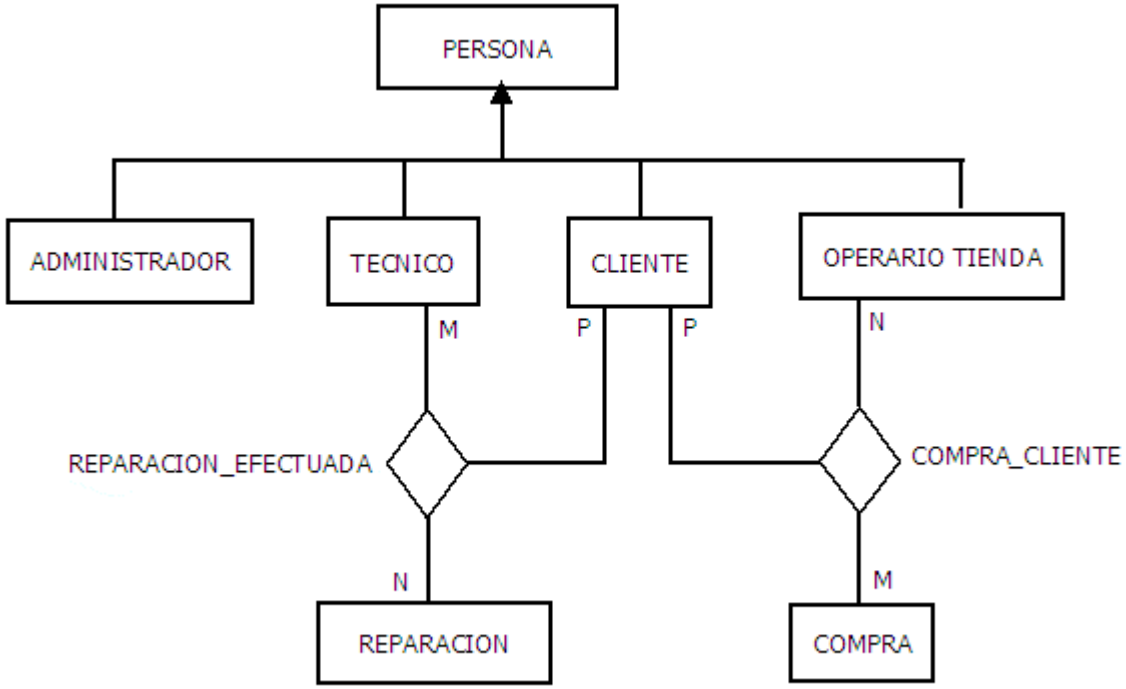
Para la persistencia de datos se ha optado por Hibernate Annotations para mapear las clases con las tablas y atributos. Gracias a estas anotaciones, la creación de herramientas para los IDE es mucho más fácil, al poder hacerse las correcciones en tiempo de compilación, tener a nuestro alcance el completado de anotaciones, como el de código java tradicional, haciendo estas tareas rutinarias y tediosas más fáciles de completar y con menos posibilidad de error.



Hibernate es un servicio de persistencia objeto/relaciones y consultas para Java. Hibernate facilita a los desarrolladores crear las clases de persistencia utilizando el lenguaje Java - incluyendo la asociación, herencia, polimorfismo y composición y el entorno de colecciones Java.

Persistencia II

Diseño del diagrama de la Persistencia



Diseño Grafico I

Las decisiones sobre la tipografía, la distribución de la información por pantalla y los colores apropiados no solamente harán que la interface sea más o menos atractiva, sino que determinarán si usabilidad y nos garantizarán su éxito.

Diseño Grafico II

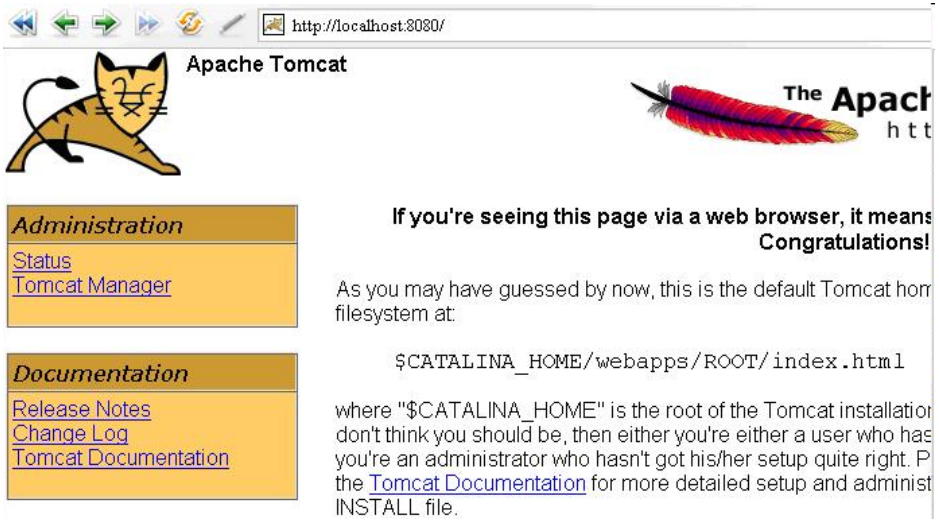
El Diseño gráfico se rige por los 6 principios que a continuación se detallan :

- Agrupación (Organizar el espacio utilizado en pantalla en bloques separados y con funciones similares)
- Visibilidad (Los controles que los usuarios deban usar de una forma más frecuente deben estar siempre visibles y con un acceso lo más rápido posible)
- Consistencia (Utilizar un distribución de la información similar para acciones similares)
- Economía (Se debe omitir cualquier elemento del sistema que no aporte ningún tipo de información al usuario)
- Color como suplemento (Se debe utilizar el color para destacar la información más importante)
- Reducción del desorden (Se puede entender este punto cómo un resumen de los 5 principios comentados anteriormente)

Servidor

El proyecto se empezó con JBoss. JBoss es un servidor de aplicaciones J2EE de código abierto implementado en Java puro. Al estar basado en Java, puede ser utilizado en cualquier sistema operativo para el que esté disponible Java. Dado que JBoss tiene sus propias librerías dio problemas con las librerías del propio proyecto. Por lo que se decidió cambiar

Tomcat es un servidor de aplicaciones open-source. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems. Aunque no tan potente como JBoss, Tomcat cubre con creces las necesidades básicas que se dieron en el proyecto.



Internacionalización

Gracias a la facilidad de integración del estándar i18n de internacionalización del framework que se ha utilizado, se ha creado una interface gráfica en diferentes idiomas, según tenga definido el usuario en el navegador.

En futuras versiones se podría utilizar la banderita tan típica en este tipo de aplicaciones que permiten al usuario cambiar a su voluntad el idioma de las páginas y de la información que estas dan.

Usabilidad I

La usabilidad es la facilidad con que las personas pueden utilizar una herramienta u otro objeto fabricado por los humanos con el fin de alcanza un objetivo concreto.

Podemos observar la sencillez y lo intuitivo que representa trabajar con este portal:

Usabilidad II

Los principales atributos son :

- Facilidad de aprendizaje (minimizar el tiempo desde el desconocimiento hasta su utilización correcta)
- Familiaridad (Conexión existente entre los conocimientos previos y los conocimientos adquiridos)
- Consistencia (Los mecanismos que se utilicen sean siempre usados de la misma manera)
- Flexibilidad (Proporcionar control al usuario para que éste se adapte al sistema)
- Recuperabilidad (Facilidad con la que se permite corregir un error una vez detectado)
- Robustez (Permitir al usuario poder cumplir sus objetivos, con el asesoramiento necesario para ello)
- Mínimo tiempo de respuesta (para expresar los cambios de estado del usuario)
- Adecuación a las tareas (Los servicios ofrecidos deben ser adaptados al modelo mental del usuario)

Ventajas del Proyecto Naoru

- Comodidad
- Sencillez
- Interfaz Intuitiva
- Accesibilidad
- Mínimo esfuerzo físico
- Seguridad
- Usabilidad
- Rapidez

Conclusiones

- Después de todo el trabajo realizado y las horas dedicadas a sacar el proyecto solo puedo decir que me siento orgulloso del producto final obtenido. No estoy satisfecho del todo porque el producto se puede mejorar, pero dado mi conocimiento inicial de todos los frameworks y de J2EE en general, era totalmente desconocido para mi, he aprendido mucho.

- Aunque se han cumplido los objetivos principales, si tuviera que empezar de cero de nuevo, el enfoque y el desarrollo se harían de diferente manera. He visto muchas posibles mejoras de cara el diseño del cuerpo de las paginas para una mejor reutilización del código, al igual que implementaría de manera diferente el tema de roles de los usuarios.

- Por lo que a software se refiere, no haría cambios ya que las plataformas elegidas me han ido muy bien, pero a lo mejor le daría la oportunidad a otros frameworks como Spring o Java Server Faces por ejemplo para el diseño de los JSP's.