Gestor de incidencias

José Aguilera Deixt
ETIG

José Juan Rodríguez

18 de Junio de 2012

Resumen del proyecto

El presente trabajo de fin de carrera se plantea en base al análisis, planificación y desarrollo de una aplicación web basada en el modelo Java EE, que permitirá la gestión de incidencias, enfocado a empresas que dan tanto soporte de su Software como del Hardware.

Java EE es una especificación que permite la creación de aplicaciones multicapa basada en componentes modulares que se ejecutan sobre un servidor de aplicaciones. El uso de este modelo arquitectónico basado en capas nos proporciona independencia y robustez además permite que cada una de las capas se centre en sus objetivos específicos, minimizando las interferencias recibidas por parte del resto de componentes de la aplicación.

Este proyecto se basará en un modelo de tres capas:

- La capa de presentación y control.
- La capa de negocio que controlará la lógica con la que se operan los datos.
- La capa que nos proporcionará el acceso a la información persistente.

Para la capa de presentación y la de negocio utilizaremos Struts2 que es un framework de aplicación Open Source desarrollado por Apache y que está basado en el patrón MVC (Modelo-Vista- Controlador).

Para la capa que gestionará los accesos a los datos de nuestra aplicación utilizaremos Hibernate que es una herramienta de software libre de mapeo objeto-relacional.

1. Índice

2. Índice de figuras

3. Introducción

3.1 Justificación del TFC y contexto

Este proyecto nace de la necesidad de poder gestionar las incidencias de una empresa como en la que estoy trabajando actualmente, en la cual tenemos un ERP especializado en empresas de suministro, sobre todo en el sector eléctrico. Conforme han pasado los años hemos ampliado bastante la cartera de nuestros clientes y ahora es cuando es necesario controlar de forma más optimizada las incidencias y mirar de canalizar todas las incidencias a través de una sola aplicación. En la actualidad recibimos estas incidencias por varios canales, incluso algunos de nuestros clientes envían esas incidencias por diferentes canales, provocando a veces trabajos duplicados y cierto descontrol. Este sistema de gestión de incidencias permitirá registrar tanto las incidencias provenientes de la aplicación de gestión como las originadas por el hardware instalado.

Para el desarrollo de esta herramienta de control se ha optado por la creación de una aplicación web, de esta forma tanto clientes como técnicos podrán tener acceso desde cualquier lugar donde tengan conexión a Internet.

3.2 Objetivos del TFC

El objetivo es la creación de una herramienta que se pueda visualizar por medio de un navegador (Explorer, Firefox…). Para conseguirlo se cree conveniente la utilización de J2EE, ya que es un sistema muy adecuado para este tipo de aplicaciones. Con la ventaja de no incurrir en gastos de licencias.

Por otro lado como reto personal, aprender el funcionamiento y la lógica de esta tecnología ya que nunca he desarrollado una aplicación J2EE y mis nociones de Java son muy limitadas.

Para el desarrollo del proyecto se ha utilizado Eclipse como entorno de desarrollo, Tomcat como servidor WEB y MySql como sistema gestor de bases de datos. Se ha optado por estas herramientas al ser algunas de las más utilizadas además de gratuitas.

Se ha decidido utilizar el modelo de patrón MVC, que separa en diferentes capas el modelo, la vista y el controlador.

3.3 Enfoque y método seguido

La metodología seguida se basa en el sistema en cascada del ciclo de vida, se define de forma estricta el ciclo de desarrollo del software de forma que no empieza una fase hasta haber acabado la anterior.

Las fases son las siguientes:

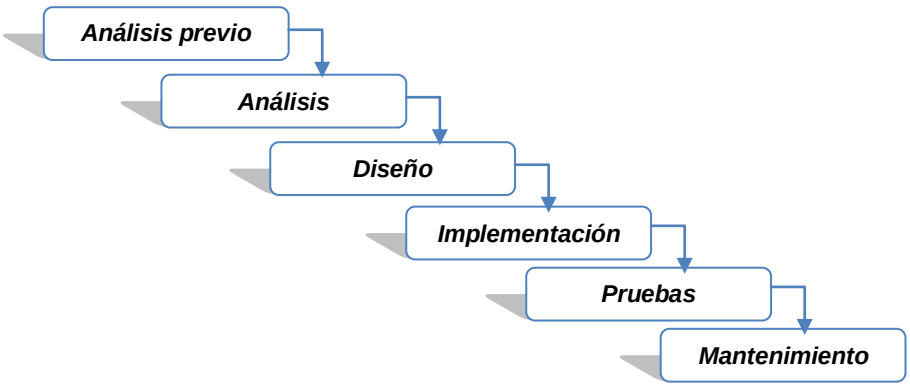


Figura 01 – Ciclo de vida

En este proyecto no se tendrá en cuenta la última fase, la de mantenimiento. Se debe tener en cuenta que cualquier fallo que se detecte en la fase de implementación o de pruebas puede producir que se deba volver a las fases anteriores para rediseñar las partes afectadas, con el inconveniente de que produzca un retraso en la planificación.

3.4 Planificación del proyecto

3.4.1 Introducción

Como Proyecto de Fin de Carrera, dentro del área de J2EE, se pretende desarrollar una aplicación de Gestión de Incidencias que funcione basada en esta arquitectura.

Las líneas generales de la aplicación son las siguientes:

- El usuario cliente o el personal técnico puede crear y consultar las incidencias mediante un portal web.
- El administrador puede asignar las incidencias al técnico que considere.
- Los técnicos registran las intervenciones que realizan sobre una incidencia y pueden cerrar esa incidencia.

También se pretende que el desarrollo sea fácilmente modificable y extensible para poder añadirle nuevas funcionalidades de forma fácil y poco costosa. En este punto nos ayudará la arquitectura J2EE.

3.4.2 Recursos de Software y Hardware

Los recursos de hardware para el desarrollo del proyecto están limitados a los recursos informáticos que posea el alumno, siendo los mínimos requeridos los que la UOC recomienda para la realización de la carrera.

Si bien el proyecto a realizar pretende implementar una arquitectura distribuida (J2EE), eso no impide que su desarrollo y pruebas se puedan realizar en un único ordenador que funcionará de forma simultánea como acceso a la aplicación web, Servidor de

Aplicaciones y Servidor de Bases de Datos. La propia arquitectura J2EE permitirá desplegar posteriormente el producto en un entorno distribuido real en caso de que fuera necesario.

Será suficiente en cuanto hardware, pues, un único ordenador para el desarrollo del proyecto que disponga de conexión a internet para las comunicaciones con el campus y el consultor.

En cuanto a requisitos de software utilizaremos los siguientes productos:

- Microsoft Word para la elaboración de la documentación textual del proyecto y la memoria.
- Microsoft Power Point para la elaboración de la presentación del proyecto.
- Enterprise Architect para la creación de los diagramas de casos de uso, interacción, bases de datos y definición de clases.
- Ganttproject para la creación y el seguimiento de la planificación del proyecto.
- Eclipse como entorno de desarrollo.
- MySQL como servidor de bases de datos.
- Tomcat como servidor de aplicaciones.

3.4.3 Schedule del proyecto

Para elaborar la planificación de este proyecto se debe tener en cuenta la disponibilidad de trabajo y las tareas que se deben realizar, cumpliendo con las fechas en las que se deben entregar los diferentes hitos marcados por las PAC's.

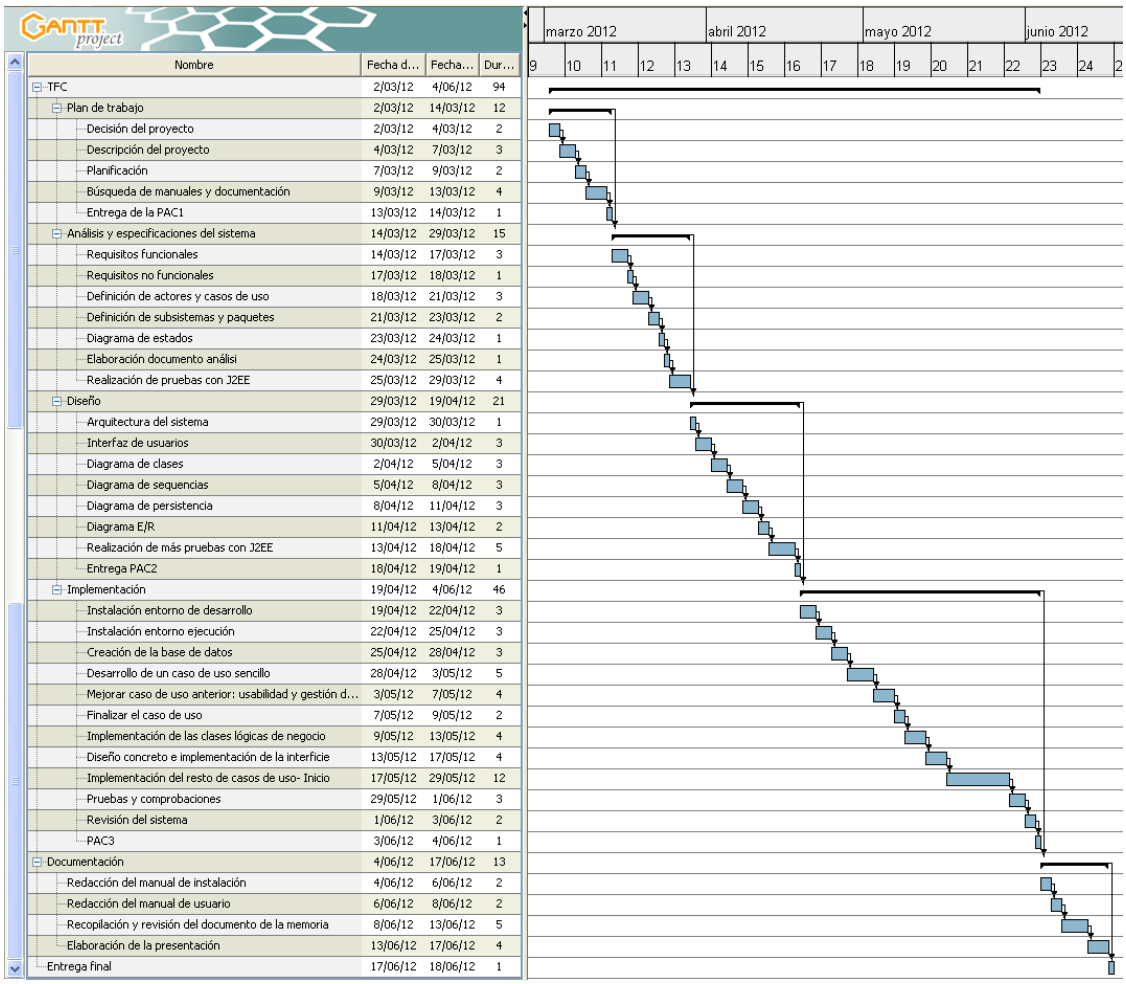
Para no tener problemas en la fase de implementación se requerirá invertir tiempo en el estudio de la tecnología J2EE, documentarse y realizar pruebas a fin de encontrar la mejor forma de desarrollar las diferentes partes de esta aplicación.

Para elaborar la planificación se debe tener en cuenta los diferentes hitos propuestos por la UOC, que son los siguientes:

- **Plan de trabajo**: 14/03/2012 Confección y entrega del plan de trabajo con la descripción del proyecto, objetivos y planificación
- **PAC2**: 19/04/2012 Entrega de los documentos de análisis y diseño del proyecto. El primero incluirá el análisis de requisitos funcionales y no funcionales, diagrama de casos de uso y diagrama de estado, el segundo la definición de la arquitectura del sistema, prototipos de pantalla, diagrama estático de clases, E/R de persistencia y diagramas de colaboración.
- **PAC3**: 04/06/2012 Esta entrega está catalogado como de control de seguimiento.
- **Entrega final**: 18/06/2012 Entrega de la memoria del proyecto, software, manual de instalación y presentación del proyecto.

Se considerará los festivos y fines de semana como días laborables.

Teniendo en cuenta todas las consideraciones se elabora el siguiente diagrama de Gantt:



Nombre	Fecha d...	Fecha...	Dur...
TFC	2/03/12	4/06/12	94
Plan de trabajo	2/03/12	14/03/12	12
Decisión del proyecto	2/03/12	4/03/12	2
Descripción del proyecto	4/03/12	7/03/12	3
Planificación	7/03/12	9/03/12	2
Búsqueda de manuales y documentación	9/03/12	13/03/12	4
Entrega de la PAC1	13/03/12	14/03/12	1
Análisis y especificaciones del sistema	14/03/12	29/03/12	15
Requisitos funcionales	14/03/12	17/03/12	3
Requisitos no funcionales	17/03/12	18/03/12	1
Definición de actores y casos de uso	18/03/12	21/03/12	3
Definición de subsistemas y paquetes	21/03/12	23/03/12	2
Diagrama de estados	23/03/12	24/03/12	1
Elaboración documento análisi	24/03/12	25/03/12	1
Realización de pruebas con J2EE	25/03/12	29/03/12	4
Diseño	29/03/12	19/04/12	21
Arquitectura del sistema	29/03/12	30/03/12	1
Interfaz de usuarios	30/03/12	2/04/12	3
Diagrama de clases	2/04/12	5/04/12	3
Diagrama de sequencias	5/04/12	8/04/12	3
Diagrama de persistencia	8/04/12	11/04/12	3
Diagrama E/R	11/04/12	13/04/12	2
Realización de más pruebas con J2EE	13/04/12	18/04/12	5
Entrega PAC2	18/04/12	19/04/12	1
Implementación	19/04/12	4/06/12	46
Instalación entorno de desarrollo	19/04/12	22/04/12	3
Instalación entorno ejecución	22/04/12	25/04/12	3
Creación de la base de datos	25/04/12	28/04/12	3
Desarrollo de un caso de uso sencillo	28/04/12	3/05/12	5
Mejorar caso de uso anterior: usabilidad y gestión d...	3/05/12	7/05/12	4
Finalizar el caso de uso	7/05/12	9/05/12	2
Implementación de las clases lógicas de negocio	9/05/12	13/05/12	4
Diseño concreto e implementación de la interficie	13/05/12	17/05/12	4
Implementación del resto de casos de uso- Inicio	17/05/12	29/05/12	12
Pruebas y comprobaciones	29/05/12	1/06/12	3
Revisión del sistema	1/06/12	3/06/12	2
PAC3	3/06/12	4/06/12	1
Documentación	4/06/12	17/06/12	13
Redacción del manual de instalación	4/06/12	6/06/12	2
Redacción del manual de usuario	6/06/12	8/06/12	2
Recopilación y revisión del documento de la memoria	8/06/12	13/06/12	5
Elaboración de la presentación	13/06/12	17/06/12	4
Entrega final	17/06/12	18/06/12	1

Figura 02 – Diagrama de Gantt

3.5 Productos obtenidos

Los productos finales obtenidos son los siguientes:

- **Memoria**: Síntesis del trabajo realizado en el TFC, mostrando claramente que se han conseguido los objetivos propuestos. Desde un punto de vista formal la memoria ha de contener aquella información relevante que permita comprender el problema planteado en el TFC, la metodología que se ha empleado para su resolución y muestre la solución del problema planteado.

- **Presentación**: Síntesis clara y concisa del trabajo realizado a lo largo del semestre y de los resultados obtenidos, ofreciendo una perspectiva general del TFC.

- **Aplicación Gestión Incidencias**, compuesto por:

    - Proyecto Eclipse con el código desarrollado y la estructura de desarrollo.
    - GestionIncidencias.war: Componente a desplegar en el contenedor web (vista y controlador) que contiene todas las clases bytecode de java, los recursos web y las librerías necesarias para realizar la instalación en un servidor web Apache Tomcat.
    - Ficheros de sentencias SQL para la creación tanto de la base de datos como para inserción de datos necesarios para realizar pruebas.
    - El manual de instalación (anexo en la memoria).
    - El manual del usuario (anexo en la memoria).

3.6 Descripción del resto de capítulos de la memoria

Una vez definido el **Plan de Proyecto** ya tenemos todas las tareas y pasos a seguir para cumplimentar todo el proceso de desarrollo.

El capítulo 4 contiene la **Documentación de Requisitos** y el capítulo 5 contiene el **Modelo de Análisis** generado a partir de esa documentación.

El siguiente paso en el proceso de desarrollo de software es el **Diseño** de la aplicación que queda documentado en el capítulo 6. Para poder realizar un diseño directamente implementable es necesario tener muy clara la **arquitectura del sistema** sobre la que se va a desarrollar. Es por ello que iniciamos este capítulo describiendo esta arquitectura, que nos ayudará a tomar las decisiones de diseño que vamos a aplicar.

En el capítulo 7 mostramos las C**onclusiones** y futuras mejoras que se podrían implementar.

Para finalizar esta memoria incluimos el **glosario** en el capítulo 8, la **bibliografía** en el capítulo 9 y un apartado de anexos en el capítulo 10. Hay que decir que estos anexos son materiales que forma parte del producto software obtenido ya que son el **Manual de Instalación** y el **Manual del Usuario**.

4. Documentación de requisitos

4.1 Descripción del sistema

La aplicación Gestión de Incidencias nace con la idea de ayudar a un departamento de soporte interno en la gestión de peticiones de soporte que sus clientes (los usuarios del sistema) le remiten ante los problemas informáticos con los que se encuentran en su sistema.

El sistema debe facilitar el registro de dichas incidencias, la asignación de los técnicos que deben resolverlas, registrar las acciones realizadas y proveer al cliente de un sistema de información que le muestre el estado de sus peticiones.

4.2 Descripción de requisitos

Requisitos funcionales

Los usuarios solo podrán ser registrados por los técnicos de la empresa que ofrece el servicio de soporte, para controlar realmente a quien queremos dar acceso, se determina no asignar más de un usuario por empresa o delegación a fin de forzar a nuestros clientes a centralizar este tipo de acciones.

Los clientes, tras su identificación en el sistema, deben poder registrar las incidencias que se les produzca, consultar las que están en curso y poder acceder a su histórico, además de poder consultar las intervenciones que se han realizado.

Para registrarlas el usuario deberá aportar una serie de información, si se trata de una incidencia de tipo Software o Hardware, modulo en el que se produce, si es un error general o solo en un equipo, una descripción breve y una detallada. Estas incidencias podrán ser introducidas por parte de los propios técnicos quedándoles ya asignadas, de esta forma se podrán registrar en el sistema incidencias reportadas por mail, vía telefónica u otros medios.

Las incidencias nuevas introducidas por el cliente quedarán en estado “pendiente” hasta el momento en que el administrador de soporte la asigne a algún técnico, en cuyo momento esta quedará en estado “asignada”.

Los técnicos podrán visualizar las incidencias pendientes que tienen asignadas y registrar las intervenciones, pudiendo cerrar la incidencia. Además podrán consultar todas las incidencias que tiene asignadas en el sistema, mediante una serie de filtros.

4.3 Identificación de los actores

De los requisitos funcionales expuestos anteriormente vemos que tenemos tres tipos de actor claramente diferenciados: cliente, técnico y administrador.

Identificamos los siguientes casos de uso por actor:

- **Cliente:** Identificación en el sistema, Registro de incidencias y Consulta de las incidencias reportadas con su identificar de cliente (pueden existir varios usuarios con el mismo código de cliente).

- **Técnico:** Identificación en el sistema, Registro de incidencias, Consulta de las incidencias que tiene asignadas, Registro de Intervenciones, mantenimiento de clientes y módulos.

- **Administrador:** Identificación en el sistema, Consulta de todas las incidencias, Registro de Intervenciones, mantenimiento de clientes, mantenimiento módulos y Asignar técnico a una incidencia.

4.4 Diagrama de casos de uso

Veamos ahora como se relacionan los actores y los casos de uso que hemos detectado:

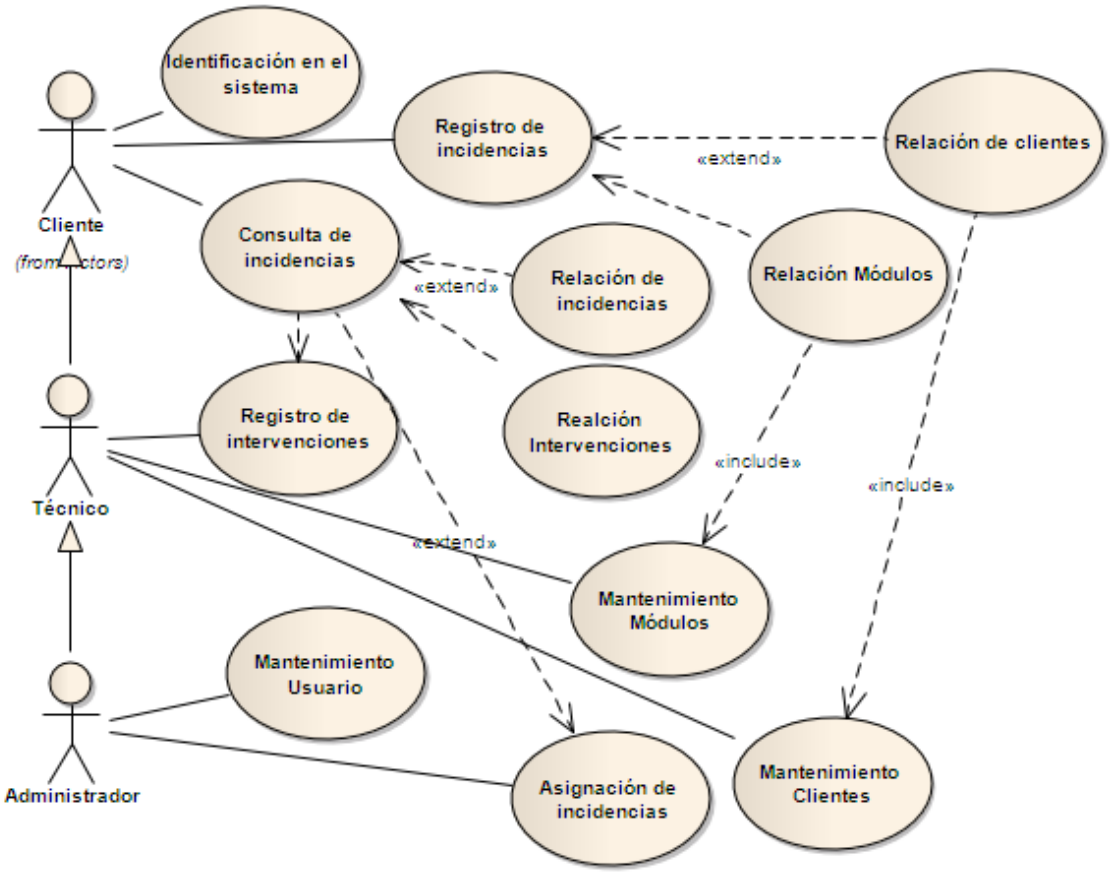


Figura 03 – Diagrama de casos de uso

4.5 Descripción de los casos de uso

4.5.1 Identificación en el sistema

Caso de uso	Identificación en el sistema
Funcionalidad	Permite al usuario identificarse en el sistema
Actores	Usuario base (cliente, técnico y administrador)
Casos relacionados	
Precondición	El usuario no está registrado en el sistema
Postcondición	El usuario ha sido identificado en el sistema y se ha cargado su perfil.
Proceso normal	1- El usuario introduce su identificador de usuario y su contraseña. 2- El sistema valida sus datos y le permite la entrada en el sistema
Alternativas de proceso	El usuario o la contraseña son incorrectos, el sistema notifica el error y vuelve a cargar la pantalla de identificación.

4.5.2 Registro de incidencias

Caso de uso	Registro de incidencias
Funcionalidad	Permite al usuario registrar una incidencia
Actores	Cliente y técnico
Casos relacionados	Consulta de incidencias
Precondición	El usuario está identificado en el sistema
Postcondición	La incidencia queda registrada en estado “pendiente”
Proceso normal	El usuario introduce los datos: Soft/Hard, modulo en el que se produce la incidencia, si es local o global, descripción breve y detallada.
Alternativas de proceso	Si el usuario es un técnico tendrá la opción de seleccionar el cliente en el que se produce la incidencia.

4.5.3 Consulta de incidencias

Caso de uso	Consulta de incidencias
Funcionalidad	Permite consultar una incidencia mostrando todos los datos relacionados con ella
Actores	Usuario base (cliente, técnico y administrador)
Casos relacionados	Relación de incidencias
Precondición	El usuario está identificado en el sistema
Postcondición	
Proceso normal	1- El sistema muestra la relación de incidencias que tiene registradas con su Id de cliente. Sí el usuario es un técnico se presentarán solo las que tiene asignadas y si es el

	administrador podrá consultar todas las incidencias. 2- El usuario podrá consultar las intervenciones que se han registrado sobre la incidencia.
Alternativas de proceso	1- El usuario puede modificar los filtros de consulta y volver a ejecutar la consulta.

4.5.4 Asignación de incidencia

Caso de uso	Asignación de incidencia
Funcionalidad	Permite asignar una incidencia a un técnico
Actores	Administrador
Casos relacionados	Consulta de incidencias
Precondición	El usuario se ha identificado en el sistema como administrador
Postcondición	El técnico queda registrado en la base de datos y la incidencia queda en estado "Asignada"
Proceso normal	1- El usuario selecciona la incidencia sobre la que va a asignar un técnico. 2- El sistema comprueba su existencia y muestra sus datos. 3- El usuario asigna a uno de los técnicos registrados en el sistema. 4- El sistema almacena el cambio.
Alternativas de proceso	2a- La incidencia está como "cerrada", el sistema permite re-abrir la incidencia y cambiar el técnico.

4.5.5 Registro de intervención

Caso de uso	Registro de intervenciones
Funcionalidad	Permite registrar las intervenciones que se realizan para cerrar una incidencia.
Actores	Técnico y administrador
Casos relacionados	Consulta de incidencias
Precondición	El usuario está identificado en el sistema como técnico o administrador.
Postcondición	La intervención queda registrada en la base de datos y si se marca como cerrada actualiza el estado de la incidencia y su fecha de cierre.
Proceso normal	1- El usuario selecciona la incidencia sobre la que va a registrar una intervención. 2- El usuario introduce la descripción de la intervención, el tiempo dedicado y puede marcar el cierre. 3- El sistema almacena la intervención y registra el cambio de estado si este se produce.
Alternativas de proceso	

4.5.6 Relación de incidencias

Caso de uso	Relación de incidencias
Funcionalidad	Permite mostrar una lista de incidencias para su posterior selección.
Actores	Usuario base (cliente, técnico y administrador)
Casos relacionados	Consulta de incidencias
Precondición	El usuario está registrado en el sistema
Postcondición	
Proceso normal	1- El sistema recibe como parámetros los filtros de selección. 2- El sistema muestra la relación de incidencias que cumplen los filtros de selección.
Alternativas de proceso	

4.5.7 Relación de clientes

Caso de uso	Relación de clientes
Funcionalidad	Permite mostrar una lista de clientes registrados en el sistema para su posterior selección.
Actores	Usuario técnico y administrador
Casos relacionados	Mantenimiento de usuarios y alta de incidencias.
Precondición	El usuario está registrado en el sistema
Postcondición	
Proceso normal	El sistema muestra la relación de todos los clientes.
Alternativas de proceso	

4.5.8 Relación módulos

Caso de uso	Relación de módulos
Funcionalidad	Permite mostrar una lista de módulos para su posterior selección.
Actores	cliente y técnico
Casos relacionados	Alta de incidencias
Precondición	El usuario está registrado en el sistema
Postcondición	
Proceso normal	1- El sistema muestra la relación de módulos.
Alternativas de proceso	

4.5.9 Relación usuarios técnicos

Caso de uso	Relación de usuarios técnicos
Funcionalidad	Permite mostrar una lista de usuarios técnicos para su posterior selección.

Actores	administrador
Casos relacionados	Asignar técnico a la incidencia
Precondición	El usuario está registrado como administrador en el sistema
Postcondición	
Proceso normal	1- El sistema muestra la relación de técnicos.
Alternativas de proceso	

4.5.10 Mantenimiento clientes

Caso de uso	Mantenimiento cliente
Funcionalidad	Permite gestionar los clientes.
Actores	Técnico y administrador
Casos relacionados	
Precondición	El usuario está registrado en el sistema
Postcondición	
Proceso normal	1- El sistema permite mantener la BBDD de clientes.
Alternativas de proceso	

4.5.11 Mantenimiento módulos

Caso de uso	Mantenimiento módulos
Funcionalidad	Permite gestionar los módulos.
Actores	Técnico y administrador
Casos relacionados	
Precondición	El usuario está registrado en el sistema
Postcondición	
Proceso normal	1- El sistema permite mantener la BBDD de módulos.
Alternativas de proceso	

4.5.12 Mantenimiento usuarios

Caso de uso	Mantenimiento de Usuarios
Funcionalidad	Permite gestionar los usuarios.
Actores	Administrador
Casos relacionados	
Precondición	El usuario está registrado en el sistema como administrador
Postcondición	
Proceso normal	1- El sistema permite mantener la BBDD de usuarios.
Alternativas de proceso	

5. Análisis de requisitos

5.1 Revisión de los casos de uso

Antes de iniciar el análisis hay que detenerse a revisar los casos de uso de los requisitos presentados al cliente, para ver si encontramos algún fallo, indeterminación o contradicción que deberíamos solucionar antes de proseguir.

En nuestro caso, los casos de uso ya fueron revisados antes de cerrar la Documentación de Requisitos, siendo está la base para nuestro análisis.

5.2 Identificación de las clases de entidad

A partir de los Casos de Uso y del Modelo de Dominio vemos que clases entidad candidatas tenemos así como sus atributos:

- Modelo de Dominio: Cliente, Usuario, Técnico, Administrador, Módulo, Incidencia e Intervención.

- Caso de Uso Identificación en el sistema: Usuario (identificador, password).

- Caso de Uso Registro de Incidencias: Incidencia (Software/Hardware, local / global, módulo, descripción breve y larga, estado, fecha, cliente (identificador), usuario petición y técnico asignado).

- Caso de Uso Consulta de Incidencias: Incidencia (rango de fechas, tipo, local o global).

- Caso de Uso Asignación Incidencia: Incidencia(código y técnico)

- Caso de Uso Registro de Intervenciones: Intervención (descripción, tiempo dedicado y fecha), Incidencia (estado).

- Caso de Uso Relación de Incidencias: Incidencia.

- Caso de Uso Relación de Clientes: Cliente

- Caso de Uso Relación de Módulos: Modulo

- Caso de Uso Relación de Usuarios técnicos: Usuario

- Caso de Uso Mantenimiento Clientes: Cliente(id, nombre)

- Caso de Uso Mantenimiento Módulos: Modulo(id, nombre)

- Caso de Uso Mantenimiento Usuarios: Usuario (id, login, password, nombre, email, telefono, tipoUsuario y idCliente)

Podemos ver que los tres tipos de actores (Usuario, Técnico y Administrador) son los diferentes usuarios del sistema que requerirán un identificador y un password para acceder al sistema. Será conveniente, por tanto, definir una superclase Usuario.

5.3 Diagramas de interacción

Para ver la dinámica de la aplicación vamos a utilizar diagramas de colaboración simplificados para los Casos de Uso más relevantes.

Caso de Uso 1: Identificación en el sistema

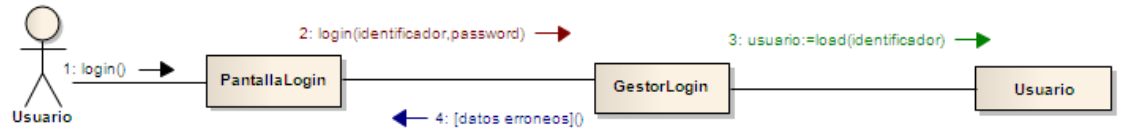


Figura 04 - Caso de Uso 1: Identificación en el sistema

Caso de Uso 2: Registro de incidencias

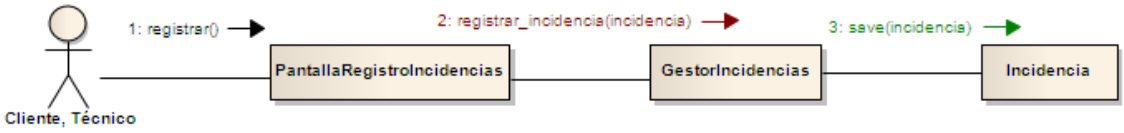


Figura 05 – Caso de Uso 2: Registro de incidencias

Caso de Uso 3: Consulta de incidencias

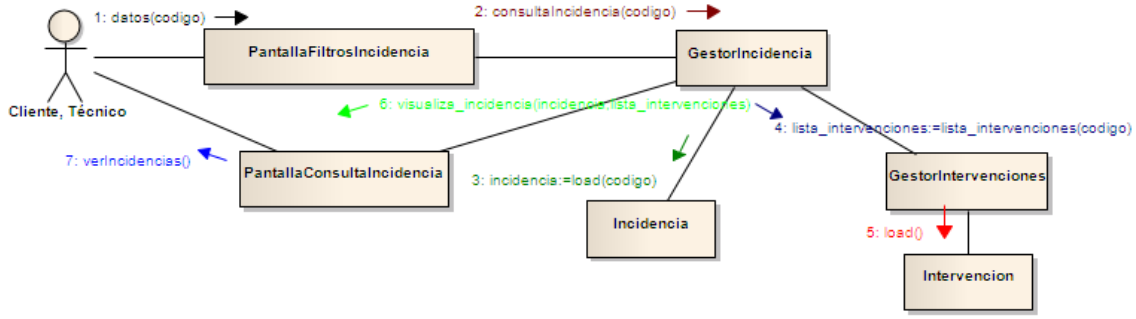


Figura 06 – Caso de Uso 3: Consulta de incidencias

Caso de Uso 4: Asignación de incidencias

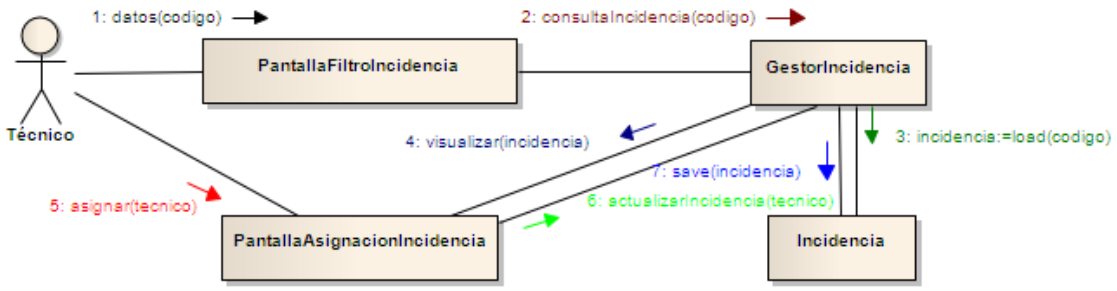


Figura 07 – Caso de Uso 4: Asignación de incidencias

Caso de Uso 5: Registro de intervenciones

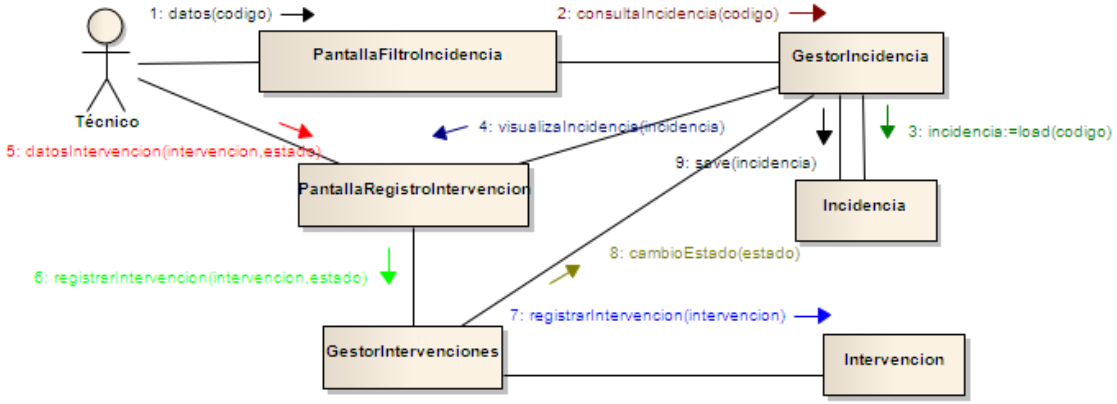


Figura 08 – Caso de Uso 5: Registro de intervenciones

Caso de Uso 6: Relación de incidencias

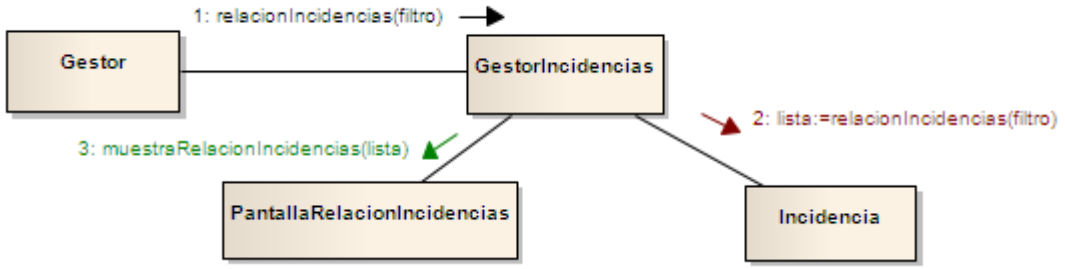


Figura 09 – Caso de Uso 6: Relación de incidencias

5.4 Identificación de las clases de análisis

A partir de los diagramas de colaboración descritos en el apartado anterior, aparecen las clases de análisis detectadas hasta el momento. Las clases de entidad ya las habíamos detectado anteriormente, pero en este momento nos aparecen las clases Control y Frontera. Además aparecen los métodos que se precisan para la interacción entre las clases. Organizando esta información en un diagrama de clases obtenemos este **Diagrama Estático de Análisis.**

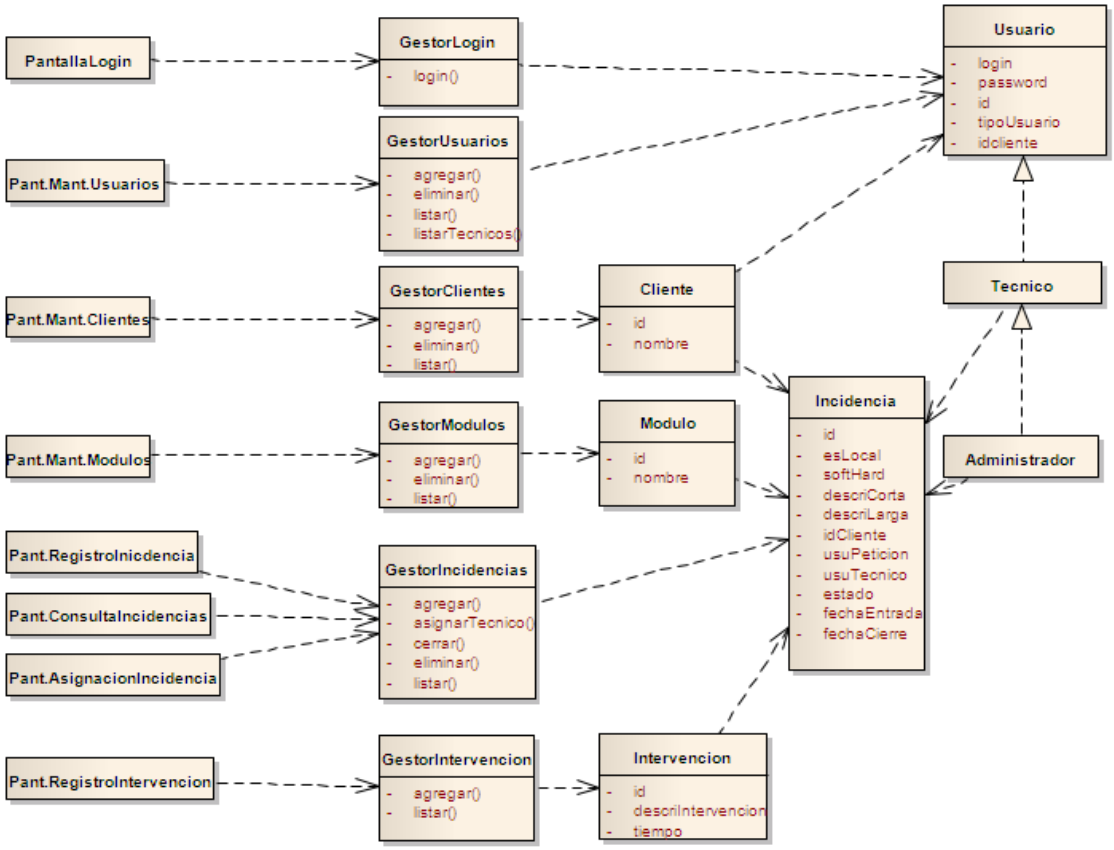


Figura 10 – Diagrama Estático de Análisis

6. Diseño

6.1 Diseño arquitectónico

6.1.1 Java EE

Java EE es una plataforma creada por Sun y cuyo objetivo es el desarrollo de aplicaciones distribuidas dirigidas principalmente a la empresa. Entre sus requisitos están la fiabilidad, la facilidad de mantenimiento, la escalabilidad, etc.

Esta plataforma está basado en Java que es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystem a principios de los años 90 y cuyos principales objetivos eran:

- Uso de la metodología orientada a objetos.
- Portabilidad del mismo programa en múltiples sistemas operativos.
- Incluir por defecto soporte para trabajo en red.
- Permitir la ejecución de código en sistemas remotos de forma segura.
- Facilidad de uso tomando elementos de otros lenguajes orientados a objetos.

La portabilidad entre plataformas que proporciona este lenguaje viene dada por el hecho de que al compilar el código fuente escrito en Java se genera un código conocido como "bytecode" que es ejecutado por lo que se conoce como la máquina virtual de Java. Esta máquina virtual es un programa escrito en código nativo de la plataforma y es el encargado de interpretar y ejecutar el código, es ella la que conoce el hardware sobre el que se ejecuta la aplicación y la que actúa como intermediaria entre ambas.

Esta portabilidad de Java hace que una aplicación Java EE también se pueda instalar fácilmente en cualquier entorno que soporte dicho lenguaje.

La especificación Java EE define un modelo de capas mediante el cual la lógica de la aplicación se divide en componentes de acuerdo con su función. Cada uno de estos componentes se puede acabar instalando en una máquina diferente (aunque no necesariamente) dependiendo de la capa a la que pertenezca.

Las capas en las que se divide el sistema son las siguientes:

- **Capa cliente**: Se ejecuta en la máquina cliente. Es la que permite al usuario interactuar con el sistema y pueden ser clientes ligeros, como un navegador web o bien clientes pesados, como una aplicación de escritorio.
- **Capa web**: Se ejecuta en un servidor Java EE. Es la encargada de obtener datos del cliente y de solicitar a la capa de negocio las operaciones necesarias.
- **Capa de negocio**: Se ejecuta en un servidor Java EE y forma el núcleo de la aplicación. En esta capa estarán representadas nuestras entidades, relaciones y reglas que implementarán nuestros procesos de negocio.
- **Capa EIS (Enterprise Information System):** Se ejecuta en un servidor de base de datos y es la que contiene los datos del negocio por lo que es la encargada de los accesos a la base de datos y de gestión de transacciones del sistema.

Gráficamente el funcionamiento de una aplicación Java EE sería el siguiente:

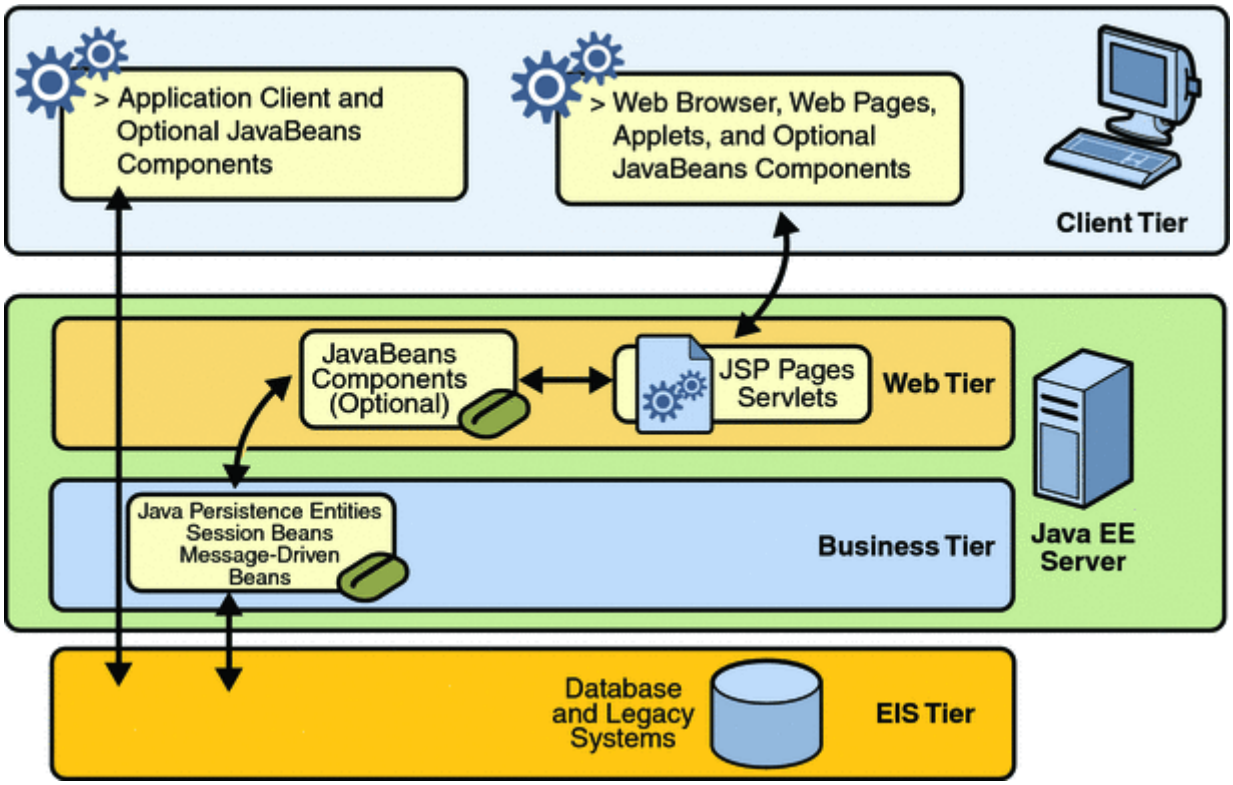


Figura 11 – Arquitectura Java EE

6.1.2 Patrón arquitectónico: MVC

Los **patrones de diseño (design patterns)** proporcionan soluciones ya probadas a problemas con características similares. Proporcionan catálogos de elementos reusables en el diseño de sistemas y estandarizan la forma en que se realiza el diseño, lo que facilita el aprendizaje.

La utilización de patrones de diseño no es una práctica obligatoria en el diseño de software, pero sí que es altamente recomendable ya que permiten el reducir el tiempo de desarrollo de una aplicación al evitar el tener que plantear una solución desde cero.

El patrón **MVC (Modelo-Vista-Controlador)** es el más ampliamente establecido desde el punto de vista arquitectónico y se caracteriza porque divide el sistema en tres partes, de forma que separa los datos de la aplicación de la interfaz de usuario y de la lógica de control.

Con esta división conseguimos, por ejemplo, que la forma de presentar unos datos sea completamente independiente de los datos en sí, con lo que es sencillo el presentar los mismos datos de formas diferentes.

Las responsabilidades de cada una de las partes de este patrón son las siguientes:

- **Modelo**: Contiene los datos con los que opera el sistema. La lógica de datos permite asegurar su integridad y facilita el derivar nuevos datos.

- **Vista**: Representa el modelo de datos y las operaciones realizadas en la capa de negocio en un formato adecuado para que el usuario pueda interactuar (mediante la interfaz de usuario).

- **Controlador**: Responde a los eventos, normalmente acciones del usuario, modificando el modelo y generalmente también en la vista, por lo que es encargado de la interacción entre los datos y la vista.

El flujo seguido por este patrón entre los diferentes componentes es el siguiente:

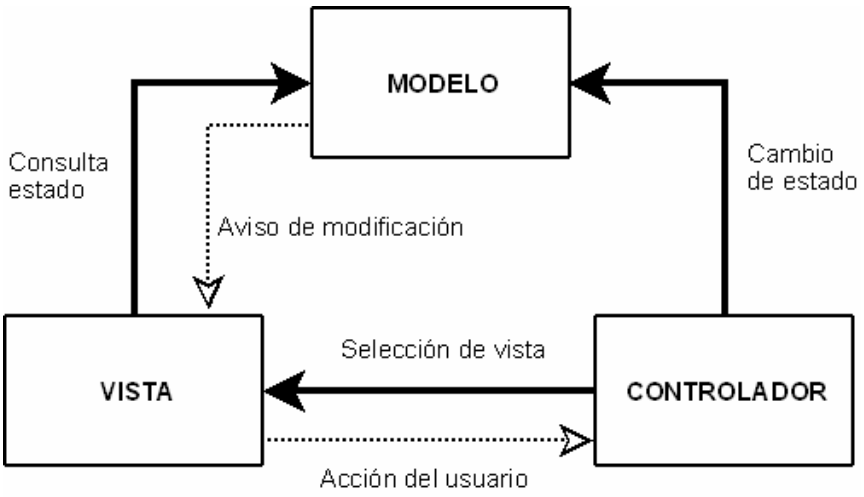


Figura 12 – Patrón MVC

1. El usuario interactúa con la vista (interfaz de usuario) realizando algún tipo de acción.
2. El controlador recibe desde la vista la notificación de la acción solicitada.
3. El controlador accede al modelo actualizándolo conforme a la acción solicitada por el usuario.
4. El controlador delega en la vista la tarea de desplegar la interfaz de usuario con la respuesta.
5. La vista recupera los datos del modelo para poder completar la interfaz con las modificaciones realizadas sobre el modelo.
6. El modelo no debe tener conocimiento directo de la vista, sin embargo en algunos casos puede notificar a la vista de que se han producido cambios.

6.1.3 Frameworks

Un **framework** es una estructura definida a partir de la cual podemos desarrollar un proyecto de software y su intención es establecer una infraestructura que se encargue de realizar las tareas de más bajo nivel necesarias en cualquier proyecto y permitir así a los desarrolladores el poder dedicar más esfuerzo a las tareas de más alto nivel propias del negocio.

6.1.3.1 Struts2

Struts2 es un framework de aplicación web open source desarrollado por Apache y basado en el patrón MVC (Modelo-Vista-Controlador) que es utilizado ampliamente y considerado de gran solidez.

Se utiliza para construir aplicaciones web basadas en servlets JSP, pudiendo ejecutarse en cualquier contenedor de servlets, incluyendo los servidores de aplicaciones Java EE.

Utiliza internamente una serie de patrones ya definidos (Singleton, Delegate,…) y además proporciona un conjunto de etiquetas JSP personalizadas que facilitan la integración del framework con las páginas JSP.

El funcionamiento de Struts2 es el siguiente, cuando el usuario hace una solicitud al servidor el FilterDispatcher la captura y determina el Action que la debe tratar.

Antes de ejecutar el Action se le aplican a la solicitud los Interceptors que hayan sido configurados y que permiten realizar como preproceso una serie de tareas más o menos comunes, como pueden ser validaciones de datos.

A continuación se ejecuta la acción que realiza los accesos para recuperar o almacenar información en la base de datos y se genera la salida mediante un Result.

Esta salida vuelve a pasar a través de los Interceptors (en orden inverso al inicial) para realizar posibles operaciones de postproceso y finalmente se devuelve la respuesta al usuario.

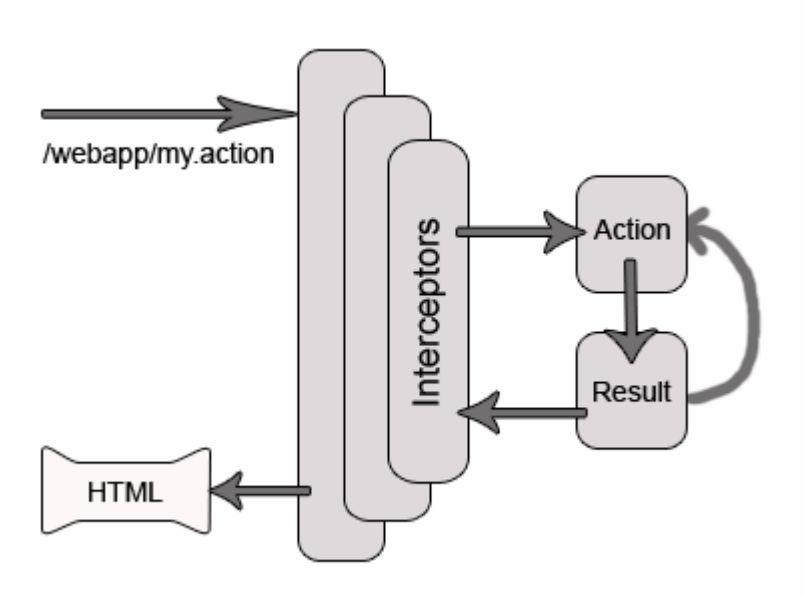


Figura 13 – Esquema Struts2

Desde el punto de vista del patrón MVC los componentes de Struts2 se distribuirían de la siguiente manera, de cara a conseguir la separación de los diferentes componentes de la aplicación:

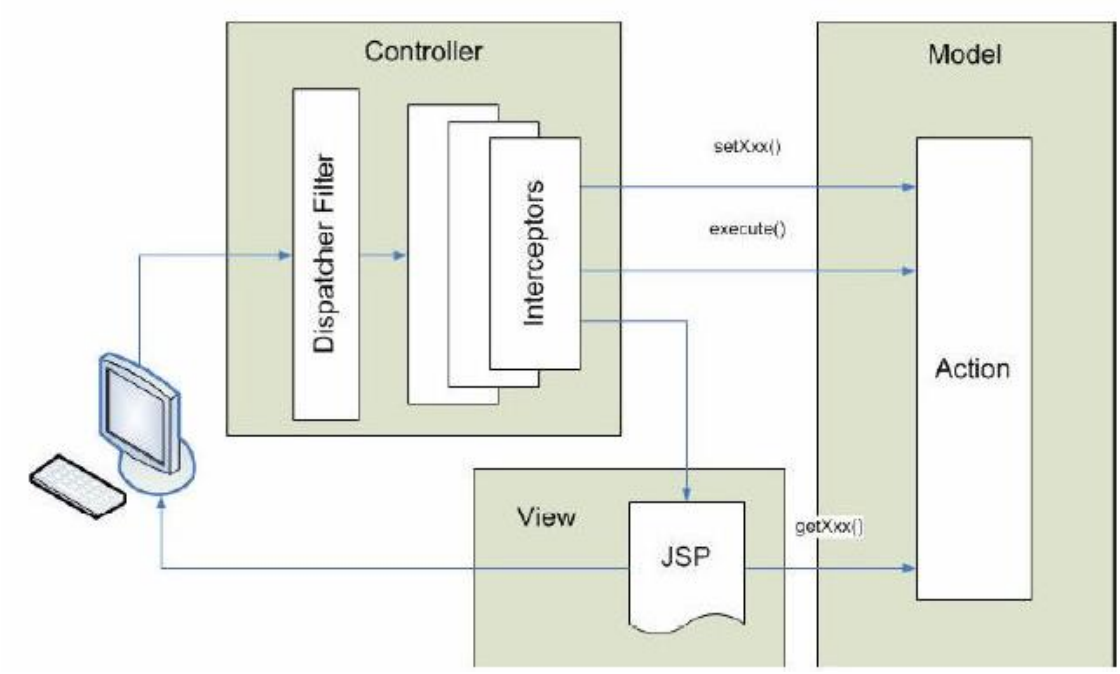


Figura 14 – Esquema Struts2-MVC

6.1.3.2 Hibernate

Hibernate es una tecnología que simplifica el acceso a base de datos, que se distribuye como una herramienta de software libre distribuida bajo los términos de la licencia GNU LGPL.

Permite establecer una correspondencia entre el modelo de la Base de Datos Relacional y una serie de clases que modelan los objetos de la aplicación (mapeo objeto-relacional), es decir, relaciona los dos modelos de datos que conviven en una aplicación, el usado en la memoria del ordenador (orientación a objetos) y el usado en las bases de datos (modelo relacional). Este mapeo hace que en la práctica se cree una base de datos orientada a objetos virtual sobre la base de datos relacional, en la que Hibernate actúa de intermediario con la aplicación que la utiliza.

Figura 15 – Esquema Hibernate

Con esta tecnología disponemos de un sistema de acceso a bases de datos relacionales de manera transparente, únicamente se deben escribir sentencias java y de manera independiente, el código escrito con Hibernate funcionará en cualquier motor de datos al que se dé soporte, ya que Hibernate permite a la aplicación el manipular datos de la base de datos operando sobre objetos.

6.1.3.3 Javaserver Faces

Para la capa de presentación Web utilizaremos un estándar definido por Sun, este es el JavaServer Faces (JSF). Este tiene una gran variedad de componentes básicos de las páginas Webs, además permite la creación de componentes más complejos mediante un conjunto de APIs.

6.1.3.4 Entorno ejecución

A continuación se detalla el software base que define el entorno de ejecución sobre el cual se ejecutará la aplicación de gestión de incidencias.
Utilizaremos Tomcat, es un servidor web que implementa las especificaciones de servlets y JSPs que necesitamos para ofrecer el acceso a la aplicación a través de

Internet. 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.
Por otro lado, en cuanto a la persistencia de los datos escogemos MySQL como sistema gestor de base de datos. Este dispone de herramientas para su administración y es un software de libre distribución que se puede instalar tanto en Windows como UNIX.

6.1.3.5 Entorno desarrollo

Utilizaremos como herramienta de desarrollo el Eclipse, se ha escogido esta herramienta por tratarse de las más reconocidas para el desarrollo de aplicaciones Java, además de poder ir ampliando sus funcionalidades mediante plugins de terceros.

6.2 Diseño de la persistencia

6.2.1 Diagrama E-R

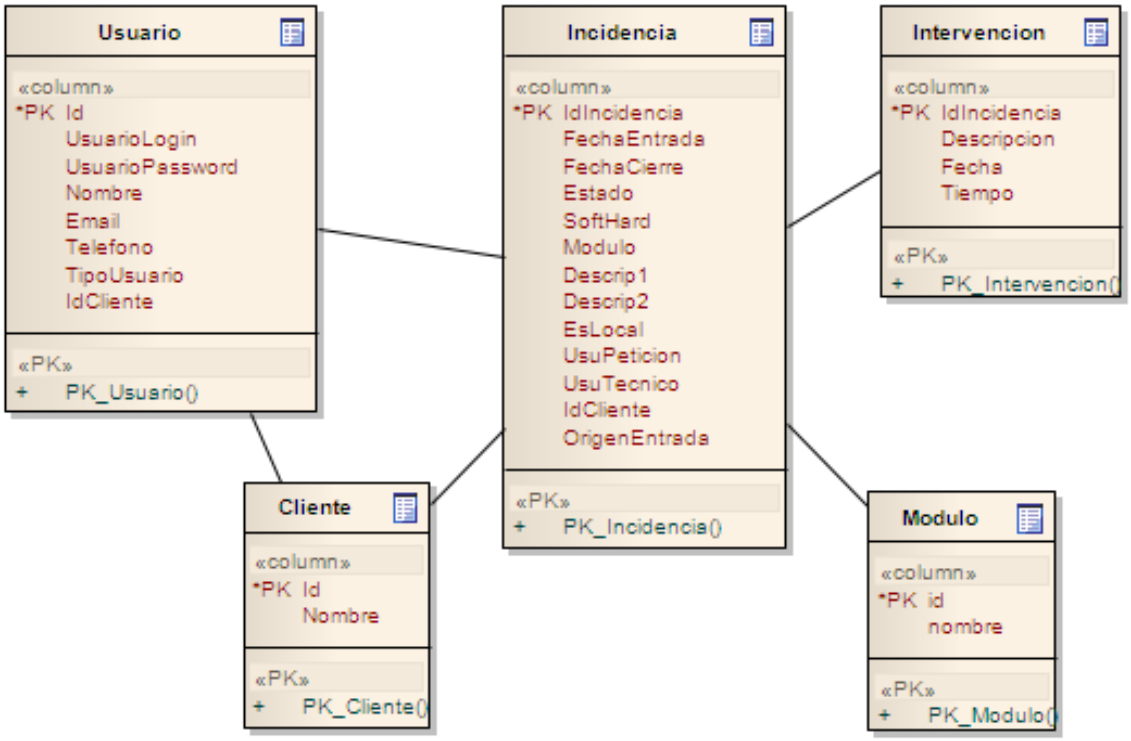


Figura 16 – Diagrama E-R

Se ha deshecho la relación de herencia de Cliente y Administrador con Usuario creando una única entidad con un atributo (tipoUsuario) que servirá para distinguirlos.

6.3 Diagrama de clases

Diagrama de las clases principales de la aplicación agrupadas por packages:

uoc.tfc.model: todas las clases que conforman la lógica de negocio, el modelo de la aplicación, estas son las que se almacenan a la base de datos.

uoc.tfc.view: todas las clases Action que son necesarias para dar respuesta al cliente vía ActionServlet. Estas clases son llamadas desde el fichero struts.xml que hace de controlador dentro del patrón MVC que tiene Struts2 implementado. Las clases Action son las encargadas de de instanciar a les controladores y a los DAO si hace falta.

uoc.tfc.controler: son las clases controladoras que implementan el acceso a la base de datos, estas clases son las encargadas de interactuar con la base de datos y por consiguiente se necesita una para cada objeto a persistir.

uoc.tfc.util: contiene las clases necesarias para utilizar Hibernate y los patrones mencionados de forma correcta.

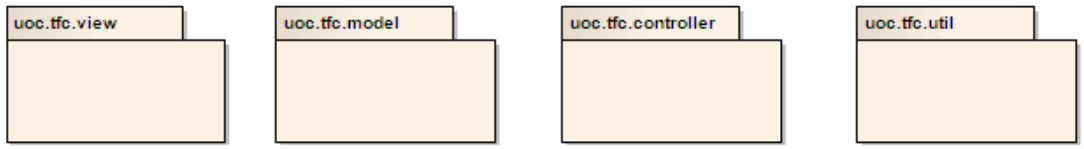


Figura 17 –Diagrama de clases package uoc.tfc

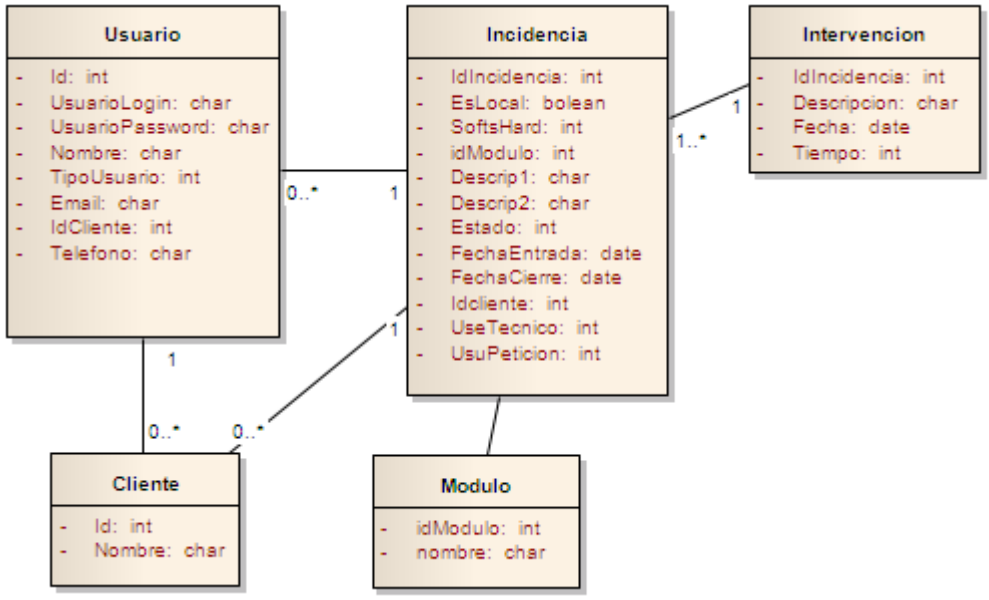


Figura 18 – Diagrama de clases package uoc.tfc.model

6.4 Diseño de la interficie de usuario

1- Identificación en el sistema

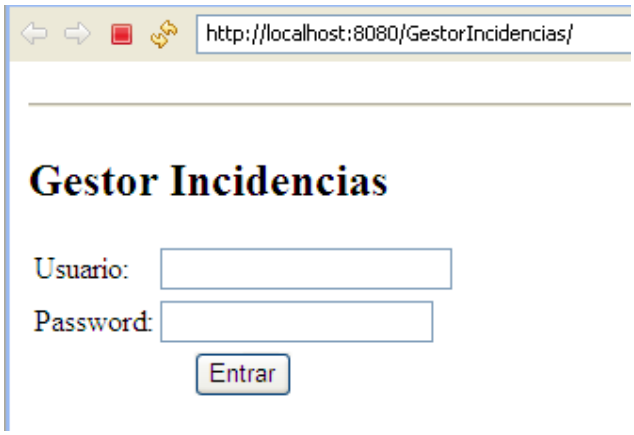


Figura 19 – Pantalla Identificación en el sistema

2- Registro de incidencia

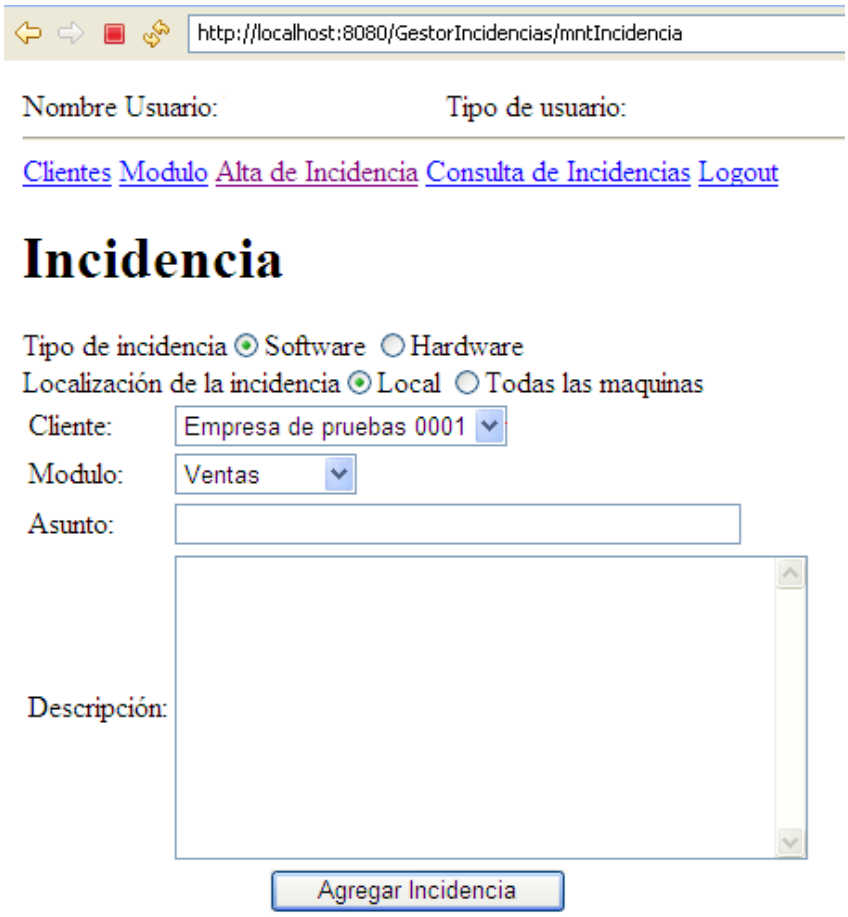


Figura 20 – Pantalla Registro de incidencia

3- Consulta de incidencias

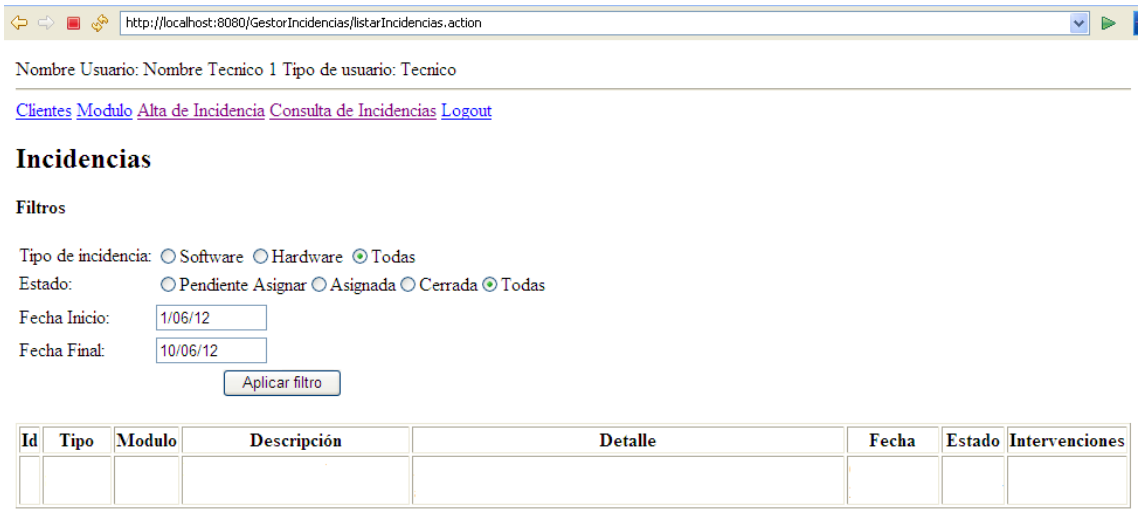


Figura 21 – Pantalla Consulta de incidencias

4- Asignación de incidencia

Figura 22 – Pantalla Asignación de incidencia

5- Registro de Intervención

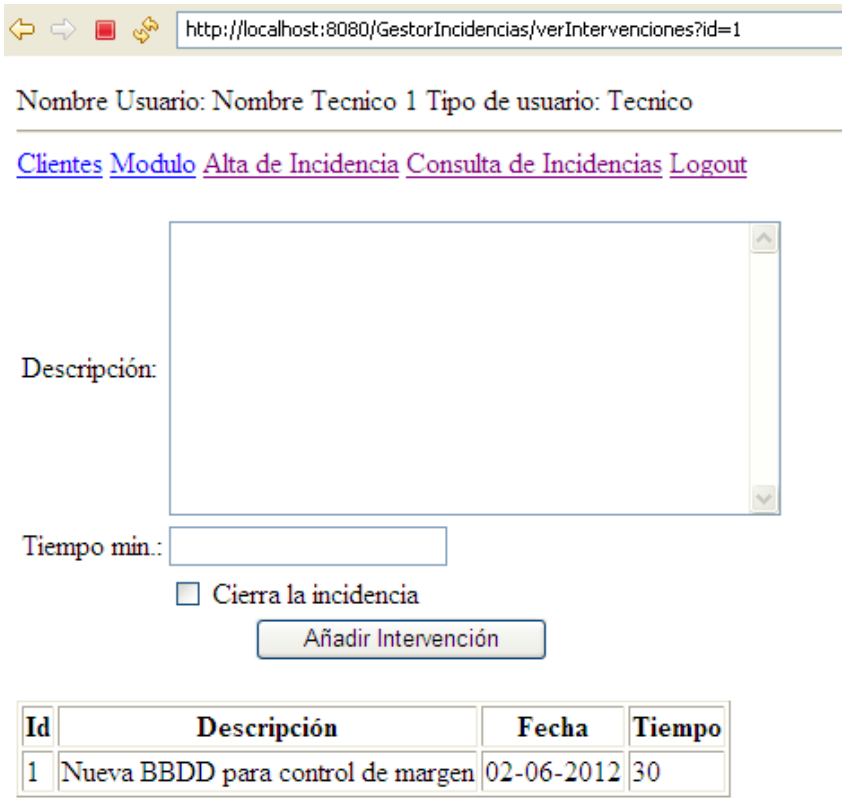


Figura 23 – Pantalla Registro de Intervención

7. Conclusiones

La conclusión tras la realización de este TFC es que en general la tecnología relacionada es difícil de asimilar para un recién llegado al mundo de Java EE, pero que una vez comprendida mínimamente es relativamente sencillo el poder avanzar.

La gran cantidad de alternativas disponibles a la hora de decidir las herramientas a utilizar es una ventaja para un experto ya que le proporciona una flexibilidad enorme que no podría tener de otra forma, pero para un principiante puede ser un gran inconveniente.

Esta gran variedad en las herramientas disponibles sumado a la gran cantidad de documentación y tutoriales que se encuentra sobre ellas en la red provoca que un usuario novato se encuentre perdido y no sepa que tecnologías son las más idóneas y como hacer uso de ellas.

No obstante se ha conseguido cumplir con los requerimientos planteados inicialmente en el proyecto, aunque sería fácil encontrar aspectos en los que la aplicación podría ser mejorada, como por ejemplo:

- Mejorar el control de errores, por ejemplo en los accesos a BBDD, las excepciones.
- Mejorar la validación de datos en pantalla.
- Toda la aplicación está desarrollada en castellano, se podría incluir el idioma en el perfil de los usuarios de forma que éstos pudieran elegir el idioma en que se les mostrara la web.

Para futuras versiones o ampliaciones considero que sería interesante:

- Poder generar estadísticas sobre el volumen de incidencias resuelto por técnico, el tiempo promedio en darles solución,… a fin de detectar la calidad del servicio de la empresa que da el soporte.
- Generar avisos vía mail al usuario cliente cuando se cierre su incidencia.
- Permitir adjuntar ficheros a la incidencia (para que puedan adjuntar una captura de pantalla o cualquier otro tipo de fichero).

Estoy francamente satisfecho con el trabajo realizado, con la tecnología aprendida y aplicada y con la base conceptual que se desprende del paradigma J2EE.

8. Glosario

Incidencia	Problema informático que tiene un usuario con el sistema que precisa de la ayuda o intervención de un especialista para su resolución.
Cliente	Persona que utiliza el sistema informático y que puede tener incidencias en su uso.
Técnico de Soporte	Especialista informático encargado de solucionar los problemas que los usuarios tienen con el sistema.
Intervención	Acción realizada para solucionar parte o toda la problemática de una incidencia de un usuario.
J2EE	Java 2 Enterprise Edition – Especificaciones para el desarrollo de aplicaciones empresariales
ORM	Object-Relational Mapping – Puente entre el mundo de los objetos y las bases de datos relacionales
MVC	Modelo-Vista-Controlador – Patrón de diseño que consiste en desacoplar los datos, la lógica de la aplicación y el interface del usuario.
Hibernate	ORM open source de reconocido prestigio en la comunidad de desarrolladores
Struts	Framework que implementa el patrón MVC para J2EE

9. Bibliografía y enlaces

Brown, Donald; Michael, Chad; Stanlick, Scott (2008). ***Struts 2 in Action***. Madrid: Anaya multimedia.
Valdés-Miranda, Claudia; Plasencia, Zoe (2010). ***Creación y diseño Web***. Madrid: Anaya multimedia.

Fuentes web consultadas:

The J2EE™ 1.4 Tutorial; Sun Microsystems; 5/12/2005
http://docs.oracle.com/javaee/1.4/tutorial/doc/

http://www.roseindia.net/

Struts2
http://mundogeek.net/archivos/2009/02/13/etiquetas-struts-2/
http://viralpatel.net/blogs/tutorial-struts2-hibernate-example-eclipse/
http://java.dzone.com/articles/struts2-tutorial-part-27
http://struts.apache.org/

JavaServerPages
http://www.coderanch.com/t/439803/HTML-CSS-JavaScript/JSP-href-tags-breaks-page
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=coregest2
http://www.programacion.com/articulo/integracion_de_jsf-_spring_e_hibernate_para_crear_una_aplicacion_web_del_mundo_real_307/3

Hibernate
http://www.davidmarco.es/tutoriales/hibernate-reference/
http://www.davidmarco.es/blog/entrada.php?id=243
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernateTools
http://hibernate.org/

Eclipse
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=eclipsetutorial

Apache Tomcat
http://tomcat.apache.org/

10. Anexos

10.1 Manual de instalación

Para la instalación la aplicación es necesario lo siguiente:

1. Disponer de un servidor Apache Tomcat 7 instalado.

   - Enlace http://tomcat.apache.org/download-70.cgi

2. Disponer de un servidor MySql instalado. La versión 5.1 funciona perfectamente.

   - Enlace http://dev.mysql.com/get/Downloads/MySQL-5.1/mysql-5.1.61-win32.msi/from/http://www.mirrorservice.org/sites/ftp.mysql.com/

3. Ejecutar el script de creación de la base de datos de gestorincidencias en el servidor MySql. Este script hace lo siguiente:

   - Crea la Base de Datos y las tablas. *(GIcreaDB.sql)*

   - Inserta registros en las tablas de Usuario, Cliente, Modulo, Incidencia e Intervencion. Los usuarios de prueba son (usuario, usuario2, tecnico y admin todos con el mismo password que el usuario). *(Inserts.sql)*

   - Crea el usuario desde MySQL para poder acceder a la base de datos (usuario: USUARI; password: CLAU)

4. Desplegar el componente war en el servidor Tomcat.

   - Para poder acceder a Manager App se debe crear un usuario modificando el fichero dentro del directorio donde tenemos instalado el apache conf/tomcat-users.xml las líneas a añadir son:
     <role rolename="manager-gui"/>
     <user username="tomcat" password="s3cret" roles="manager-gui"/>

   - Para arrancar el servidor se debe ejecutar en la carpeta donde se tenga instalado el Tomcat *\bin\startup.bat* y para detenerlo *\bin\shutdown.bat (en Unix los de extensión .sh)*.

5. Ejecutar en el cliente la url: http://localhost:8080/GestorIncidencias/

10.2 Manual de usuario

El objeto de la aplicación Gestión de Incidencias es permitir el registro de incidencias por parte del usuario a los que se les presentan, la asignación de las mismas a los técnicos encargados de solventarlas y el registro de las intervenciones que se han realizado para su resolución.

Tenemos tres tipos de usuario en esta aplicación:

- **Usuario Cliente**: Es el usuario al que se le presentan las incidencias y por tanto primer proveedor de información de esta aplicación.

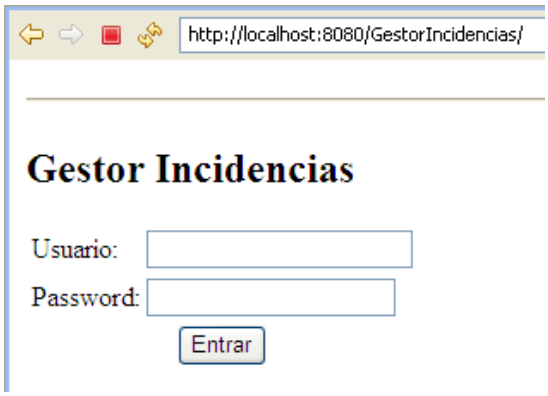
- **Usuario Técnico**: Son los encargados de resolver las incidencias y de registrar en el sistema la descripción de sus intervenciones y el tiempo que han empleado para ello.

- **Usuario Administrador**: Es el usuario que verifica la corrección de los datos introducidos y responsable de la asignación de las incidencias a los técnicos más indicados.

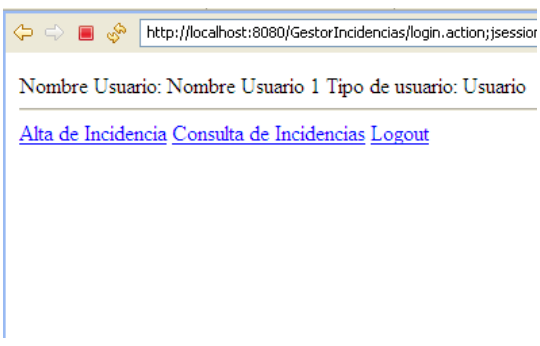
Para cada tipo de usuario la aplicación provee de las opciones necesarias para la realización de sus acciones que detallamos a continuación.

Opciones de usuario cliente

Identificación en el sistema:

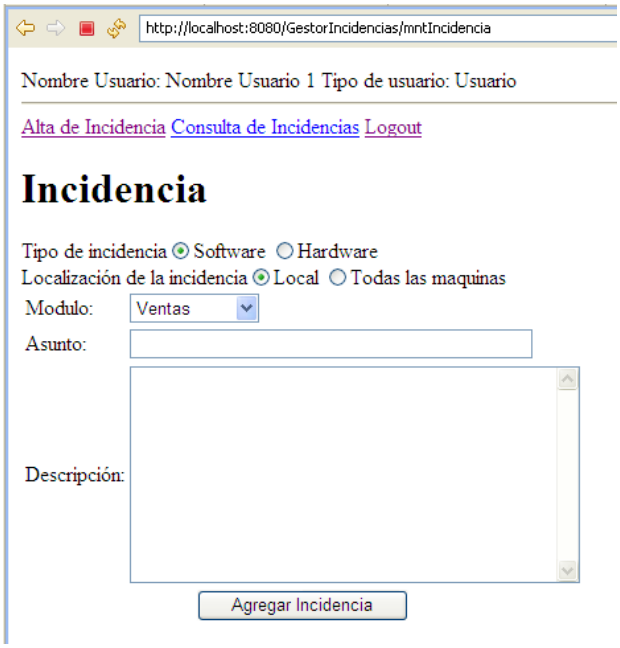


Para poder utilizar la aplicación cualquier usuario debe identificarse en el sistema introduciendo su identificador de usuario y su password.



Tras la validación de la identificación el sistema muestra las opciones que están activas para el usuario con perfil Cliente.

Registro de incidencias:



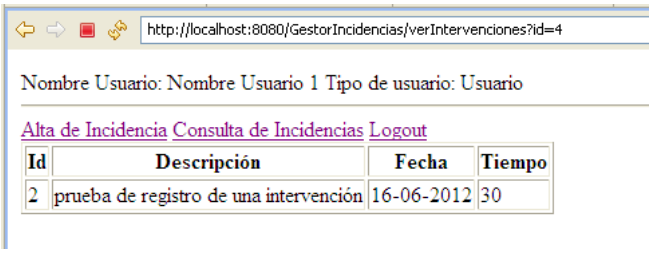
Al pulsar la opción de "Alta de Incidencia" el sistema presenta la pantalla de entrada de datos que el usuario deberá cumplimentar.

Una vez cumplimentado los datos el usuario deberá pulsar el botón "Agregar Incidencia".

Consultar las incidencias de la empresa asociada al usuario:

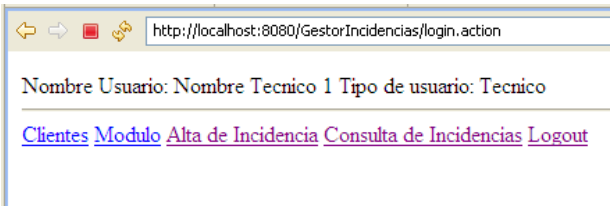
Al pulsar la opción "Consulta de incidencias" el sistema presenta una pantalla con unos selectores de filtro que permiten acotar la consulta.

Consultar las intervenciones de una incidencia



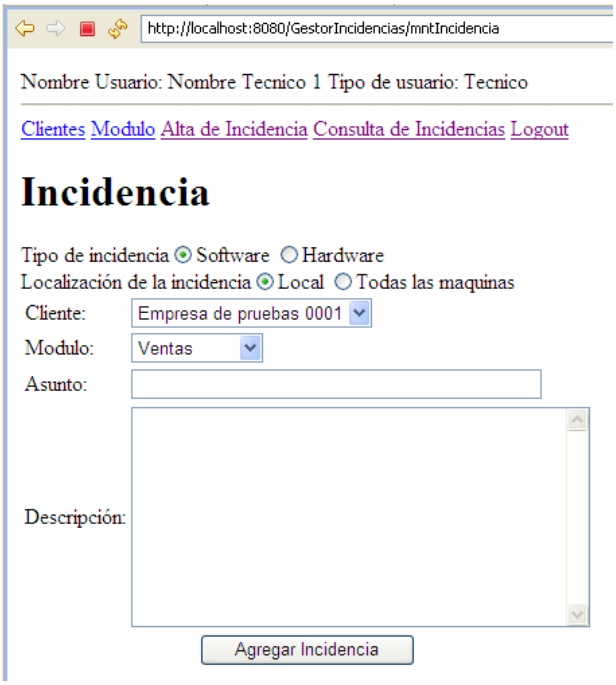
Desde la pantalla de consulta al hacer clic sobre la opción "Ver" de la columna Intervenciones, el usuario podrá consultar todas las intervenciones que se han realizado sobre esa incidencia.

Opciones del usuario Técnico



Una vez identificado en el sistema aparecen todas las opciones del usuario con perfil Técnico.

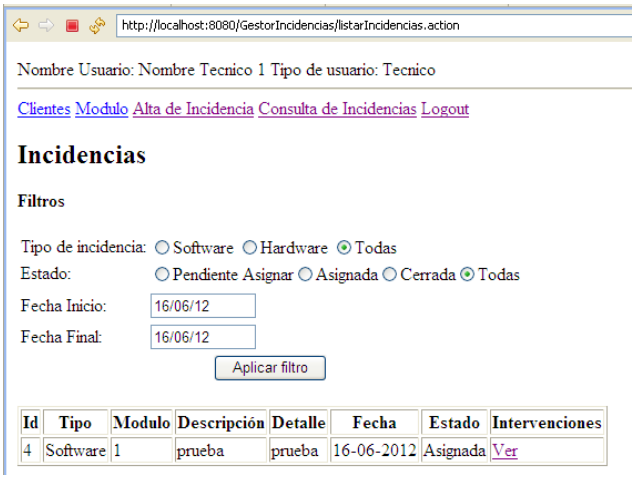
Registro de Incidencia



Al dar de alta una incidencia el sistema le permite seleccionar una empresa cliente.

Una vez actualizada la incidencia le quedará asignada directamente.

Consulta de Incidencias



La opción de consulta de incidencias permite mediante una serie de filtros acotar la consulta, solo presentará las incidencias que el técnico tiene asignadas.

Registro y consulta de intervenciones

Al hacer clic sobre la opción "Ver" de la columna intervenciones el sistema presentará la siguiente pantalla donde se podrá registrar nuevas intervenciones (para cerrar la incidencia bastará con marcar la opción antes de añadir la intervención) y ver todas las intervenciones que tiene registrada esa incidencia.

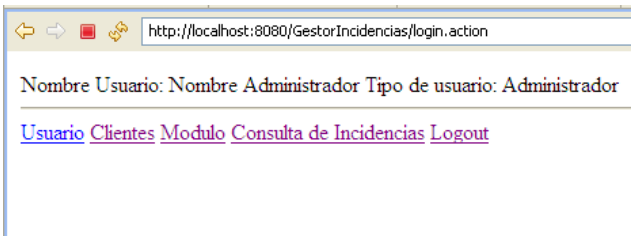
Mantenimiento Clientes

La opción "Clientes" permitirá gestionar los clientes del sistema.

Mantenimiento de Módulos

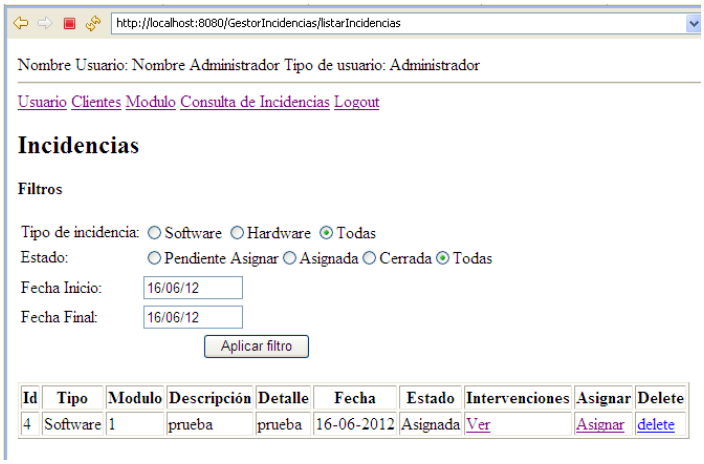
Desde la opción de "Módulos" se podrá gestionar los la BBDD de módulos.

Opciones del usuario Administrador



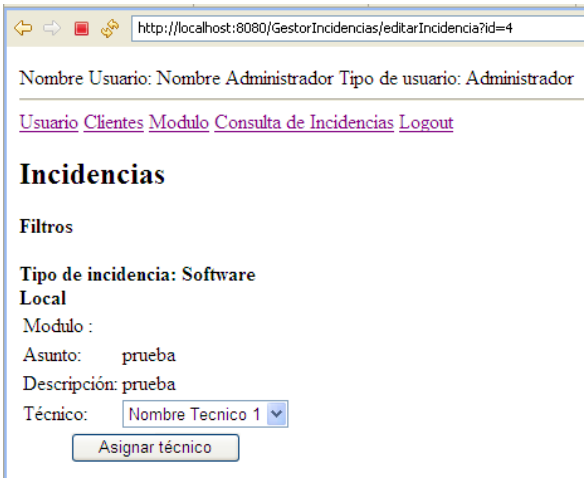
Una vez identificado en el sistema aparecen todas las opciones del usuario con perfil Administrador.

Consulta de incidencias



En esta opción el administrador podrá consultar todas las incidencias del sistema, asignarles un técnico o eliminarlas.

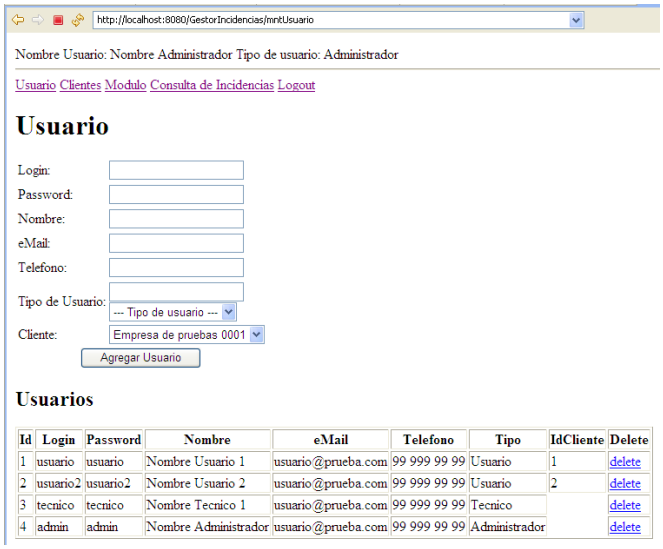
Asignar Técnico



Al acceder a la opción de asignar se presentará los datos relativos a la incidencia y un selector de técnicos para la asignación

Esta opción también puede ser usada para reabrir una incidencia.

Mantenimiento de Usuarios



Desde esta opción el administrador podrá mantener los usuarios del sistema.

El resto de opciones del administrador comparten la misma funcionalidad que la descrita para el usuario técnico.