# DEncuesta

**Chavero Adrien**

ETIS

**Jose Juan Rodriguez**

14/01/09

# 2. Dedicatoria i agradecimientos

Me gustaría ante todo agradecer a mi familia por su apoyo durante todo esta carrera de ETIS.  Y mas por su amor incondicional durante toda mi vida.

También doy las gracias al Consultor Jose Juan Rodriguez que ha sabido responder mis pesadas, y a veces, incoherentes preguntas.

Gracias también a todos los alumnos que hicieron el TFC en el área JEE antes que yo. Han sido de gran ayuda para inspirarme en la realización de esta memoria.

# 3. Resumen

Esto es la memoria del TFC de los estudios ETIS en el área JEE. En este trabajo se realiza la planificación, análisis, diseño e implementación de un sistema que gestiona encuestas. El proyecto se titula DEncuesta y permite crear y responder encuestar, así como visualizar sus resultados. Gran parte del trabajo esta dedicado al estudio de Java Enterprise Edition ya que el proyecto se desarrolla con esta tecnología. En concreto se utiliza JSPs, servlets y EJBs con un patrón MVC para desarrollar el sistema. A parte de utilizar el lenguaje Java también se usa HTML y CSS.

# 4. Índice de contenidos

## Índice de contenido

DEncuesta....1
2. Dedicatoria i agradecimientos....2
3. Resumen....3
4. Índice de contenidos....4
5. DEncuesta....6
5.1. Introducción....6
5.1.1. Justificación del TFC y contexto en el cual se desarrolla: punto de partida y aportación ....6
5.1.2. Objetivos del TFC....6
5.1.3. Enfoque y método seguido....7
5.1.4. Planificación del proyecto....7
5.1.4.1. Diagrama de Gantt....8
5.1.4.2. Detalles de las tareas....8
5.1.5. Productos obtenidos....10
5.1.6. Breve descripción de los capítulos de la memoria....10
5.2. Java Enterprise Edition....11
5.2.1. Que es?....11
5.2.2. Enterprise JavaBeans (EJB)....12
5.2.3. Servlets....12
5.2.4. JavaServer Pages (JSP)....12
5.2.5. Contenedores....13
5.2.6. Instalación....13
5.3. Fase de Análisis....14
5.3.1. Descripción textual....14
5.3.2. Casos de uso....15
5.3.2.1. Caso de uso: Registrarse....16
5.3.2.2. Caso de uso: Login....17
5.3.2.3. Caso de uso: VerResultadosEncuesta....17
5.3.2.4. Caso de uso: ResponderEncuesta ....18
5.3.2.5. Caso de uso: CrearEncuesta....18
5.3.2.6. Caso de uso: EliminarEncuesta....19
5.3.2.7. Caso de uso: ListarEncuestas....19
5.3.3. Clases entidad....20
5.3.4. Diagramas de colaboración simplificados....22
5.3.5. Interfaz gráfica ....23
5.3.5.1. Pantalla login....23
5.3.5.2. Pantalla registrarse....23
5.3.5.3. Pantalla listar encuestas....24
5.3.5.4. Pantalla ver resultados encuesta....25
5.3.5.5. Pantalla crear encuesta....27
5.3.5.6. Pantalla crear encuesta secciones....27
5.3.5.7. Pantalla responder encuesta....29
5.3.5.8. Pantalla acerca de....30
5.4. Fase de Diseño....31
5.4.1. Arquitectura en tres capas....31
5.4.2. Tecnologías JEE aplicadas al proyecto y el patrón....32

5.4.3. Revisión de las clases de entidad....35
5.4.4. Gestión de los datos permanentes....35
5.4.5. Diagramas de secuencia....36
5.4.5.1. Diagrama de secuencia de CrearEncuesta....36
5.4.5.2. Diagrama de secuencia de verResultadosEncuesta (accediendo a través de ListarEncuesa)....38
5.5. Implementación....40
5.5.1. Entorno inicial....40
5.5.2. Ultimas decisiones de diseño....41
5.5.3. Estado actual de la implementación....41
5.5.4. Estructura de archivos....42
5.5.5. Persistencia - la base de datos....43
5.5.6. Fase de pruebas....45
5.5.7. Futuro del proyecto; cambios y mejoras....45
5.5.8. Archivos de despliegue ....46
5.6. Conclusiones....48
6. Glosario....49
7. Bibliografía....51
8. Anexo....53

# 5. DEncuesta

## 5.1. Introducción

### 5.1.1. Justificación del TFC y contexto en el cual se desarrolla: punto de partida y aportación

Este TFC es la culminación de los conocimientos adquiridos a lo largo de esta carrera de ETIS (Engnyeria Tecnica de Informatica de Sistemes).

El proyecto al cual hace referencia esta memoria, es un sistema de gestión de encuestas. Con DEncuesta (nombre del proyecto) se puede crear y responder encuestas, así como visualizar sus resultados. El área de este TFC es JEE (Java Enterprise Edition) y por lo tanto el resultado del proyecto se basara en esta tecnología.

DEncuesta pretende ser un sistema útil para toda persona que se vea enfrentada al problema de tratar con encuestas. Por otro lado este trabajo puede ser de utilidad para futuros estudiantes que quieran desarrollar un proyecto JEE.

### 5.1.2. Objetivos del TFC

Los principales objetivos de este proyecto son aprender a gestionar proyectos informáticos, profundizar los conocimientos del lenguaje de programación Java y explorar el mundo JEE.

Por otro lado el proyecto pretende ser una solución al problema de gestionar encuestas. La idea es proporcionar, a empresas, instituciones o individuos, un sitio al cual puedan remitirse cuando necesiten efectuar encuestas. Desde una única pagina web se ofrece la posibilidad de efectuar todo el proceso relativo a una encuesta. En las ultimas encuestas que he participado, una de ellas de la UOC, me enviaron un email con las preguntas y tuve que responder al email con mis respuestas. Con DEncuesta sencillamente me habrían enviado la direccion web (url) de la encuesta y el creador de la encuesta tendría sus resultados en ese mismo instante.

### 5.1.3. Enfoque y método seguido

Todo el proyecto esta planteado, analizado, diseñado e implementado por Adrien Chavero. Todo el proceso esta supervisado por el consultor de TFC, Jose Juan Rodriguez.

Como mencionado anteriormente el área de este TFC es JEE y por lo tanto el proyecto será de desarrollado con esta tecnología.

Aunque los detalles técnicos de JEE, así como el diseño del proyecto quedan pendientes de ser explicados en posteriores puntos de esta memoria, desde el principio de la concepción del proyecto queda claro que con JEE se puede llevar a cabo el programa (JSP, servlets, EJBs, JSF, etc... , permiten conceptualmente programar DEncuesta).

Los costes de este proyecto son bastante elevados, entendiendo por costes principalmente tiempo dedicado al desarrollo. En una búsqueda he encontrado un proyecto similar (pero muy bien acabado) que tuvo un coste de 2000 horas con un equipo de 3 personas. Por lo tanto estimo que, si me centro en los aspectos mas importantes y elimino opciones y acabado, el proyecto puede estar finalizado en los términos (4 meses para todo el desarrollo).

El método seguido esta estrechamente relacionado con los trabajos parciales que hay que devolver. Estos plantean el desarrollo en cuatro fases: planificación, análisis, diseño y implementación. Mas adelante hay una descripción de estas fases. También es necesario dedicar un tiempo al estudio de la tecnología JEE. Todo el trabajo dedicado a este proyecto quedará reflejado en esta memoria.

### 5.1.4. Planificación del proyecto

Una vez definida la idea general del proyecto es momento de hacer la planificación temporal de todo el trabajo.

Las entregas parciales del proyecto determinan fechas clave para cada fase del desarrollo. Estas son:

PAC1 – Planificación:  01/10/08

PAC2 – Análisis y diseño: 05/11/08

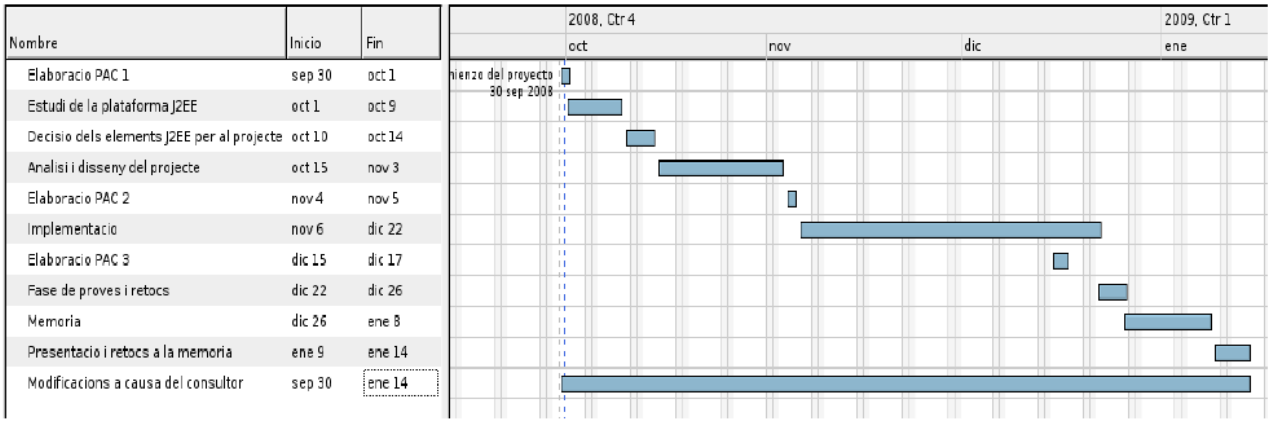
PAC3 – Implementación (no total) y pruebas: 17/12/08

Estos 3 trabajos son de carácter opcional pero son de gran ayuda, ya que el consultor hará una valoración del trabajo efectuado.

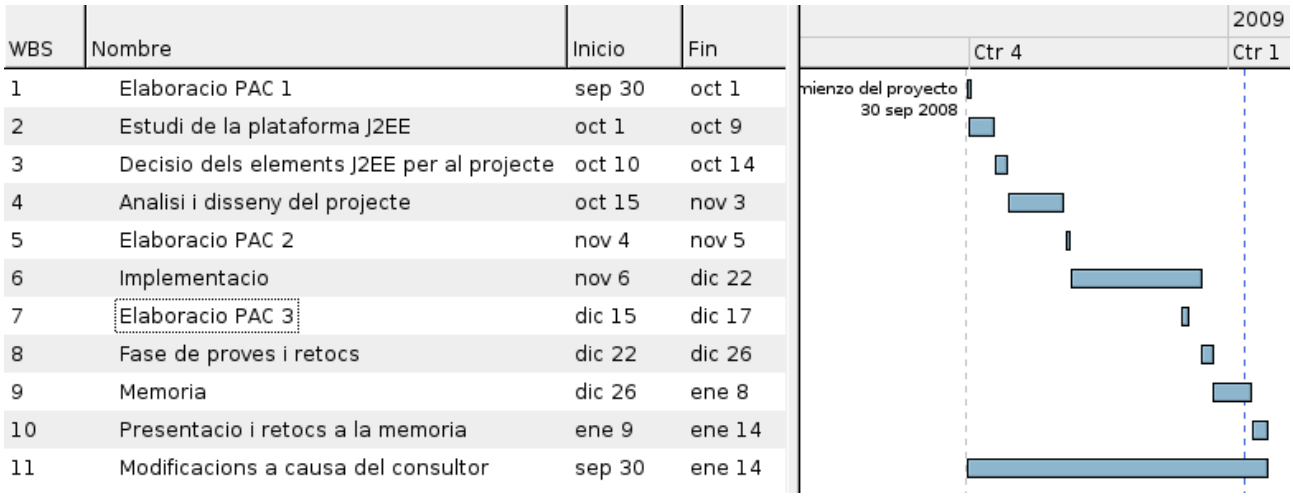
La entrega final esta fechada el 14/01/08

Un diagrama de Gantt recoje toda la planificación.

### 5.1.4.1. Diagrama de Gantt




(Abajo el mismo diagrama con un zoom mas pequeño:)




### 5.1.4.2. Detalles de las tareas

A continuación están los detalles de cada tarea.

<u>Estudio de la plataforma JEE</u>

JEE ofrece muchas y complejas opciones; antes de lanzarse en el desarrollo es necesario hacer un estudio de esta plataforma e identificar que elementos usaremos mas tarde. En el apartado 3 de esta memoria se hace un breve resumen de que es JEE.

Análisis

En esta fase se traducen los requisitos del sistema a un lenguaje mas formal. Normalmente los requisitos provienen de la fase de recolecta y documentación de requisitos. No obstante en este trabajo, se incluye esta fase en la del análisis, obteniendo el modelo de análisis directamente. En la fase de análisis no se tiene en cuenta aun como implementaremos el proyecto, sencillamente se describe el programa desde una perspectiva funcional (con el enfoque OO).

Los resultados de esta fase serán:

- Los casos de uso
- Las clases fundamentales
- Algún diagrama de colaboración/estados
- Esquemas de las interfaces gráficas

Diseño

Esta fase se sitúa al final de la elaboración y principios de la construcción. En el diseño ya tendremos en cuenta que usaremos JEE e intentaremos de reutilizar: clases, componentes, patrones y frameworks. Hay que revisar los diagramas de clases del análisis para adecuarlos al lenguaje Java (eliminar herencias múltiples, etc...). También es el momento de diseñar la persistencia.

Para tener las ideas claras en la fase de implementación, crearemos un par de diagramas de secuencia que muestren como funciona la aplicación.

Implementación

En esta fase programaremos todo le código que sera el producto final, tomando todas la decisiones de implementación necesarias. Se da por entendido que la fase de pruebas esta incluida en la implementación.

Memoria y presentación

Esta ultima tarea da como fruto este mismo documento y una presentación PowerPoint que sera un resumen de todo el trabajo realizado para llevar a cabo este proyecto.

### 5.1.5. Productos obtenidos

El esfuerzo dedicado en este trabajo dará como fruto una aplicación web que gestiona encuestas, esta memoria y una presentación que resume el trabajo.

### 5.1.6. Breve descripción de los capítulos de la memoria

Los capítulos de esta memoria están estrechamente relacionados con las tareas temporales que hemos descrito en el apartado planificación.
Tendremos por lo tanto un capitulo introducción (lo que ha leído hasta ahora), a continuación uno llamado “Java Enterprise Edition” que describe la tecnología JEE; después “Fase de análisis” (con el contenido que hemos descrito antes), “Fase de diseño” (también descrito en la planificación) y “Implementación” con todo lo relativo a la programación del proyecto.
Por ultimo tenemos las conclusiones, un glosario (con todos los términos relevantes), la bibliografía y los anexos.

## 5.2. Java Enterprise Edition

### 5.2.1. Que es?

Java Platform, Enterprise Edition o Java EE es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de N niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma Java EE está definida por una especificación. Similar a otras especificaciones del Java Community Process, Java EE es también considerada informalmente como un estándar debido a que los suministradores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son conformes a Java EE; estandarizado por The Java Community Process / JCP.

Java EE incluye varias especificaciones de API, tales como JDBC, RMI, e-mail, JMS, Servicios Web, XML, etc y define cómo coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servlets, portlets (siguiendo la especificación de Portlets Java), JavaServer Pages y varias tecnologías de servicios web. Esto permite al desarrollador crear una Aplicación de Empresa portable entre plataformas y escalable, a la vez que integrable con tecnologías anteriores. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componentes desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de en tareas de mantenimiento de bajo nivel.

(Fuente: Wikipedia)

A continuación definimos ciertos elementos del JEE:

### 5.2.2. Enterprise JavaBeans (EJB)

Los EJB proporcionan un modelo de componentes distribuido estándar del lado del servidor. El objetivo de los EJB es dotar al programador de

un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) 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.

Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor.

Existen 2 tipos fundamentales de EJBs:

- EJB de entidad que encapsulan datos (y pueden hacerse persistentes)
- EJB de sesión que sirven de fachada para los servicios proporcionados por el servidor.

### 5.2.3. Servlets

La palabra servlet deriva de otra anterior, applet, que se refería a pequeños programas escritos en Java que se ejecutan en el contexto de un navegador web. Por contraposición, un servlet es un programa que se ejecuta en un servidor.

El uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web.

### 5.2.4. JavaServer Pages (JSP)

JavaServer Pages (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

Las JSP's permiten la utilización de código Java mediante scripts. Además es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Librerías de Etiquetas (TagLibs o Tag Libraries) externas e incluso personalizadas.

JSP puede considerarse como una manera alternativa, y simplificada, de construir servlets. Es por esto que una página puede hacer todo lo que un servlet puede hacer, y viceversa.

### 5.2.5. Contenedores

Para poder ejecutar una aplicacion JEE es necesario un contenedor que sea JEE compliant. Estos son normalmente conocidos como Application Server o Servidores de aplicación. Los EJB, servlets y JSP "corren" dentro del estos.

Existe un subgrupo de contenedores que solo albergan servlets y JSP (no los EJB), estos son llamados contenedores de servlets (ejemplo: Tomcat).

Algunos de los servidores de aplicaciones JEE certificados son los siguientes:

- Jboss: Existe una versión de código abierto soportada por la comunidad y otra empresarial.
- WebLogic Application Server 10.0 from BEA Systems
- IBM WebSphere Application Server Community Edition 2.0, based on Apache Geronimo
- Oracle Containers for Java EE 11
- Glassfish, un servidor de aplicaciones de código abierto de Sun

### 5.2.6. Instalación

La instalación JEE para le proyecto se efectúa en un AMD x2 3800 con un sistema operativo Linux, Debian Testing.

Se tuvo que instalar el archivo java_app_platform_sdk-5_06-linux.bin descargado desde la pagina web oficial de SUN. (En este archivo viene incluido el Java Standard Edition). El servidor de aplicaciones que viene con este paquete es Glassfish v2ur2 y usa la base de datos Derby para la persistencia.

Para probar el JEE se descargó NetBeans y con sencillo programa comprobamos que todo funciona correctamente.

Para probar el proyecto, como se puede ver en la sección de implementación, se instaló también el servidor de aplicaciones JBoss v5.0.0GA en un Windows XP.

## 5.3. Fase de Análisis

Como se menciona en el apartado de Planificación, en esta fase vamos a describir el programa desde el punto de vista funcional. Para ello mostraremos los casos de uso, un diagrama de las clases de entidad, un par de diagramas de colaboración simplificados y la interfaz gráfica. No obstante antes continuar, haremos una descripción textual del sistema.

### 5.3.1. Descripción textual

El proyecto DEncuesta pretende ser un sistema que gestiona todo lo relativo a encuestas. Por lo tanto queremos que los usuarios puedan crear encuestas, responderlas y ver los resultados.

Para responder una encuesta requerimos que el usuario pueda identificarse, de lo contrario no podríamos saber que respondió cada persona. Por otro lado, el hecho que tengan que estar identificados es una primera medida de seguridad; queremos que las encuestas muestren resultados útiles y consecuentemente una persona solo puede responder una sola vez un cuestionario. Es por esta ultima razón que para responder cuestionarios anónimos (el sistema no muestra jamas que respondió cada persona), el sistema pida que el usuario este registrado y debidamente identificado.

Para crear una encuesta también nos interesa que el usuario que crea este identificado. Eso es debido a que las encuestas pueden ser privadas, es decir que solo el que creo la encuesta pueda ver los resultados.

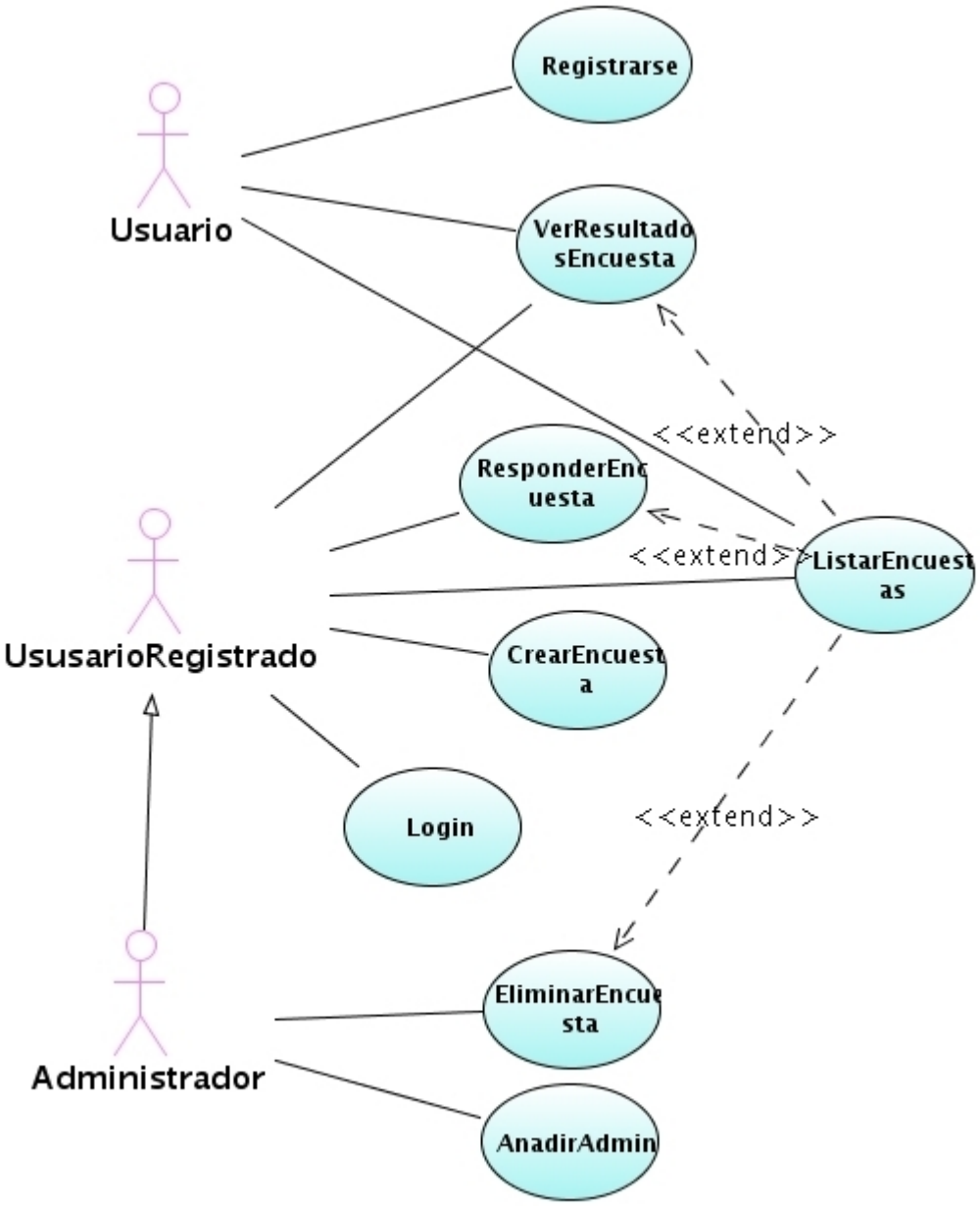
Para ver los resultados no requerimos que la persona interesada en los datos este identificada.

Por otra parte parece ser necesario que alguien pueda borrar encuestas, ya que pueden ser inadecuadas (racistas, ofensivas, etc..). Para ello tendremos un tipo especial de usuario que llamaremos administrador.

Por ultimo, después de algo de reflexión, vemos que el sistema ha de ser capaz también de listar todas las encuestas que se han creado en el sistema. Sino las personas interesadas solo podrían ver resultados (o responder) las encuestas que alguien le indico donde estaban (urls). El listado muestra todas las encuestas y para cada una de ellas un link para responderla y otro para ver los resultados. (Evidentemente el sistema solo permitirá ver los resultados al creador en caso de que la encuesta sea privada).

### 5.3.2. Casos de uso

La descripción textual anterior se puede formalizar con un casos de uso. Los actores están claros: Usuarios, Usuarios registrados y Administradores.


Registrarse
Usuario
VerResultado
sEncuesta
<<extend>>
ResponderEnc
uesta
<<extend>>
ListarEncuest
as
UsusarioRegistrado
CrearEncuest
a
Login
<<extend>>
EliminarEncue
sta
Administrador
AnadirAdmin

#### 5.3.2.1. Caso de uso: Registrarse

Resumen de la funcionalidad: Permite crear una cuenta en DEncuesta

Actores: Usuario

Casos de uso relacionados: Login

Precondición: El usuario no tiene aun una cuenta.

Postcondición: La persona tiene una cuenta y queda identificado en el sistema (y podrá identificarse mas adelante, en otra session) o la persona no ha podido registrarse con lo cual no puede identificarse.

Descripción: Al usuario sin identificar se le pedirá su DNI, su nombre, su primer apellido, su segundo apellido, su email, su país y un password. Si ha rellenado bien los datos se le creara la cuenta en DEncuesta. Si ya existía una cuenta con el mismo DNI se le denegara la creación de cuenta. Al crear la cuenta quedara directamente identificado en el sistema. La cuenta persistente permite al usuario identificarse en futuras ocasiones.

#### 5.3.2.2. Caso de uso: Login

Resumen de la funcionalidad: Permite identificarse en el sistema

Actores: UsuarioRegistrado

Casos de uso relacionados: Registrarse

Precondición: ninguna

Postcondición: El usuario esta identificado o no.

Descripción: El usuario da su DNI así como su password. Si la cuenta existe y el password es correcto, el usuario esta identificado, en caso contrario no lo esta.

#### 5.3.2.3. Caso de uso: VerResultadosEncuesta

Resumen de la funcionalidad: Permite ver los resultados de una encuesta

Actores: Usuario, UsuarioRegistrado (por herencia Administrador también)

Casos de uso relacionados: ListarEncuestas

Precondición: La encuesta existe.

Postcondición: ninguna

Descripción: El usuario puede poner la url de la encuesta para ver sus resultados (la url tiene como parámetro el id de la encuesta) o llegar aquí a través de ListarEncuestas. Si la encuesta aun no tiene participantes el sistema se lo indicara en vez de mostrar resultados. Si la encuesta es privada, solo el creador podrá ver los resultados; si el usuario no es el creador, el sistema se lo notificara. Si la encuesta es anónima solo se mostraran los resultados globales y no se podrán ver los resultados individuales.

#### 5.3.2.4. Caso de uso: ResponderEncuesta

Resumen de la funcionalidad: Permite responder una encuesta

Actores: UsuarioRegistrado

Casos de uso relacionados: ListarEncuestas

Precondición: El usuario registrado esta debidamente identificado y no existe ninguna respuesta a esta encuesta por parte de este mismo usuario.

Postcondición: La respuesta esta almacenada en el sistema.

Descripción: El usuario puede poner la url de la encuesta para responderla (la url tiene como parámetro el id de la encuesta) o bien llegar aquí a través de ListarEncuestas. Un usuario identificado solo puede responder una vez cada encuesta. Si la encuesta no es anónima tendrá la opción de poner un comentario a cada pregunta que responda. Al terminar, la respuesta del usuario quedara almacenada en el sistema.

#### 5.3.2.5. Caso de uso: CrearEncuesta

Resumen de la funcionalidad: Permite crear una encuesta

Actores: UsuarioRegistrado (por herencia Administrador también)

Casos de uso relacionados: ninguno

Precondición: El usuario registrado esta debidamente identificado.

Postcondición: La encuesta esta almacenada en el sistema.

Descripción: El usuario registrado y identificado, da un nombre a la encuesta y escoge si es anónima y si es privada. A continuación añade/elimina Secciones y cuando la encuesta esta lista la almacena en el sistema.

#### 5.3.2.6. Caso de uso: EliminarEncuesta

Resumen de la funcionalidad: Permite borrar una encuesta.

Actores: Administrador

Casos de uso relacionados: ListarEncuesta

Precondición: El usuario esta identificado y es administrador. La encuesta a borrar existe.

Postcondición: La encuesta esta borrada así como todas las repuestas de esta.

Descripción: El administrador puede borrar encuestas poniendo directamente la url (con un parámetro id que identifica la encuesta) o bien a través de ListarEncuestas. Al terminar, la encuesta y las respuestas de esta estaran eliminadas.

#### 5.3.2.7. Caso de uso: ListarEncuestas

Resumen de la funcionalidad: Permite listar las encuentas presentes en el sistema

Actores: Usuario, UsuarioRegistrado (por herencia Administrador también)

Casos de uso relacionados:

VerResultadosEncuesta, ResponderEncuesta, EliminarEncuesta

Precondición: ninguna

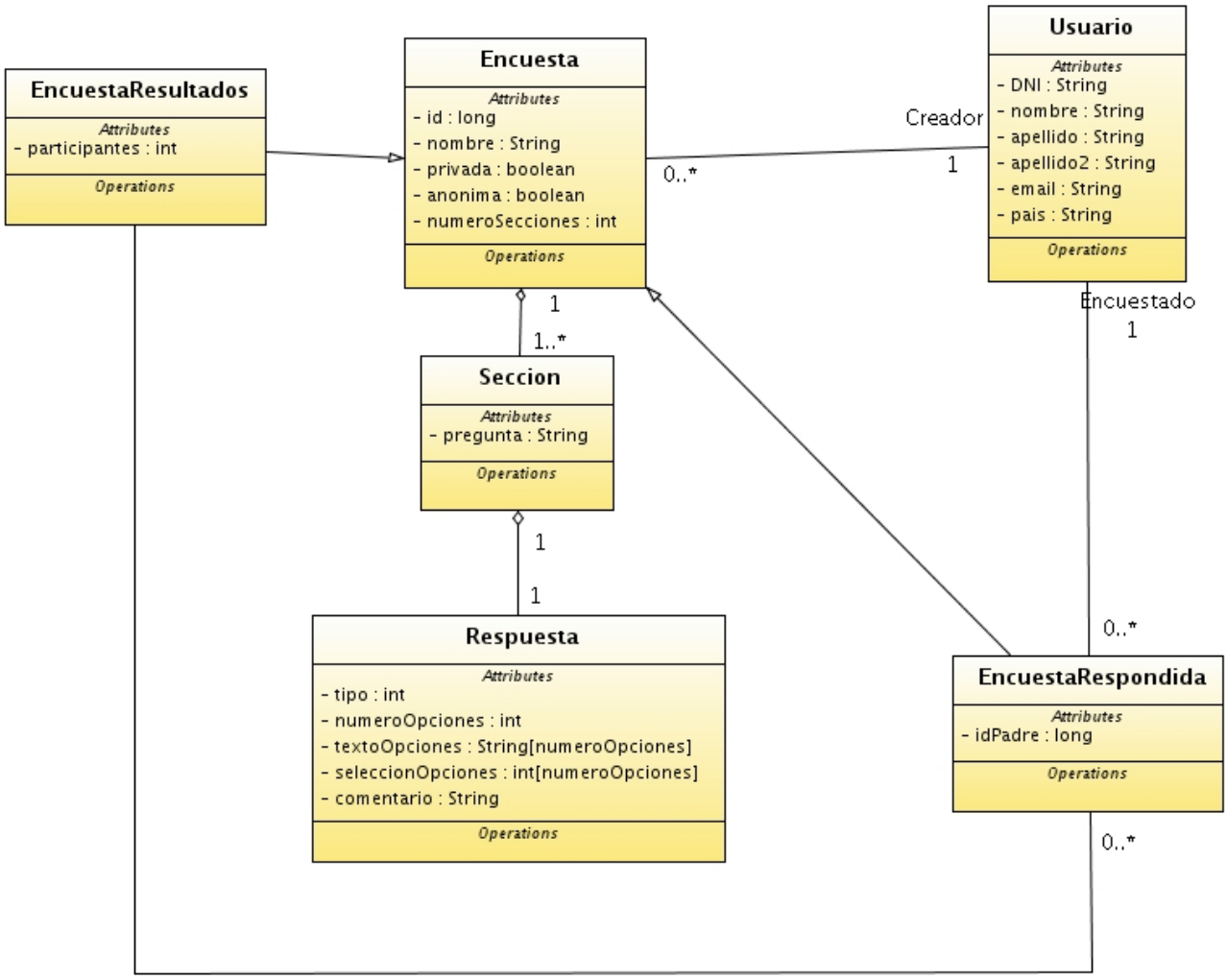
Postcondición: ninguna

Descripción: Lista todas la encuestas y para cada una de ellas pone un link para responder la encuesta y otro para ver los resultados. Si el usuario que usa ListarEncuestas es Administrador, además pondrá un link para borrar la encuesta.

### 5.3.3. Clases entidad

Después de analizar los casos de uso se puede crear las clases de entidad.

El siguiente diagrama reúne todas la clases de entidad con sus relaciones y asociaciones.




Explicaciones:

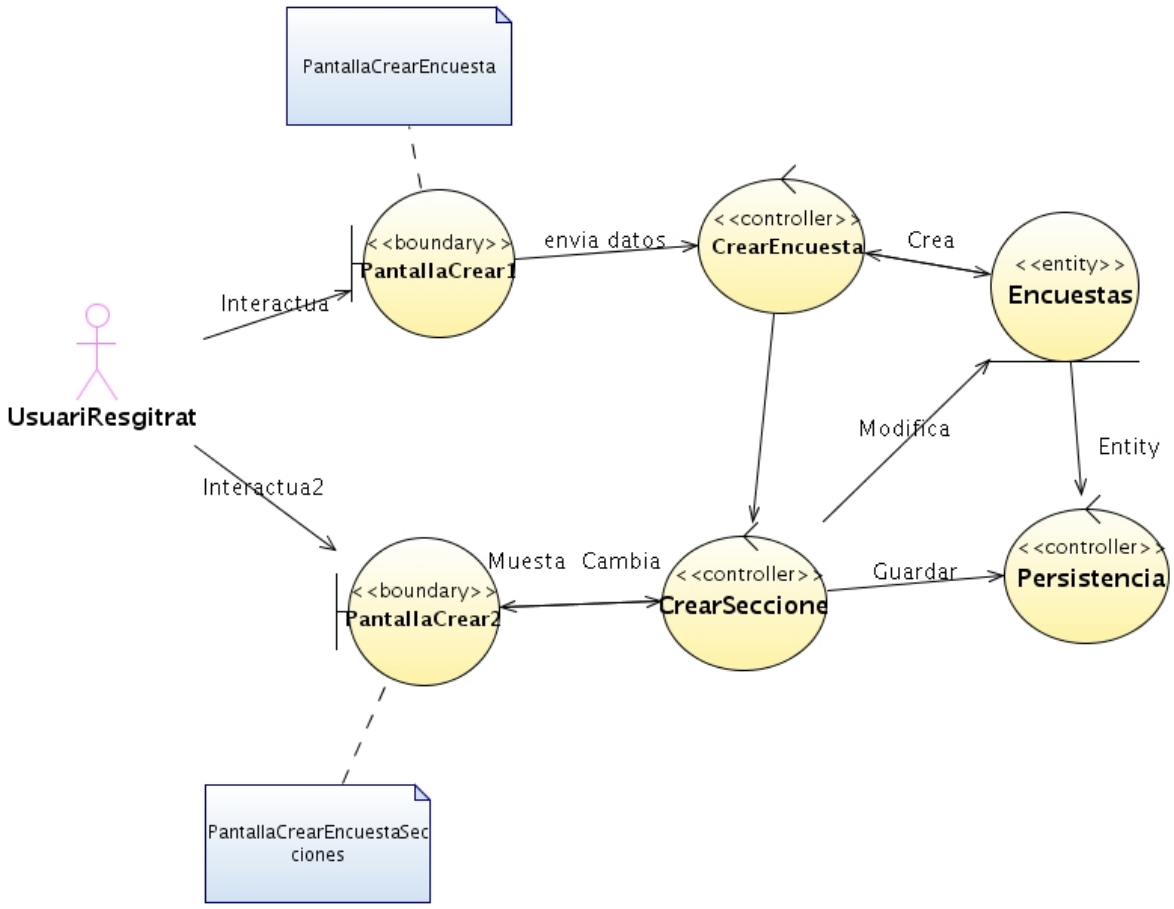
En una Encuesta hay varias secciones. En una Sección solo hay una pregunta (atributo String) con su respuesta asociada (objeto Respuesta). Cada objeto Respuesta tiene un tipo (lista despegable,

radio button, check box), un numeroOpciones (que dice por ejemplo cuantos radio buttons hay), el String textoOpciones[] (por ejemplo: "si" el primer radio button, "no" el segundo, {"si","no"}), int seleccionOpciones[] ( que tiene un 1 por cada opción activada, por ejemplo {1,0} del radio button anterior significa que se escogió "si") y un comentario (para los cuestionarios no anónimos).

He considerado 3 tipos de objetos encuestas:

- Encuesta, que se usa como plantilla para la encuesta, el valor de seleccionOpciones es solo para guardar los valores por defecto de la encuesta. Solo se usa este tipo de objeto para la creación de encuestas y para crear la vista al responder las encuestas.
- EncuestaRespondida, que se usa para guardar los datos de la encuesta que se acaba de responder y evidentemente para ver los resultados individuales. La única diferencia con Encuesta es que aquí el seleccionOpciones guarda los resultados y que existe una nueva referencia a Usuario (el usuario que respondió la encuesta) además del que viene con la súper clase, el usuario creador.
- EncuestaResultados es la tercera. Se usa temporalmente, es decir al pedir los resultados globales de una encuesta se crea es tipo de objeto. La primera diferencia con su súper clase Encuesta es un atributo participantes que nos dice cuantas personas han participado en la encuesta en cuestión. La segunda es que tiene una colección de objetos EncuestaRespondida (un EncuestaRespondida por cada respuesta a la encuesta). La tercera es una diferencia a nivel de instancia, int seleccionOpciones[] ya no tiene valores 1 o 0 sino que el numero de personas que respondieron cada opción (por ejemplo en un radio button "si", "no", seleccionOpciones podría valer {20, 80} que significa que 20 respondieron que si y 80 que no).

## 5.3.4. Diagramas de colaboración simplificados



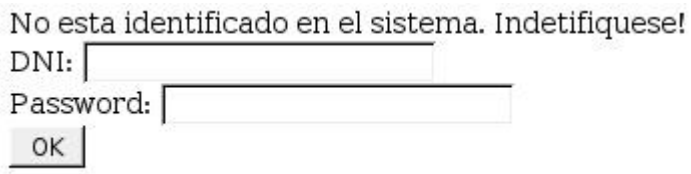
## 5.3.5. Interfaz gráfica

El ultimo punto de la fase de análisis es definir la interfaz gráfica. Como que aún no sabemos como vamos a implementar el sistema, pongo las pantallas que interactúan con el usuario de forma generica (mas tarde decidiremos con que las implementamos (html, applets, etc..)). Podemos suponer que existe un menú que nos lleva a estas ventanas. El menu esta formado por los enlaces:

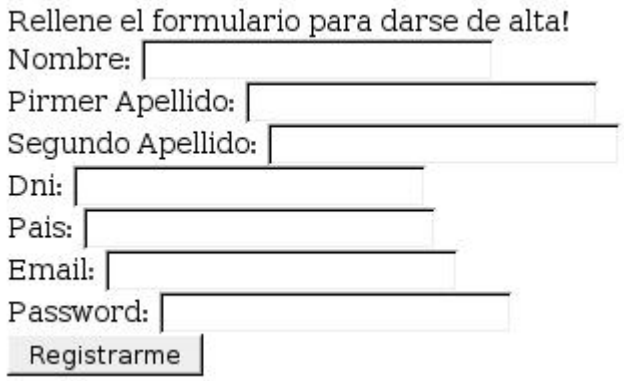
- Login
- Registrarse
- Crear
- Listar
- Acerca de

Las pantallas son las siguientes:

### 5.3.5.1. Pantalla login




### 5.3.5.2. Pantalla registrarse



### 5.3.5.3. Pantalla listar encuestas

Estas son las encuestas disponibles actualemente en el sistema. (Nota: los resultados de las encuestas privadas, solo apareceran si es usted el creador de la encuesta (ha de estar identificado en el sistema))

| Nombre | Creador | Anonima | Resultados | Reponder |
|---|---|---|---|---|
| Prueba | Arc Sed, Descra | No | Resultados | Responder |
| Mas pruebas | Arc Sed, Descra | No | Resultados | Responder |
| Inteligencia | Arc Sed, Descra | Si | Resultados | Responder |
| Pluto | tio toi, Otro | No | Resultados | Responder |

### 5.3.5.4. Pantalla ver resultados encuesta

# Prueba

## Creada por Arc Sed, Descra

---

## Han participado 5 personas.

---

Te gusta Linux?

**1** personas han seleccionado: **Si**

20%

**4** personas han seleccionado: **No**

80%

Prefieres Linux o Windows?

**1** personas han seleccionado: **Linux**

20%

**4** personas han seleccionado: **Windows**

80%

Que sistemas operativos tienes instalados?

**1** personas han seleccionado: **Linux**

20%

**5** personas han seleccionado: **Windows**

100%

**1** personas han seleccionado: **MacOs**

20%

**2** personas han seleccionado: **Otros**

40%

---

## Los siguientes usuarios respondieron la encuesta, puede ver sus respuestas siguiendo el link.

Arc Sed, Descra Ver resultados
Mismo Mismito, Yo Ver resultados
Tio Macho, Pepe Ver resultados
Aquel Dealli, El Ver resultados
tio toi, Otro Ver resultados

### 5.3.5.5. Pantalla crear encuesta

**Defina su nueva encuesta**

Titulo/nombre:

Anonima: ☐ (En las encuestas anonimas no se muestran las respuestas de cada individuo, solo se pueden visualizar los resultados globales.)

Privada: ☐ (En las encuestas privadas solo el usuario que creo la encuesta puede ver los resultados globales/individuales.)

Continuar

### 5.3.5.6. Pantalla crear encuesta secciones

Posibilidad 1:

# TFC

**Creada por Arc Sed, Descra**

Modo Edición - Guardar encuesta

---

La encuesta esta VACIA!!
Pulse Crear para comenzar una nueva seccion. No olvide pulsar 'Añadir a la encuesta' cuando crea que la seccion ya esta bien.

---

**Seccion editable**

Pulsa aqui para añadir una pregunta en la que la respuesta es escojer una sola opción

Crear radio button

Pulsa aqui para añadir una pregunta en la que la respuesta es escojer varias opciones

Crear checkbox

Posibilidad 2:

# TFC

## Creada por Arc Sed, Descra

Modo Edición - Guardar encuesta

---

**Te esta costando?**

○ Si
○ No
Comentario:

subir
bajar
borrar

**Que has hecho?**

☐ Leer TFCs pasados
☐ No dormir
☐ Dejarlo para muy tarde
Comentario:

subir
bajar
borrar

---

## Seccion editable

Pregunta (escriba abajo una pregunta que desea formular en la encuesta):

○ Texto de la opción:
◉ Texto de la opción:

Crear una opcion mas

Añadir a la encuesta | Empezar de nuevo esta seccion

### 5.3.5.7. Pantalla responder encuesta

# Mas pruebas

## Creada por Arc Sed, Descra

---

Haces deporte?

○ Mucho
○ Poco
○ Nada
Comentario:

Que deportes te gustan?

☐ Natacion
☐ Esqui
☐ Correr
☐ Winsurf
Comentario:

Enviar

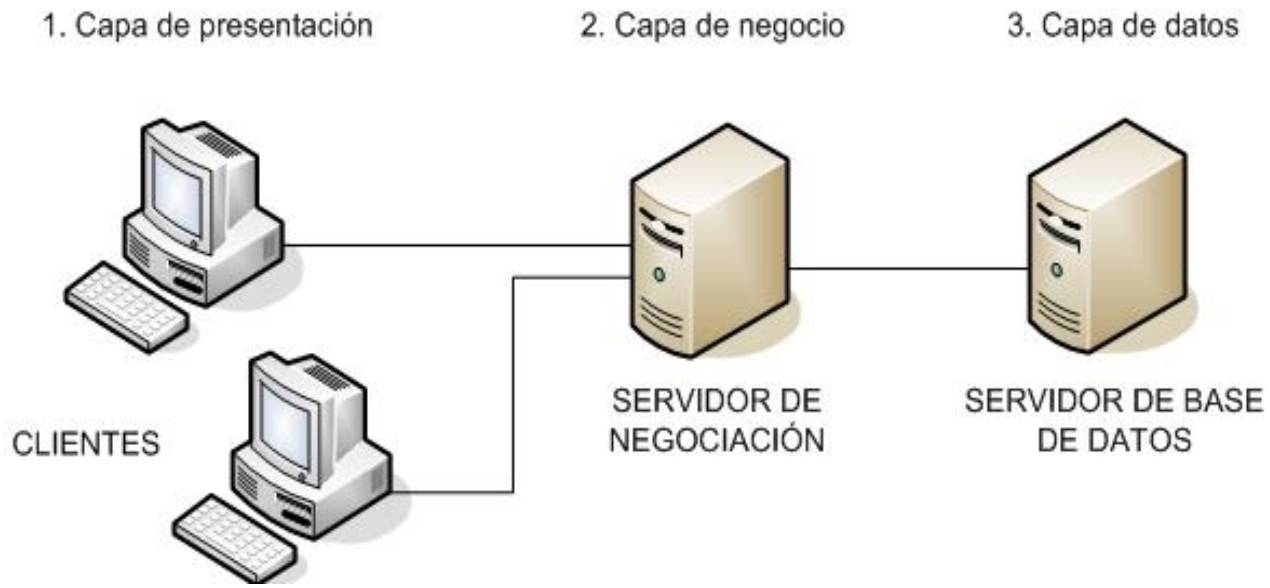
#### 5.3.5.8. Pantalla acerca de...

## 5.4. Fase de Diseño

Como comentamos en la sección planificación, la fase de diseño es el momento en el cual decidimos la manera de implementar el proyecto. Es momento de decidir que patrones vamos a usar, que tecnologías son las mas adecuadas, etc...

### 5.4.1. Arquitectura en tres capas

La aplicación DEncuestas tendrá una arquitectura en tres capas como muestra el siguiente esquema.




Justificación:

Una arquitectura en tres capas ofrece una serie de ventajas:

- Mantiene separada la lógica del negocio de los accesos de los usuarios, disminuyendo los metodos y codigo visible desde el usuario; esto fomenta la seguridad.
- Ofrece portabilidad ya que permite transportar las capas por separado y efectuar el mínimo numero de cambios al pasar de una plataforma a otra.

- Permite aprovechar las capas enteras para nuevas aplicaciones.
- Optimiza el mantenimiento; es mas sencillo hacer cambios, actualizaciones o ampliaciones a la aplicación.

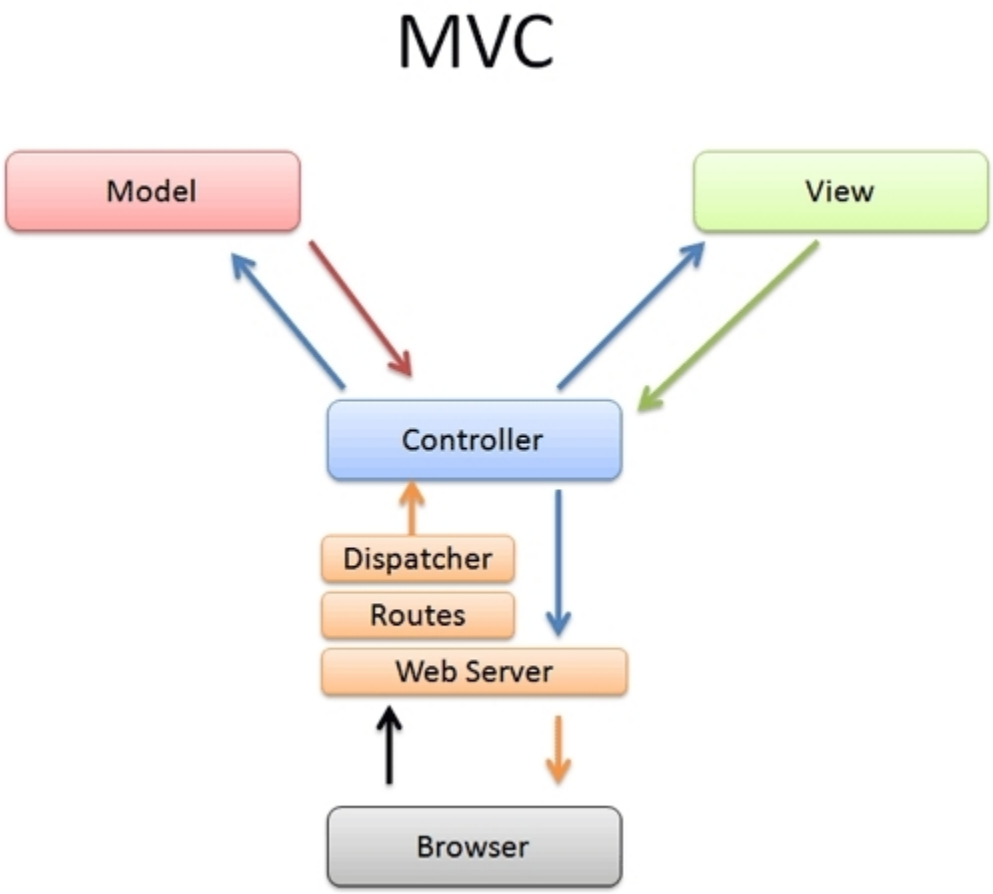
### 5.4.2. Tecnologías JEE aplicadas al proyecto y el patrón

Primero de todo decidimos que el sistema estará basado en paginas web con contenido dinámico y usaremos JEE.

Para el desarrollo del proyecto utilizaremos una variación del patrón MVC. MVC son la siglas de Modela Vista Controlador.

MVC es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario,  y la la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página; el modelo es el sistema de gestión de base de datos y la lógica de negocio; y el controlador es el responsable de recibir los eventos de entrada desde la vista.

Aquí seguido hay un esquema de MVC en el ámbito de paginas web:




De hecho quizás la variación que usamos se aproxima mas al MVC 2, pero en nuestro esquema el controlador es varios Servlets y no uno solo (además no siempre haremos peticiones al controlador sino que a veces llamaremos directamente a la vista).

Antes de continuar definamos como representaremos el modelo, la vista y el controlador.

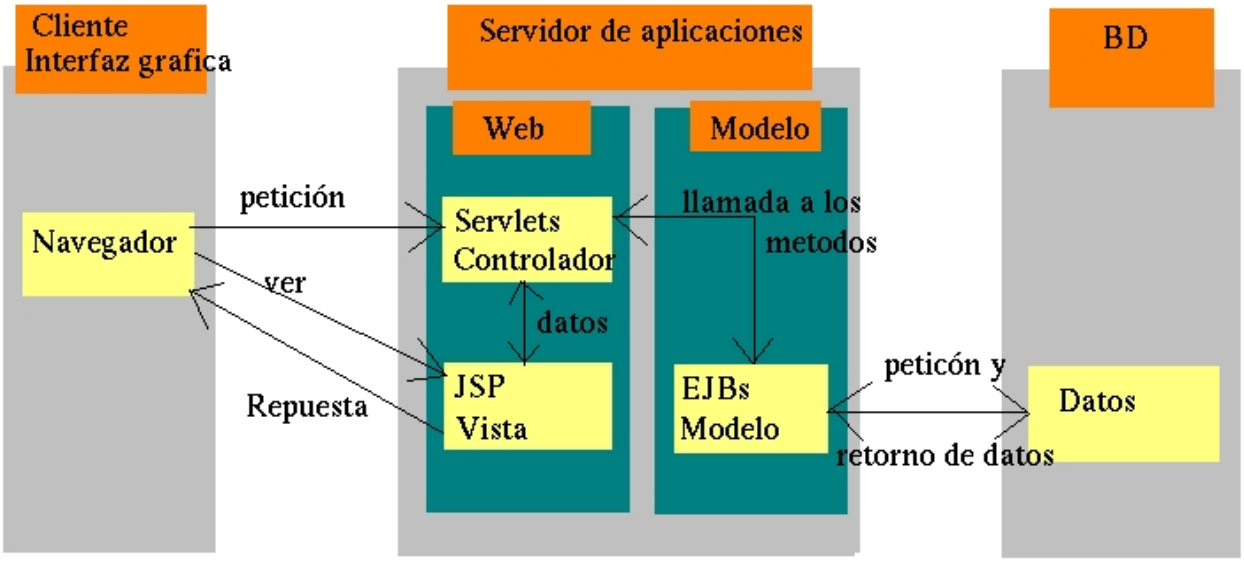
Para la vista he decidido utilizar JSP (Java Server Pages). Recordemos que JSP es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML.

El controlador es una suma de Servlets. Recordemos que un Servlet es un objeto Java que vive dentro de un contenedor de servlets (o bien un servidor de aplicaciones), que recibe parametros y efectua una acción (como mostrar resultados) con esos datos.

En nuestro caso los servlets acabaran siempre por llamar a un JSP (vista) para indicar al usuario el resultado de las operaciones.

El modelo lo representaremos con un Enterprise Java Beans (EJB) de sesión que efectuará todas las operaciones de negocio interactuando con EJB de entidad.

Con un esquema veremos mejor como esta construido el sistema:




Vemos como los JSPs (vista) son los únicos que muestran datos al usuario a través del navegador.

Todas la interacciones que tiene el usuario con el sistema se traducen en la llamada a un servlet (controlador).

Los servlets (controladores) son los únicos que pueden llamar a servicios de los EJBs (modelo).

En el proyecto todos los aspectos del modelo están unificados en un solo EJB llamado DencuestaSession (con su respectiva interfaz DencuestaSessionLocal).

He escogido el sistema JSP-Servlet-EJB principalmente por su flexibilidad. El proyecto se podría haber deserarollado con el

framework struts y podríamos haber ganado facilidad de mantenimiento. Al ser un proyecto relativamente poco extenso, las ventajas no serian muy grandes y sin embargo tendríamos una estructura muy rígida impuesta.

Respecto a la opción de utilizar JSF se puede decir que es innecesaria; todo lo que se puede hacer con JSF se puede hacer con JSP-Servlet-EJB. JSF añade una abstracción que considero opcional en este proyecto.

### 5.4.3. Revisión de las clases de entidad

Es en este momento, fase de diseño, es cuando decidimos el lenguaje de programación con el que se va a implementar el programa. Este lenguaje es Java. Revisando las clases de entidad expuestas en el análisis, vemos que la traducción sera directa ya que no hay herencia múltiple, los nombres están normalizados y los tipos de los atributos están soportados.

Queda por determinar como implementaremos las colecciones de objetos pero eso lo dejaremos para la implementación (sabemos que en Java hay multitud de tipos de colecciones y con eso nos basta para validar las clases a nivel de diseño).

### 5.4.4. Gestión de los datos permanentes

Ciertas clases del diseño han de ser persistentes para que el programa recupere el estado de ejecuciones anteriores. Estas son Encuesta, EncuestaRespondida y Usuario. (Las implementaremos como Enterprise Java Beans de entidad).

Hay dos tipos de persistencia para los EJB de entidad:

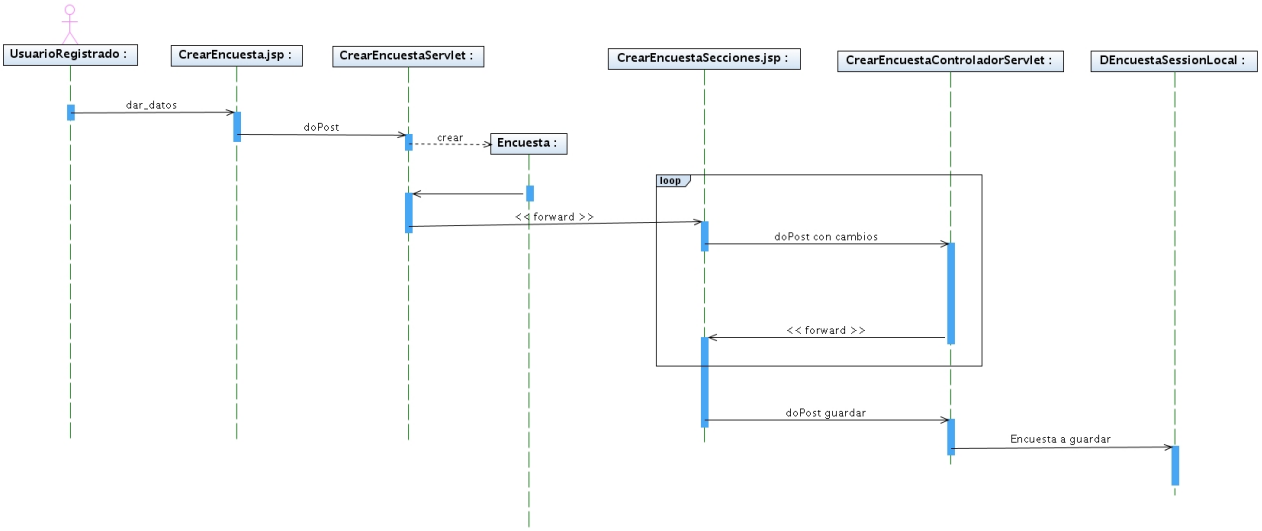
- Persistencia gestionada por el contenedor (CMP): el contenedor se encarga de almacenar y recuperar los datos del objeto de entidad mediante el mapeo de una tabla de la base de datos.
- Persistencia gestionada por el bean (BMP): el propio objeto entidad se encarga, mediante una base de datos u otro 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.

De momento escogemos CMP, aunque durante la fase de implementación verificaremos si coincide con los resultados esperados. En caso contrario, el contenedor no nos proporciona las tablas esperadas, usaremos BMP determinando de manera programable como serán las tablas de la BD.

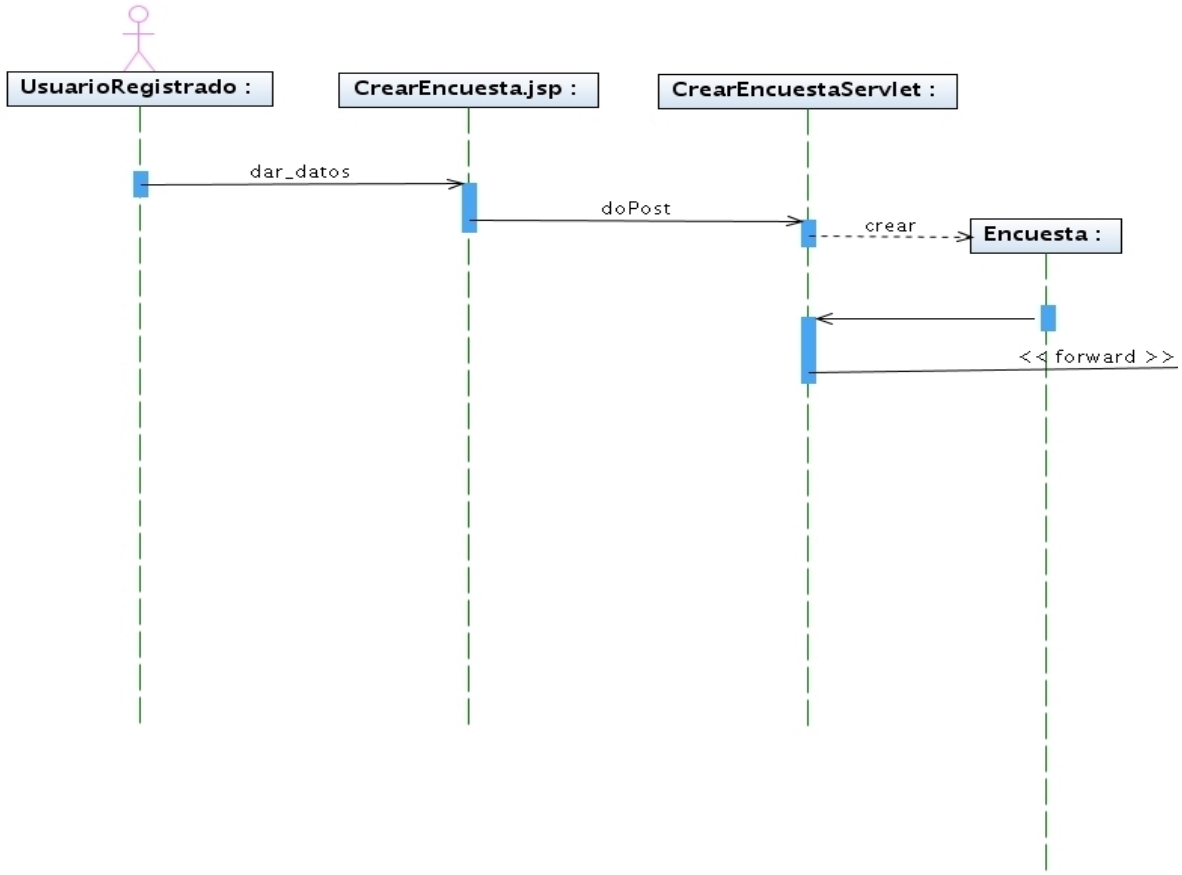
## 5.4.5. Diagramas de secuencia

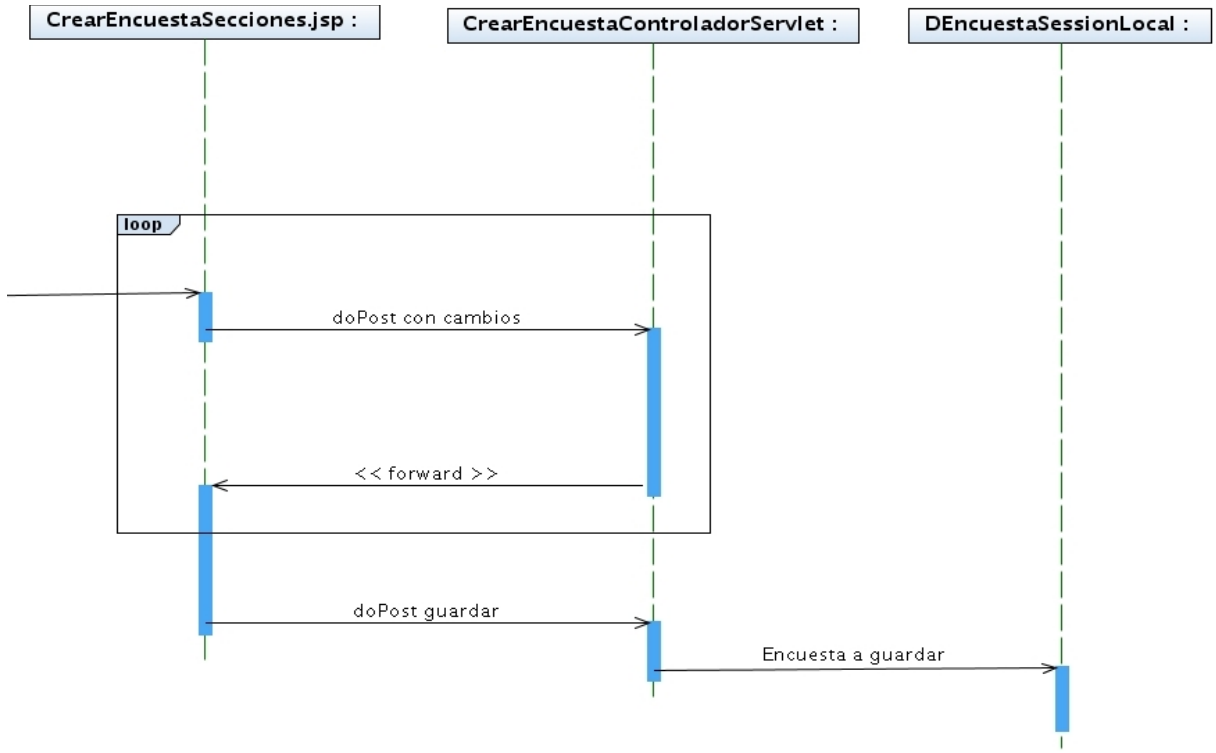
A continuación presentamos dos diagramas de secuencia de dos casos de uso que ilustran las decisiones tomadas en el diseño.

### 5.4.5.1. Diagrama de secuencia de CrearEncuesta

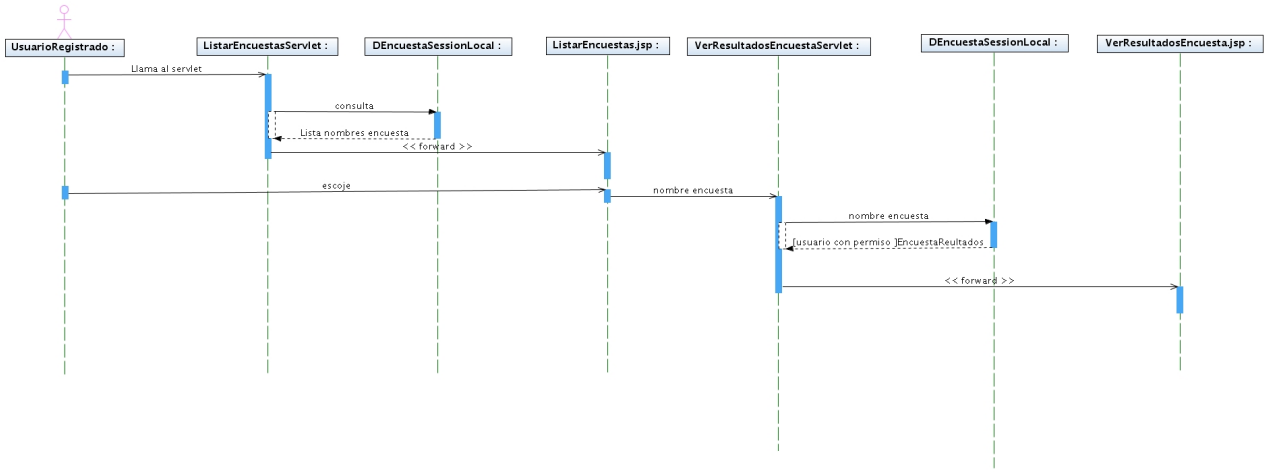



A continuación el mismo diagrama partido en dos para mejor lectura (se ha de entender que la segunda imagen va a la derecha de la primera):

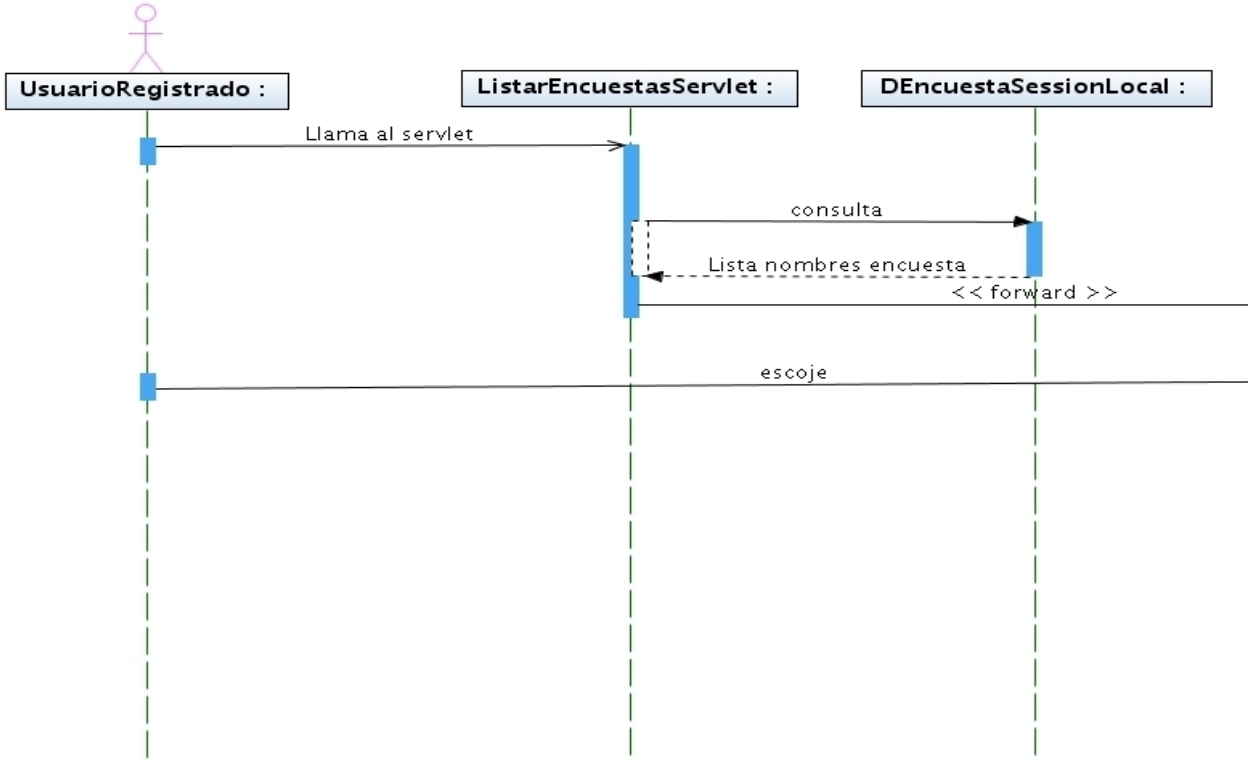

UsuarioRegistrado :
CrearEncuesta.jsp :
CrearEncuestaServlet :
dar_datos
doPost
crear
Encuesta :
<< forward >>

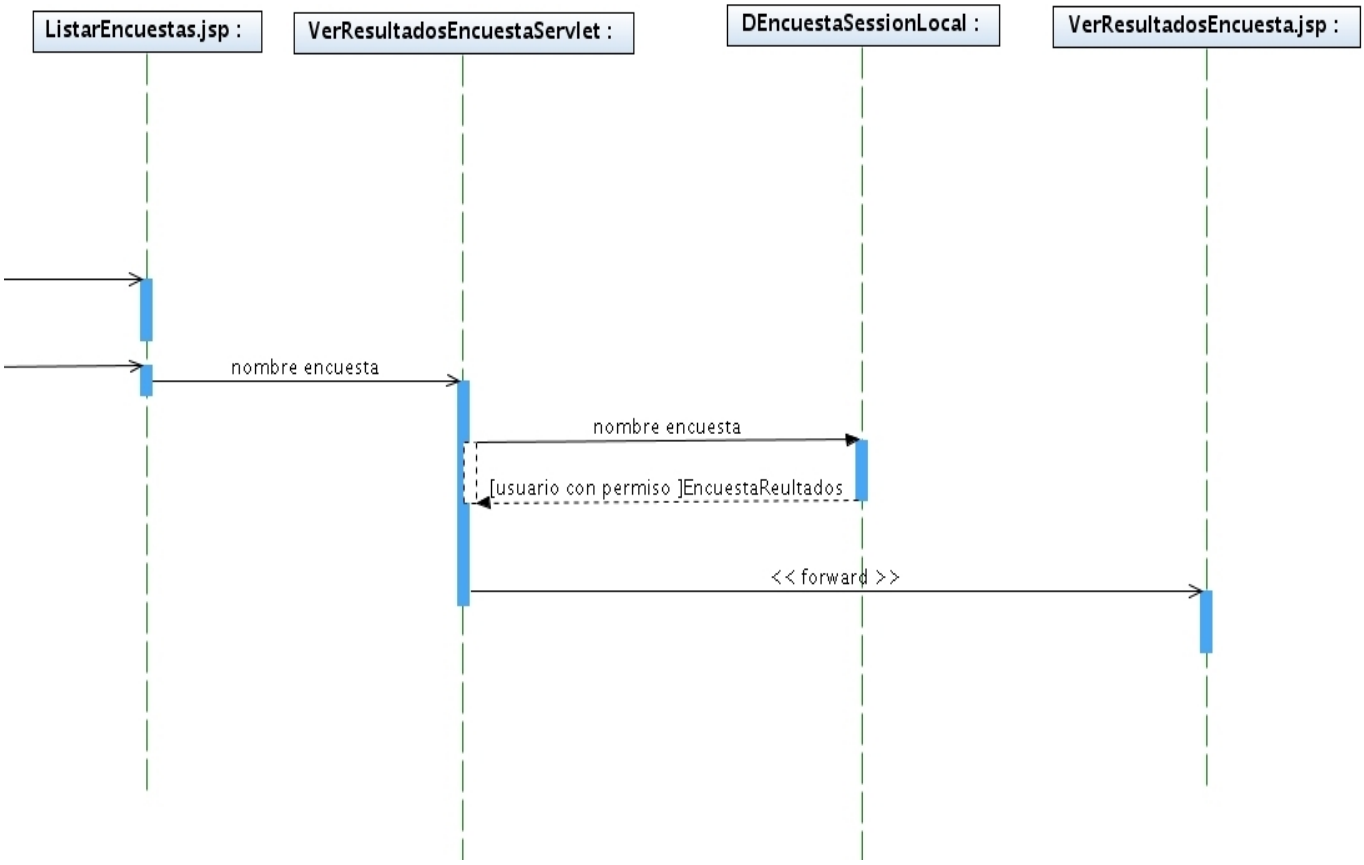

CrearEncuestaSecciones.jsp :
CrearEncuestaControladorServlet :
DEncuestaSessionLocal :
loop
doPost con cambios
<< forward >>
doPost guardar
Encuesta a guardar

### 5.4.5.2. Diagrama de secuencia de verResultadosEncuesta (accediendo a través de ListarEncuesa)




A continuación el mismo diagrama partido en dos para mejor lectura (se ha de entender que la segunda imagen va a la derecha de la primera):


UsuarioRegistrado :
ListarEncuestasServlet :
DEncuestaSessionLocal :
Llama al servlet
consulta
Lista nombres encuesta
<< forward >>
escoje


ListarEncuestas.jsp :
VerResultadosEncuestaServlet :
DEncuestaSessionLocal :
VerResultadosEncuesta.jsp :
nombre encuesta
nombre encuesta
[usuario con permiso ]EncuestaReultados
<< forward >>

## 5.5. Implementación

Después de hacer el análisis y el diseño, llega el momento de implementar el proyecto.

### 5.5.1. Entorno inicial

Todos los componentes usados en la implementación (maquina virtual java, IDE, servidor de aplicaciones,etc...) del proyecto DEncuesta han sido inicialmente instalados en un Debian testing con el kernel 2.6.24.3, donde se ha efectuado todo el desarrollo. Posteriormente en la fase de pruebas el proyecto ha sido probado en un Windows XP. Al tratarse de un proyecto Java la migración de sistema operativo es muy fácil de llevar a cabo.

Para el desarrollo del sistema he decido utilizar el IDE java NetBeans por su gran versatilidad y porque soporta el desarrollo de todos los tipos de aplicación Java, incluyendo Enterprise Java Bean, tecnología sobre la cual se basa el sistema DEncuesta. La version de NetBeans que he usado es la NetBeans IDE 6.1 (Build 200805300101).

Como servidor de aplicaciones J2EE he decido utilizar para el desarrollo Glassfish de SUN que venia con la plataforma java enterprise edition 5 06 (archivo java_app_platform_sdk-5_06-linux.bin). La versión de este Glassfish es v2 UR2. En la fase de pruebas se ha probado con JBOSS versión 5.0.0GA.

En el proyecto a parte de usar las tecnologías JSP y servlets también se hace uso de Enterprise Java Beans. Utilizar Tomcat para los jsp y servlets parece una mala opción ya que tendría que tener funcionando Tomcat y por otra parte un servidor de aplicaciones para los EJBs. Glassfish hace de contenedor tanto para los servlets/jsp como para los EJBs.

Las versión del JDK de Java es 1.6.0_10.

Evidentemente para interactuar con el sistema DEncuesta hace falta un navegador web. (para mas información, ya que algunas características de CSS solo funcionan en un determinada tipo de navegador, el sistema se ha probado que con Firefox)

Los EJB usan persistencia, el servidor de base de datos instalado es Derby que es el que venia con la plataforma java (archivo java_app_platform_sdk-5_06-linux.bin).

### 5.5.2. Ultimas decisiones de diseño

Hasta el momento de la implementación había dejado de lado ciertas cuestiones de diseño del proyecto de lado. Estas son las siguientes:

- Para la persistencia he decidido usar el método CMP, es decir gestionada por el contenedor. He decidido que sea así ya que las ventajas de que la persistencia sea gestionada por el programador (BMP) son casi nulas. Son muy pocos los Beans de entidad y he verificado las tablas creadas automáticamente por el contenedor, y estas satisfacen mis deseos. Programando la tablas personalmente habría acabado haciéndolas igual, y consecuentemente ha sido un ahorro de tiempo. En el punto Persistencia- la base de datos se describen estas tablas.
- Como ya dije en la fase de análisis y diseño, sigo el modelo MVC, la vista siendo programada con jsp. En un intento de separar mas aun las datos de la vista, he decido usar Cascade Style Sheets (CSS). De esta forma he creado un css global para la presentación de los datos html (jsp).
- Todas las colecciones de objetos están implementadas con la clase Java Vector que se encuentra en el paquete java.util. Los arrays en Java son suficientes para guardar tipos básicos de datos, y objetos de una determinada clase cuyo número conocemos de antemano. Algunas veces deseamos guardar objetos en un array pero no sabemos cuantos objetos vamos a guardar. Una solución es la de crear un array cuya dimensión sea más grande que el número de elementos que necesitamos guardar. La clase *Vector* nos proporciona una solución alternativa a este problema. Un vector es similar a un array, la diferencia estriba en que un vector crece automáticamente cuando alcanza la dimensión inicial máxima. Además, proporciona métodos adicionales para añadir, eliminar elementos, e insertar elementos entre otros dos existentes.

### 5.5.3. Estado actual de la implementación

El proyecto Dencuesta esta implementado en el momento actual en un 95%. Esencialmente falta por programar la acción de borrar encuestas.

De momento para borrar encuesta hace falta entrar en la base de datos y borrar las entradas a base de sentencias sql.

Las funciones que puede efectuar el sistema en su estado actual son las siguientes:

- Identificarse en el sistema, dejar de estar identificado y registrarse.
- Listar las encuestas creadas
- Responder una encuesta
- Ver los resultados globales de una encuesta
- Ver los resultados individuales de una encuesta
- Crear encuestas

Las funciones que faltan son:

- Borrar encuestas

Cabe añadir que para implementar esta ultima funcionalidad, el proyecto esta ya preparado para sea fácilmente implementable.

### 5.5.4. Estructura de archivos

Junto a esta memoria viene un archivo comprimido ZIP con todo lo necesario para compilar y desplegar el sistema.

Al descomprimir el archivo tenemos tres carpetas:

- DEncuestaGlassfish: que contiene un archivo EAR para desplegar el sistema en el servidor de aplicaciones Glassfish (versión 2 ur 2)
- DEncuestaJboss : que contiene un archivo EAR para desplegar el sistema en el servidor de aplicaciones Jboss 5.0.0.GA

(mas adelante hay una explicación de la diferencia entre estos archivos EAR.)

- sources: que contiene una carpeta llamada DEncuesta.

A continuación suponemos que estamos dentro de la carpeta DEncuesta anteriormente mencionada y describimos su contenido:

Estas son las carpetas:

- build: aquí es donde se han empaquetado los archivos DEncuesta-ejb.jar (con todos los EJBs y demás clases compìlados) y DEncuesta-war.war (con todos los servlets compilados y los JSP así como los CSS)
- Dencuesta-ejb: aquí están todos los archivos necesarios de los EJBs (las fuentes, las compilaciones y archivos de configuración). El paquete DEncuesta-ejb.jar mencionado anteriormente contiene los archivos "útiles" de esta carpeta. El contenido de las subcarpetas son:
  - build: todas los EJB y demas clases compilados
  - dist: el paquete Dencuesta-ejb.jar antes mencionado
  - nbproject: archivos de configuración
  - src: todas las fuentes

- DEncuesta-war: aquí están todos los archivos necesarios de los servlets y JSPs (las fuentes, las compilaciones y archivos de configuración). El paquete DEncuesta-war.war mencionado anteriormente contiene los archivos "útiles" de esta carpeta. El contenido de las subcarpetas son:
  - build: todos los JSPs, CSS y imagenes de la web
  - dist: el paquete Dencuesta-war.war antes mencionado
  - nbproject: archivos de configuración
  - src: todas las fuentes de los servlets

- dist: aquí esta el archivo EAR (es decir todo lo necesario para hacer funcionar la aplicación.) Viene a ser los archivos Dencuesta-ejb.jar y Dencuesta-war.war empaquetados.
- Nbproject: contiene archivos de configuración

### 5.5.5. Persistencia - la base de datos

Como ya hemos mencionado la persistencis es CMP.

Estas son las tablas que crea el contenedor Glassfish:

Tabla Usuarios:

Tabla Encuestas:

Esta tabla sirve para guardar los objetos Encuesta así como EncuestaRespondida. El campo DTYPE determina de que tipo es cada entrada en la tabla.

### 5.5.6. Fase de pruebas

La fase de pruebas se desarrollo como es habitual, con errores y soluciones. Sin embargo quiero que halla constancia de un error que quizás se encuentra el lector al probar el programa.

Al desplegar varias veces seguidas el programa, a veces, deja de funcionar. El servidor de aplicaciones detecta mal los métodos de los objetos y de vez en cuando salta una excepción “método no hallado”. Basta con volver a desplegar el archivo EAR y acaba volviendo a la normalidad.

Por otro lado cabe comentar que existen dos servlets, que se usaron durante la fase de pruebas, que crean encuestas de manera estática. Los he dejado en el proyecto para hacer pruebas en un futuro (si por un momento el modulo de crear encuestas no funciona, siempre se pueden crear encuestas de esta forma).

### 5.5.7. Futuro del proyecto; cambios y mejoras

Evidentemente en un futuro lo primero que hay que hacer es implementar la acción de eleminar encuestas. Un primer usuario administrador se tiene que añadir de forma estatica, despues este podra establecer usuarios como administradores.

Actualmente las encuesta tienen dos tipos de respuestas, radio button y check box. En futuras versiones se pretende añadir mas tipos; la inserción de imagenes y video en las preguntas tambien es una mejora que es implementable.

Gracias al patron MVC podemos reutilizar parte del codigo, en este caso el modelo, y redefinir completamente la vista (con el controlador). En vez de usar html (dinamicamente creado con jsp) podemos en versiones posteriores utilizar applets. Los applets permitirian un mayor dinamismo que el html.

### 5.5.8. Archivos de despliegue

He creado dos archivos listos para desplegarse; uno para Glassfish v2ur2 y otro para JBoss v5.0.0.GA .

Estos dos archivos EAR son idénticos a excepción del persistence.xml ya que cada servidor de aplicaciones usa su propia base de datos (y otros parámetros).

Este es el contenido para cada servidor de aplicaciones:

Glassfish

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0"
xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
  <persistence-unit name="DEncuesta-ejbPU" transaction-type="JTA">
    <provider>oracle.toplink.essentials.PersistenceProvider</provider>
    <jta-data-source>jdbc/sample</jta-data-source>
    <properties>
      <property name="toplink.ddl-generation" value="create-tables"/>
    </properties>
  </persistence-unit>
</persistence>
```

JBoss

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0"
xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
  <persistence-unit name="DEncuesta-ejbPU" transaction-type="JTA">
```

```
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
     <jta-data-source>java:/DefaultDS</jta-data-source>
     <properties>
            <property name="hibernate.dialect" value=
"org.hibernate.dialect.HSQLDialect"/>
       <property name="hibernate.hbm2ddl.auto" value="create-drop"/>
     </properties>
  </persistence-unit>
</persistence>
```

Para otros servidores de aplicaciones es necesario modificar este archivo.

## 5.6. Conclusiones

La realización de este proyecto me ha sido de gran utilidad. He revisado conocimientos de toda la carrera y me he introducido en el mundo JEE.

Respecto al Java Enterprise Edition podemos decir que:

- JEE conlleva un desarrollo complejo pero con muy buenas características.
- Se necesita tiempo para adaptarse a su “filosofía” y para hacer las pruebas necesarias que permiten comprender bien como funciona.
- Elegir que herramientas y tecnologías de JEE queremos usar no es una tarea fácil. A mitad del proyecto estuve apunto de rehacerlo desde cero para utilizar Struts.

Durante el desarrollo del proyecto he tenido muchas otras labores que no me han permitido dedicarme al 100% en el TFC sin embargo estoy satisfecho del producto final obtenido. La excepción quizás ha sido no haber tenido mas tiempo para el diseño, como ya he mencionado antes, la idea de usar struts me hubiese gustado.

Por ultimo realizar esta memoria ha sido interesante, sobre todo porque en un futuro me gustaría dedicarme a la investigación.

# 6. Glosario

DEncuesta: Nombre de este proyecto

Encuesta: Una encuesta es un conjunto de preguntas normalizadas dirigidas a una muestra representativa de la población o instituciones, con el fin de conocer estados de opinión o hechos específicos.

TFC: Trabajo de Fin de Carrera

ETIS: Acrónimo de Ingeniera Técnica de Informática de Sistemas

PAC: Acrónimo de Prueba de Evaluación Continuada (Proba de Avaluació Continuada)

JEE: Java Enterprise Edition (ver 5.2.1.)

JSP: Java Server Pages (ver 5.2.4.)

Servlet: (ver 5.2.3)

EJB: Enterprise Java Bean (ver 5.2.2.)

JSF: Java Server Face es un framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa JavaServer Pages como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías como XUL.

Struts: es una herramienta de soporte para el desarrollo de aplicaciones bajo el patrón MVC bajo la plataforma J2EE.

Glassfish: es un servidor de aplicaciones que implementa las tecnologías definidas en la plataforma JEE y permite ejecutar aplicaciones que siguen esta especificación. Es gratuito y de código libre.

JBoss: es un servidor de aplicaciones JEE de código abierto implementado en Java puro. Al estar basado en Java, JBoss puede ser utilizado en cualquier sistema operativo que lo soporte.

CMP: Container Managed Persistence (ver 5.4.4.)

BMP: Bean Managed Persistencce (ver 5.4.4.)

MVC: Acronimo de Modelo Vista Contenedor. Es un patrón de diseño (ver 5.4.2.)

HTML: siglas de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de paginas web.

CSS: Acrónimo Cascade Style Sheet, es un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML .

OO: Orientado a Objetos

BD: Base de Datos

# 7. Bibliografía

Para el análisis y el diseño me he basado en gran parte en los apuntes de la materia “Engenyeria del Programari” de la carrera de ETIS.

Para la memoria he consultado las memorias de TFC de alumnos que hay en la biblioteca de la UOC.

El resto de información la he encontrado en internet.

Estas son la direcciones:


**GENERAL**

Wikipedia, la enciclopedia libre

http://es.wikipedia.org

Programación en castellano

http://www.programacion.net

**JEE (EJB, servlets, JSP, JSF, etc..) y patrones**

An introduction to the Java EE Platform

http://java.sun.com/javaee/5/docs/firstcup/doc/p2.html#gcqyb

http://www.programacion.net/java/tutoriales/J2EE

Integración de EJB3 y JSF

http://www.jroller.com/jl_monteagudo/entry/integraci_oacute_n_de_ejb3

JSP y EJB – Foros del Web

http://www.forosdelweb.com/f67/jsp-ejb-426756/

Servlets Essentials

http://www.novocode.com/doc/servlet-essentials/chapter2a.html

Tutorial: EJB con Netbeans 5.5

http://diesil-java.blogspot.com/2007/11/tutorial-ejb-30-enterprise-beans-con.html

Introduction to the Struts Web Framework

http://www.netbeans.org/kb/60/web/quickstart-webapps-struts.html

Aplicación paso a paso con Struts

http://www.monografias.com/trabajos28/aplicacion-paso-paso-struts/aplicacion-paso-paso-struts.shtml

JSP tutorial

http://www.jsptut.com/Sessions.jsp

MVC con servlets y JSPs

http://ribesoftware.wikispaces.com/MVC+con+servlets+y+JSPs?f=print

JSP vs Struts vs JSF vs .....

http://www.velocityreviews.com/forums/t149027-jsp-vs-struts-vs-jsf-vs-swing-vs-tapestry-vs-spring-vs-rails.html

Transactions in Web components

http://java.sun.com/javaee/5/docs/tutorial/doc/bncjg.html

Getting started with EJB

http://www.conceptgo.com/gsejb/eb04.html

**CSS**

Creating a CSS layout from scratch

http://www.subcide.com/tutorials/csslayout/index.aspx

CSS tutorial

http://www.w3schools.com/css

**HTML**

HTML tutorial

http://www.w3schools.com/html/DEFAULT.asp

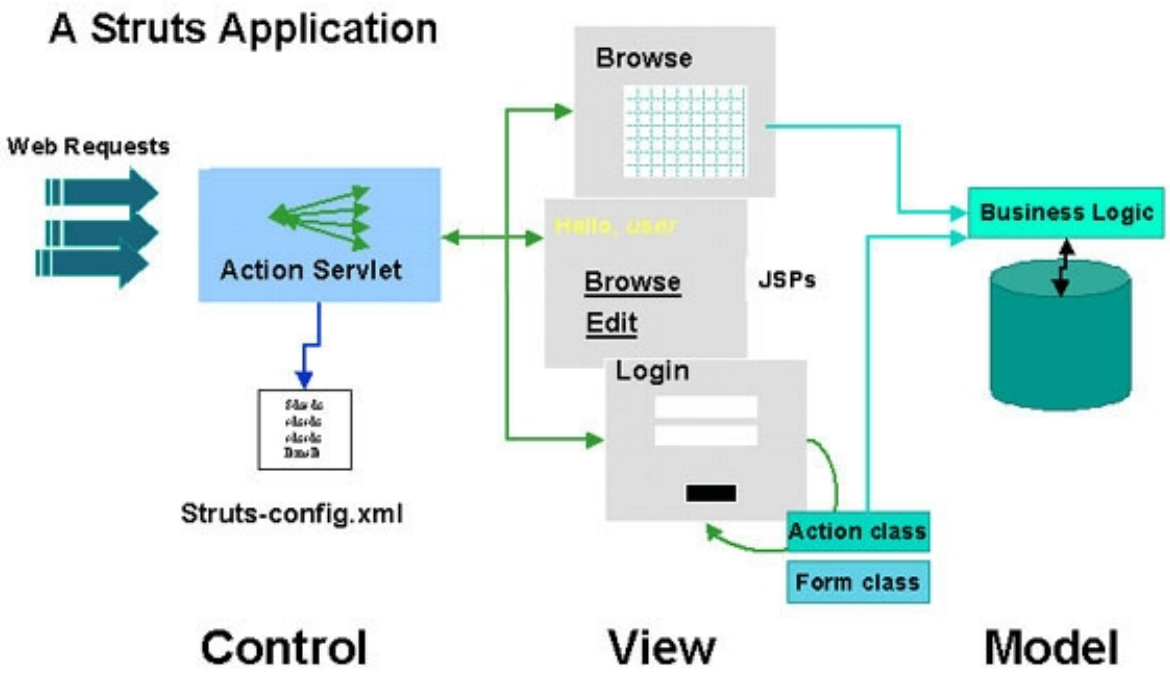
# 8. Anexo

Ya que he dudado de si usar Struts, pongo en este apartado una breve descripción de este. (Fuente Wikipedia)

**Struts** es una herramienta de soporte para el desarrollo de aplicaciones bajo el patrón MVC bajo la plataforma J2EE (Java 2, Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts.

Struts permite reducir el tiempo de desarrollo. Su carácter de software libre y su compatibilidad con todas las plataformas en que Java Entreprise esté disponible, lo convierte en una herramienta altamente disponible.

Struts se basa en el patron del Modelo-Vista-Controlador (MVC) el cual se utiliza ampliamente y es considerado de gran solidez. De acuerdo con este patrón, el procesamiento se separa en tres secciones diferenciadas, llamadas el modelo, las vistas y el controlador.

Cuando se programan aplicaciones Web con el patrón MVC, siempre surge la duda de usar un solo controlador o usar varios controladores, pues si consideramos mejor usar un solo controlador para tener toda nuestra lógica en un mismo lugar, nos encontramos con un grave problema, ya que nuestro controlador se convierte en lo que se conoce como "fat controller", es decir un controlador de peticiones, Struts surge como la solución a este problema ya que implementa un solo controlador (ActionServlet) que evalúa las peticiones del usuario mediante un archivo configurable (struts-config.xml).

Este diagrama termina de explicar el funcionamiento: