

UØCsurveys

Aplicación web para la realización de cuestionarios

Marcos Roig Ramón
Ingeniería Técnica en Informática de Sistemas

Consultor: Ferrán Prados Carrasco

Curso: 2013/2014-2

08 de Junio de 2014

“Le dedico el presente trabajo a Nat, por su apoyo incondicional que me ha dado durante toda la carrera y especialmente estos últimos meses.”

Tabla de contenido

1. Introducción.....	3
1.1. Justificación del TFC.....	3
1.2. Objetivos.....	3
1.3. Enfoque y método a seguir.....	4
1.4. Planificación.....	5
1.4.1. Hitos principales.....	5
1.4.2. Planificación temporal y diagrama de Gantt.....	5
1.5. Productos obtenidos.....	7
1.6. Descripción de resto de capítulos de la memoria.....	9
2. Especificación y análisis.....	10
2.1. Actores.....	10
2.2. Casos de uso y diagramas de casos de uso.....	11
2.2.1. Casos de uso comunes a los diferentes actores.....	12
2.2.2. Actor Consultor/Profesor.....	13
2.2.3. Actor Estudiante.....	18
2.2.4. Actor Administrador.....	19
2.3. Requisitos no funcionales.....	20
2.3.1. Rendimiento.....	20
2.3.2. Usabilidad.....	21
2.3.3. Accesibilidad.....	21
2.3.4. Seguridad.....	21
2.3.5. Requisitos tecnológicos.....	21
3. Diseño.....	22
3.1. Análisis de las principales tecnologías.....	22
3.1.1. Justificación de la elección de tecnologías.....	22
3.1.2. Plataforma J2EE (Java EE).....	23
3.1.3. Arquitectura de la aplicación.....	24
3.1.4. Patrones de diseño.....	26
3.1.4.1. MVC.....	26
3.1.4.2. DAO.....	27

3.1.5. Frameworks.....	28
3.1.5.1. Struts 2.....	28
3.1.5.2. Hibernate.....	30
3.1.5.3. Java Server Pages (JSP).....	30
3.2. Diseño de clases.....	31
3.3. Diseño de la base de datos.....	32
3.4. Modelización de la interfaz.....	33
3.4.1. Mapa navegacional.....	33
3.4.2. Prototipo de la interface gráfica de usuario.....	35
4. Ampliación y propuestas de mejora.....	37
5. Valoración económica.....	38
6. Conclusiones.....	40
7. Bibliografía y referencias.....	41
ANEXOS.....	43
A. Acceso a la aplicación.....	43
B. Capturas de pantalla de la aplicación.....	44

1. Introducción.

Mediante el presente documento se pretende explicar el desarrollo del trabajo final de carrera. Ámbito, objetivos y método de trabajo seguido durante el último semestre para su desarrollo. En este hemos abordado el desarrollo de una aplicación web (**UØCsurveys**) para la realización de cuestionarios (test, encuestas,...) en un entorno de trabajo colaborativo en un ámbito docente como es el de la UOC.

En este contexto y mediante este documento se pretende hacer la presentación del alcance del proyecto, enumerando los requerimientos y funcionalidades de la aplicación web, así como la distribución de las tareas y planificación temporal que se ha seguido a lo largo del semestre.

1.1. Justificación del TFC.

El TFC que se presenta consistirá en el desarrollo de una aplicación Web que facilite la generación de diferentes tipos de cuestionarios por parte de los consultores. Estos cuestionarios podrán ser de diferente tipo (encuestas, test, exámenes,...) y a partir de ellos, una vez cumplimentados, se podrán obtener diferentes informes y estadísticas relativas a la información recogida mediante los mismos que permitan realizar un seguimiento del proceso de aprendizaje de los estudiantes en unos casos u obtener información relativa a la asignatura mediante encuestas en otros.

Asimismo, con el presente trabajo se pretende poner en práctica todos los conocimientos adquiridos a lo largo de la carrera de diferentes asignaturas, como Programación Orientada a Objetos, Bases de Datos, Ingeniería del Software,... aplicándolos a un problema concreto y transformándolo en un proyecto informático.

1.2. Objetivos.

El objetivo principal de este TFC ha sido profundizar en el uso de la tecnología Java y adquirir conocimientos sobre la arquitectura J2EE así como adquirir experiencia en las técnicas de desarrollo de software, desde las fases iniciales de identificación de los requisitos hasta la entrega e implantación del producto final. Durante el desarrollo se han trabajado diferentes áreas que implicaban el estudio de nuevas herramientas y recursos o profundizar en los conocimientos ya adquiridos de las mismas.

Algunos de los aspectos que se han tratado durante el desarrollo del proyecto son entre otros:

- Estudio de los **frameworks** y estándares disponibles (Struts 2, Hibernate,...) para las diferentes capas de la aplicación, que facilitan el desarrollo de este tipo de aplicaciones desarrollados por la comunidad y distribuidos bajo licencia libre.
- Modelización de la interface web primando la usabilidad de la misma.

- Utilización de mecanismos de seguridad y de autenticación de usuarios.
- Uso de patrones de diseño MVC.
- Uso del servidor de aplicaciones Apache TOMCAT, para gestionar servlets, JSP, ...
- Uso y configuración del entorno de desarrollo Eclipse .
- Uso de GIT como sistema de control de versiones integrado en Eclipse.
- Uso de un SGBD relacional como MySQL.

1.3.Enfoque y método a seguir.

El proyecto desarrollado en el TFC no pretende implementar todas las funcionalidades que serían propias de una aplicación Web con estas características.

En este proyecto hemos aplicado el ciclo de vida clásico o en cascada que nos ha permitido dividir el proyecto en varias etapas además de permitirnos una compaginación clara con el calendario de entregas previsto a lo largo del semestre de las distintas partes del desarrollo del TFC.

Siguiendo este modelo clásico, el proyecto pasaría progresivamente por las siguientes etapas:

- **Planificación.** En esta tratamos de planificar cada una de las fases en la que se dividirá el proyecto y el tiempo estimado que tendremos que emplear en cada una de ellas.
- **Análisis de Requisitos.** Define los requerimientos específicos del proyecto y como se producirá la interacción de los diferentes actores con el producto final.
- **Diseño de la arquitectura.** Define los requisitos de diseño de la arquitectura que tendrá la aplicación, tanto desde un punto de vista interno como externo.
- **Implementación.** Es esta etapa es donde se ha desarrollado la aplicación. Se escribe el código y se genera el producto ejecutable así como la documentación complementaria necesaria.
- **Pruebas.** Se verifica que el producto desarrollado se corresponda con los requisitos y se depura la aplicación de posibles errores en etapas anteriores.
- **Memoria y presentación:** Se realiza la presentación del proyecto con toda la información recogida en la memoria junto con el archivo que contiene el producto final concreto.

En todo momento durante el desarrollo del TFC, y para mejorar la tolerancia a los cambios y la adaptabilidad de estas etapas, se establecieron ciclos de retroalimentación en el análisis final de cada etapa. Esto permitió una retroalimentación entre etapas para poder revisar los requisitos e incorporar las correcciones de los errores que íbamos encontrando.

Se pretendió también, desde el comienzo del TFC ir avanzado en el estudio de la arquitectura de aplicaciones Web basadas en JSP y del software necesario para su ejecución, realizando una breve investigación de las opciones disponibles y realizando algunas pruebas de implementación para evaluarlas. Las principales tecnologías utilizadas en el proyecto han sido, en concreto Java Server Pages, Struts 2, Apache Tomcat, Hibernate,... de las cuales se buscó material bibliográfico que permitiera el estudio de las mismas para su aplicación el proyecto.

1.4. Planificación.

La elaboración de este trabajo se dividió en 4 etapas principales que daban como resultado un documento a entregar por cada una de ellas. La fecha de comienzo del proyecto se estableció en el día de inicio del semestre, mientras que la fecha límite de la entrega de esta memoria es el día 8 de Junio de 2014. Durante el desarrollo del proyecto se han realizado una serie de entregas parciales que se han correspondido a uno o varios hitos fundamentales del ciclo de desarrollo de software habitual.

1.4.1. Hitos principales.

	Descripción
Plan de trabajo.	Justificación, objetivos del proyecto y planificación temporal del mismo
Especificación y análisis	Especificación y análisis
Diseño	Diseño de la arquitectura de la aplicación
Codificación, memoria y presentación virtual	Entrega de la memoria del proyecto, presentación del mismo y del producto final.

1.4.2. Planificación temporal y diagrama de Gantt.

La gráfico 1 muestra un Diagrama de Gantt en el que aparecen reflejadas las diferentes etapas que se previeron para el desarrollo del TFC. En el diagrama aparecen las tareas ordenadas por etapas y se remarcan especialmente las fechas de entrega oficiales.

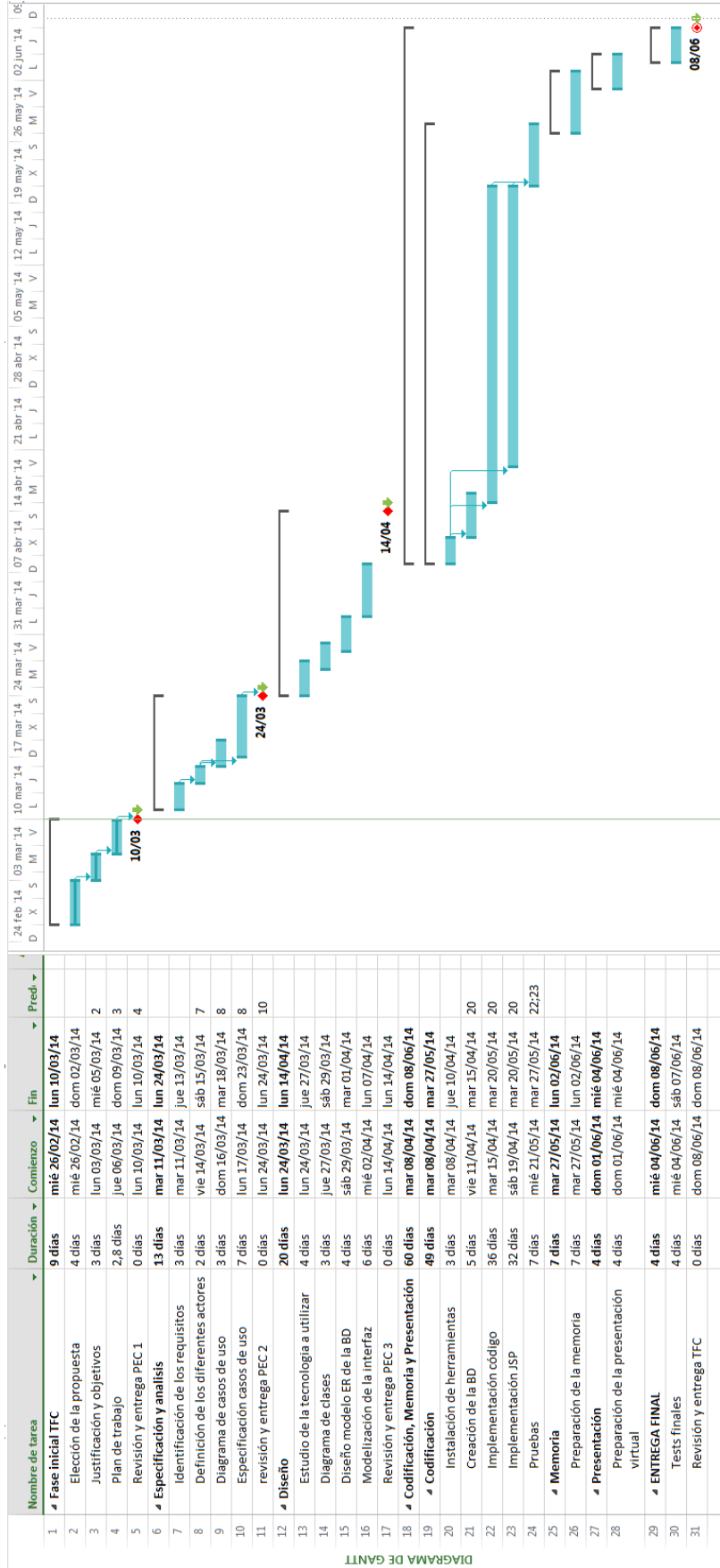


Figura 1. Diagrama de Gantt plan de trabajo TFC.

Para realizar la planificación temporal hemos utilizado Microsoft Project 2013. En el calendario que se adoptó para el desarrollo del proyecto, se habían considerado laborables todos los días de la semana siendo la jornada de trabajo de tres horas.

Las estimaciones iniciales que se habían considerado no se han podido aplicar a lo largo de todo el proyecto. En las primeras fases el seguimiento sí que fue el esperado y se pudo cumplir con todos los objetivos que se habían previsto, pero una vez iniciada la fase de implementación el seguimiento de este plan ha sido escaso. El motivo que ha provocado esto ha sido otro plantan de trabajo, de mi empresa actual, de la que soy un recurso más y que ha privado de la mayor parte del tiempo que esperaba dedicarle a la etapa de codificación, memoria y presentación.

En mi rol de director del proyecto, hubiera tomado la decisión de asignar estas tareas a otro programador para así poder cumplir los plazos de desarrollo, pero dado que todos los roles de los recursos implicados en el desarrollo los asumía yo esto no ha sido posible.

La consecuencia final, ha sido que no se han desarrollado todas las funcionalidades previstas, que posteriormente detallaremos, para la fecha establecida de entrega final. Se han tenido que dejar algunas de ellas para una futura implementación que se llevaría a cabo en las próximamente para cumplir todos los requisitos previstos para la aplicación.

1.5.Productos obtenidos.

El producto obtenido con este proyecto ha sido una aplicación web accesible para tres roles distintos de usuarios para los que existen una serie de funcionalidades que les permiten entre otras cosas:

- **Acceso al sistema.**
Permite a los diferentes usuarios identificarse para acceder a las distintas partes del sistema cada una de ellas específica para cada tipo de usuario autenticado.
- **Realizar Altas de diferentes tipos de usuarios.**
Permite dar de alta en el sistema a consultores y estudiantes.
- **Modificar datos usuario.**
Permite que los usuarios puedan modificar sus datos personales que están almacenados en la BD del sistema.
- **Realizar búsquedas de usuarios.**
Realizar búsquedas de usuarios en la BD del sistema. Se ha implementado la búsqueda de usuario mediante su "id" aunque sería fácilmente ampliable para realizar búsquedas por NIF o por cualquier otro criterio que se considere oportuno.

- **Obtener acceso a herramienta para la gestión de la Base de Datos (BD).**
Opción que enlaza con la BD del sistema y que permite realizar una gestión directa de la misma mediante PhpMyAdmin.
- **Opciones de cambio de contraseñas.**
Página que permite que los usuarios cambien su contraseña de acceso al sistema.
- **Listar usuarios.**
Página que permite obtener un listado de los usuarios del sistema (administrador) o de los estudiantes (consultor). A partir de este listado se tiene acceso a los datos de los usuarios.
- **Crear Tema.**
Página que permite dar de alta un Tema que sirve de base para posteriormente organizar las preguntas.
- **Crear Pregunta.**
Página que permite crear nuevas preguntas que posteriormente podrán utilizarse en los cuestionarios.
- **Crear Opción.**
Permite introducir las respuestas de las distintas preguntas creadas. Estas opciones se marcaran como cierta o falsa lo que permitirá las tareas relacionadas con la corrección de cuestionarios de tipo examen o test.
- **Crear cuestionario.**
Página relativa a la creación de cuestionarios. Desde ella se podrán añadir/eliminar las preguntas que se desee al cuestionario. Una vez se considere que el cuestionario está finalizado mediante el botón correspondiente se accederá a la página para completar los datos como título y descripción del cuestionario para posteriormente almacenarlo en la BD del sistema.
- **Listar cuestionarios.**
Página que permite obtener un listado de los cuestionarios del sistema para el consultor activo. A partir de este listado se tiene acceso a los datos del cuestionario.
- **Edición de datos (preguntas, temas,...)**
En algunas de las páginas esta activa la posibilidad de modificar los datos, de las preguntas o temas entre otros, con el correspondiente botón para guardar los cambios.
- **Eliminar Tema.**
Opción que permite eliminar un tema.

- **Gestión de Preguntas.**
Al acceder al área de gestión de preguntas aparece un listado de los temas disponibles con las preguntas que tiene cada uno de ellos. Desde este listado se accede a cada uno de los temas con su lista respectiva de preguntas. También existe un botón para crear nuevos temas.
- **Gestión de Estudiantes.**
Al acceder al área de gestión de estudiantes aparece un listado de los estudiantes desde la que se puede acceder a los datos de cada uno de ellos como se ha indicado anteriormente.
En esta área existe un menú con diferentes opciones que permiten nuevas funcionalidades que se están desarrollando.
- **Gestión de Cuestionarios.**
Área de gestión de cuestionarios en la que aparece un listado con los cuestionarios y un menú con diferentes opciones.

Todos los productos obtenidos que se han detallado y algunos otros se basan en los casos de Uso que se expondrán en los próximos capítulos de la presente memoria.

1.6.Descripción de resto de capítulos de la memoria.

En los siguientes capítulos se realiza una exposición más detallada de las diferentes etapas que se han seguido en el desarrollo del Trabajo Final de Carrera.

- **Capítulo 2 Especificación y análisis:**
En él se definen los diferentes actores que se han considerado que interactuarán con el sitio Web. Se han definido los casos de uso y se ha realizado el diagrama correspondiente. También se establecen en él los requisitos no funcionales.
- **Capítulo 3 Diseño:**
En él se realiza un análisis de las principales tecnologías que se han utilizado en el desarrollo de proyecto.
- **Capítulo 4 ampliaciones y propuestas de mejora:**
Exponemos las posibles ampliaciones futuras de la aplicación y opciones de mejora, tanto con la inclusión de nuevas funcionalidades o mejoras en la interfaz gráfica.
- **Capítulo 5 Valoración económica.**
- **Capítulo 6 Conclusiones.**
- **Capítulo 7 Bibliografía.**
- **Anexos**

2. Especificación y análisis.

En este apartado se ha definido el comportamiento que debe tener la aplicación, definiendo los requerimientos funcionales de la misma. Esta es una etapa importante en el desarrollo de software y con ella se pretende definir de forma precisa el problema a resolver.

Hemos utilizado los casos de uso como medio para especificar los requisitos funcionales y junto con UML, lenguaje estándar que permite crear modelos visuales del software (actores, relaciones entre actores, diagramas de casos de uso,...), hacer la documentación de estos.

2.1. Actores.

Los actores son las entidades externas (usuarios) que interactúan con el sistema. En este caso distinguiremos tres tipos de actores, cada uno de los cuales podrá llevar a cabo una serie de actividades, algunas de ellas comunes para varios de los actores y otras exclusivas que sólo podrá desempeñar un actor específico correctamente identificado. Todos los actores considerados deberán autenticarse para poder acceder e interactuar con el sistema.

Para definir a los actores se ha tenido en cuenta el organigrama de usuarios que podemos identificar cuando accedemos al campus virtual de la UOC.

- **Estudiante:** Es un alumno de la UOC. A ellos estarán destinados los distintos formularios que los Consultores generen. Cuando entre en la página del sistema tendrá accesibles una serie de opciones como consultar sus datos personales, responder los formularios, obtener diferentes informes relativos a los cuestionarios que haya realizado.
- **Consultor:** Son los encargados de la creación de los formularios (test autoevaluación, encuestas, Exámenes,...). Podrán consultar los listados de estudiantes, obtener información detallada de estos, obtener diferentes informes estadísticos relativos a los diferentes formularios generados que ya hayan sido completados por los estudiantes.
- **Administrador:** Es el encargado de llevar a cabo las tareas administrativas del sitio web. Tiene el control total del sistema y una de sus funciones principales es la administración de usuarios. Muchas de sus funciones estarán accesibles a través del sitio Web a las cuales podrá acceder después de autenticarse con las credenciales administrativas.

También tiene acceso a los servidores y realiza la gestión directa de la base de datos utilizando diferentes herramientas.

2.2.Casos de uso y diagramas de casos de uso.

En este apartado se muestran los diagramas de los casos de uso junto con la descripción textual de los mismos para los tres perfiles previstos de interacción con el sitio Web.

Para facilitar la visualización del diagrama de casos de uso se ha añadido a este un nuevo actor al que hemos denominado “usuario” que engloba los casos de uso comunes a los diferentes actores que interactúan con el sistema.

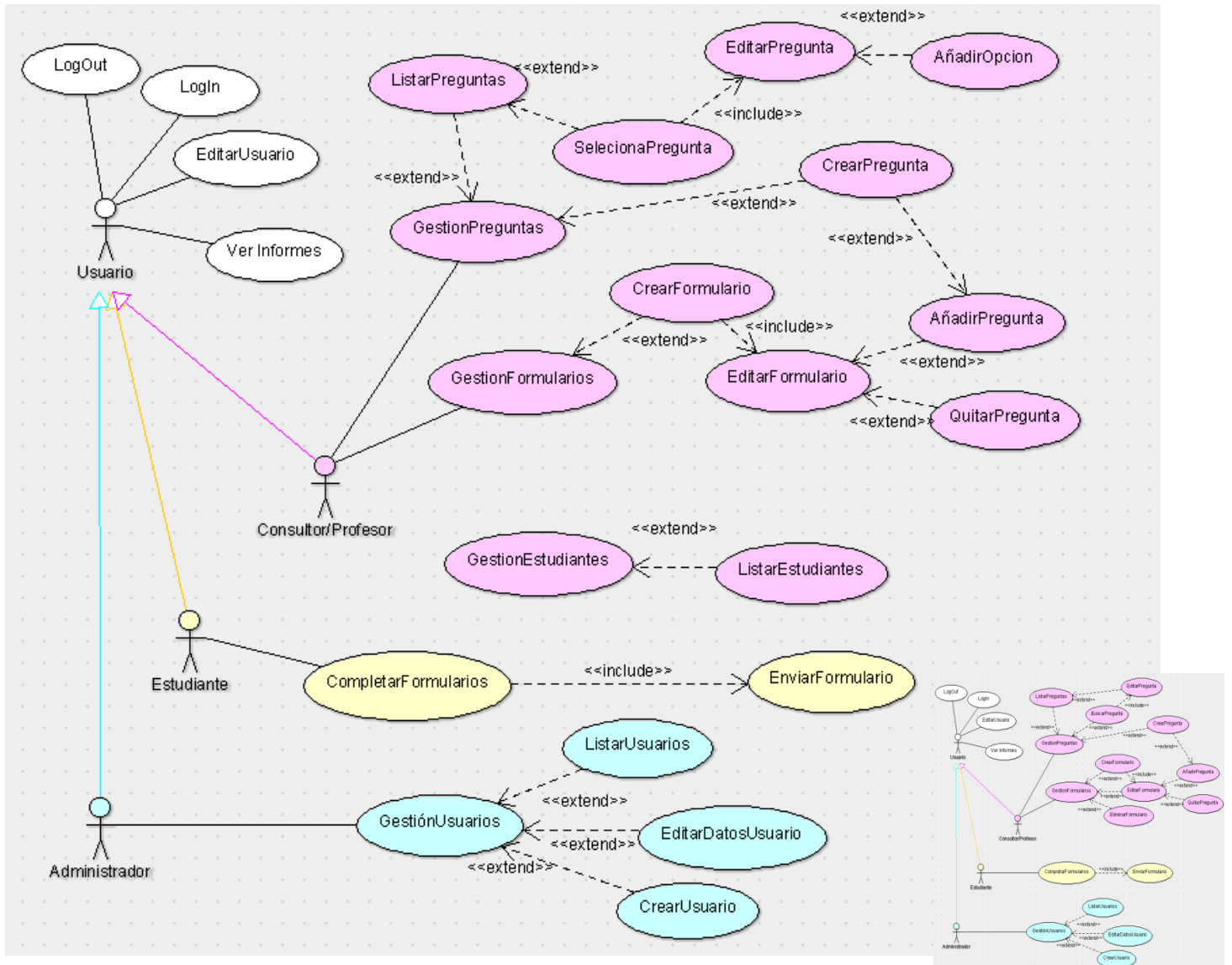


Diagrama anterior

Figura 2. Diagrama de casos de uso.

2.2.1. Casos de uso comunes a los diferentes actores.

Son los casos que en la figura 1 se han asociado al actor “usuario”.

Caso de Uso: Login	
Resumen de la funcionalidad	Entrar en el sistema como usuario registrado para poder utilizar las funcionalidades disponibles para el actor cuyas credenciales se han utilizado para acceder al sistema.
Actores	Estudiante, Consultor/Profesor y Administrador.
Casos de uso relacionados	<u>Ninguno</u>
Precondición	El usuario existe en la base de datos de usuarios registrados del sistema.
Postcondición	El sistema valida el acceso del usuario.
Flujo normal	
<ol style="list-style-type: none"> 1. El usuario introduce sus datos de acceso. 2. La aplicación verifica los datos y se le da acceso al sistema como Estudiante, Consultor/Profesor o Administrador. 	
Flujo alternativo	
2.1. Los datos introducidos son incorrectos por lo que el sistema muestra un error.	

Caso de Uso: LogOut	
Resumen de la funcionalidad	Finalizar la sesión de un usuario.
Actores	Estudiante, Consultor/Profesor y Administrador.
Casos de uso relacionados	<u>Ninguno.</u>
Precondición	El usuario ha iniciado sesión con sus datos de acceso.
Postcondición	El sistema cierra la sesión del usuario.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El usuario selecciona la opción para cerrar sesión. 2. El sistema finaliza la sesión del usuario. 	
Flujo alternativo	
Ninguno.	

Caso de Uso: Editar Usuario	
Resumen de la funcionalidad	Modificar los datos personales.
Actores	Estudiante, Consultor/Profesor y Administrador.
Casos de uso relacionados	<u>Ninguno.</u>
Precondición	El usuario ha iniciado sesión.
Postcondición	Se muestran los datos personales del usuario para que los pueda modificar.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El usuario selecciona la opción “<u>Editar Usuario</u>”. 2. Se muestran sus datos para que los modifique. 	
Flujo alternativo	
Ninguno.	

Pendiente de implementar al 100%

Caso de Uso: Ver Informes	
Resumen de la funcionalidad	Se encarga de mostrar a los distintos usuarios diferentes estadísticas. Estas estadísticas o resultados serán distintos según sean las credenciales del actor que ejecute este caso de uso.
Actores	Consultor/Profesor, Estudiantes, Administrador.
Casos de uso relacionados	<u>Pendientes de definir.</u>
Precondición	Que existan resultados o estadísticas disponibles.
Postcondición	El usuario obtiene información o estadísticas según lo solicitado por este en base a las opciones disponibles para cada actor.
Flujo normal principal	
1. El usuario una opción para obtener resultados sobre formularios ya completados y dados como finalizados.	
Flujo alternativo	
Ninguno.	

2.2.2. Actor Consultor/Profesor.

Este perfil se corresponde al del usuario registrado que ha realizado "LogIn" en la aplicación. A este perfil se accede únicamente tras una correcta identificación del usuario.

A continuación se detallan los diferentes casos de uso para este actor.

Caso de Uso: Gestión Preguntas	
Resumen de la funcionalidad	Se encarga de mostrar al Consultor/Profesor la interfaz para la gestión de las preguntas.
Actores	Consultor/Profesor.
Casos de uso relacionados	<u>Listar Preguntas, Buscar Pregunta, Crear Pregunta.</u>
Precondición	El usuario se ha identificado en el sistema como Consultor/Profesor.
Postcondición	Se muestra la interfaz del gestor de preguntas.
Flujo normal principal	
1. El Consultor/Profesor selecciona "Gestión Preguntas"	
Flujo alternativo	
Ninguno.	

Caso de Uso: Crear Pregunta	
Resumen de la funcionalidad	Se encarga de añadir una pregunta a la base de datos.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Preguntas, Añadir Opción.</u>
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Gestión Preguntas</u> ".
Postcondición	Se añade la pregunta a la base de datos.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona "<u>Crear Pregunta</u>". 2. El sistema muestra un formulario para introducir datos de la nueva pregunta. 3. El Consultor/Profesor introduce los datos y los confirma. 4. La pregunta pasará a estar disponible en la BD con lo que se podrán realizarse consultas y modificaciones de la misma o incluirse en los formularios. 	
Flujo alternativo	
3.1. Faltan datos o son incorrectos por lo que el sistema muestra un aviso pidiendo que se completen correctamente estos datos.	

Caso de Uso: Añadir Opción	
Resumen de la funcionalidad	Permite añadir respuestas al formulario.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Crear Pregunta.</u>
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Crear Pregunta</u> ".
Postcondición	Se ha añadido una posible respuesta a la pregunta.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona una pregunta. 2. El consultor añade posibles respuestas, 3. Es sistema actualiza los datos de la pregunta. 	
Flujo alternativo	
Ninguno.	

Caso de Uso: Listar Preguntas	
Resumen de la funcionalidad	Muestra un listado de preguntas existentes en la BD.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Preguntas, Editar Pregunta.</u>
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Gestión Preguntas</u> ". Se tendrá la opción de seleccionar una pregunta para editarla.
Postcondición	Se muestra una lista de las preguntas.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona "<u>Listar Preguntas</u>". 2. El sistema muestra una lista de las preguntas. 3. El administrador podrá editar las preguntas. 	
Flujo alternativo	
3.1. El Consultor/Profesor selecciona una pregunta para editarla ejecutándose el caso de uso " <u>Editar Pregunta</u> ".	

Caso de Uso: Selecciona Pregunta	
Resumen de la funcionalidad	Selecciona una pregunta dentro de la base de datos.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Preguntas</u> , <u>Editar Pregunta</u> .
Precondición	Ninguna.
Postcondición	Se accede a la ficha de la pregunta.
Flujo normal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona un tema de la lista. 2. Se indica la pregunta a buscar de la lista de preguntas del tema. 3. El sistema muestra la pregunta indicada. 	
Flujo alternativo	
Ninguno	

Caso de Uso: Editar Pregunta	
Resumen de la funcionalidad	Se encarga de modificar los datos relativos a una pregunta existentes en la base de datos.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Buscar Pregunta</u> , <u>Listar Preguntas</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Listar Preguntas</u> " o " <u>Buscar Pregunta</u> ". La pregunta existe en la BD y no forma parte de formularios ya completados.
Postcondición	Los datos de la pregunta se han modificado en el sistema.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona o busca la pregunta cuyos datos quiere modificar. 2. El sistema muestra un formulario con los datos actuales de la pregunta. 3. El Consultor/Profesor cambia los datos que considere oportunos. 4. El sistema actualiza los datos en la base de datos. 	
Flujo alternativo	
Ninguno.	

Caso de Uso: Gestión Formularios	
Resumen de la funcionalidad	Se encarga de mostrar al Consultor/Profesor la interfaz para la gestión de los formularios.
Actores	Consultor/Profesor.
Casos de uso relacionados	<u>Crear Formulario</u> , <u>Editar Formulario</u> , <u>Eliminar Formulario</u> .
Precondición	El usuario se ha identificado en el sistema como Consultor/Profesor.
Postcondición	Se muestra la interfaz del gestor de formularios.
Flujo normal principal	
2. El Consultor/Profesor selecciona " <u>Gestión Formularios</u> "	
Flujo alternativo	
Ninguno.	

Caso de Uso: Crear Formulario	
Resumen de la funcionalidad	Se encarga de añadir un formulario a la base de datos.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Formularios</u> , <u>Editar Formulario</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Crear Formulario</u> ".
Postcondición	Se añade un formulario a la base de datos.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona "<u>Crear Formulario</u>". 2. Se accede a las páginas correspondientes para crear el formulario añadiéndole preguntas. 3. Una vez añadidas las preguntas se finaliza el formulario y se guarda en la BD. 	
Flujo alternativo	
Ninguno.	

Se ha considerado que no sería necesario eliminar los cuestionarios del sistema. Aunque en una futura implementación se desarrollará el caso de uso añadiendo una serie de restricciones en su uso.

Caso de Uso: Eliminar Formulario	
Resumen de la funcionalidad	Se encarga de eliminar un formulario a la BD.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Formularios</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Gestión Formulario</u> ". El formulario existe en la BD y no ha sido completado por ningún estudiante.
Postcondición	Se ha eliminado el formulario de la BD.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona el formulario a eliminar. 2. El sistema elimina el formulario de la BD tras solicitar confirmación. 	
Flujo alternativo	
Ninguno.	

La edición de formularios se ha limitado a su fase de creación. En una futura implementación se facilitarán las herramientas para realizar ediciones de formularios, manteniéndose la integridad de la BD y conservando los datos para futuras estadísticas como podría ser para los datos obtenidos de cuestionarios de tipo encuesta.

Caso de Uso: Editar Formulario	
Resumen de la funcionalidad	Permite modificar los formularios nuevos que se han creado antes de que se hagan accesibles a los estudiantes para su cumplimentación.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Gestión Formularios</u> , <u>Crear Formulario</u> , <u>Añadir Pregunta</u> , <u>Quitar Pregunta</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Gestión Formularios</u> " o " <u>Crear Formulario</u> ".
Postcondición	Se ha podido modificar el formulario añadiéndole o eliminando preguntas.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor añade o quita una pregunta del formulario. <ol style="list-style-type: none"> 1.1. Se ejecuta el caso de uso "<u>Añadir Pregunta</u>". 1.2. Se ejecuta el caso de uso "<u>Quitar Pregunta</u>". 2. Es sistema actualiza los datos en la base de datos. 	
Flujo alternativo	
Ninguno.	

Caso de Uso: Añadir Pregunta	
Resumen de la funcionalidad	Permite añadir preguntas al formulario.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Crear Formulario</u> , <u>Crear Pregunta</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Crear Formularios</u> ".
Postcondición	Se ha añadido una pregunta en el formulario.
Flujo normal principal	
<ol style="list-style-type: none"> 4. El Consultor/Profesor selecciona una pregunta a añadir al formulario. 5. Es sistema actualiza los datos del formulario. 	
Flujo alternativo	
1.1. Si no existe la pregunta se ejecuta el caso de uso " <u>Crear Pregunta</u> " para crearla y posteriormente añadirla al formulario.	

Caso de Uso: Quitar Pregunta	
Resumen de la funcionalidad	Permite quitar preguntas al formulario.
Actores	Consultor/Profesor
Casos de uso relacionados	<u>Crear Formulario</u> .
Precondición	El Consultor/Profesor ha ejecutado el caso de uso " <u>Crear Formularios</u> ".
Postcondición	Se ha quitado la pregunta indicada del formulario.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Consultor/Profesor selecciona una pregunta a quitar. 2. Es sistema actualiza los datos del formulario que se está creando. 	
Flujo alternativo	
Ninguno.	

2.2.3. Actor Estudiante.

Este actor se corresponde al de los estudiantes registrados como tal en la base de datos del sistema y que han realizado “LogIn” en la aplicación. A este perfil se accede únicamente tras una correcta identificación del usuario.

A continuación se detallan los diferentes casos de uso para este actor.

Pendiente de implementar al 100%

Caso de Uso: Completar Formularios	
Resumen de la funcionalidad	Permite cumplimentar a los estudiantes los diferentes formularios que se han creado y que se han puesto a disposición de los estudiantes para que los completen.
Actores	Estudiante
Casos de uso relacionados	<u>Enviar Formulario.</u>
Precondición	Existen formularios a disposición de los estudiantes para completarlos.
Postcondición	Se ha accedido a un formulario para cumplimentarlo.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El estudiante selecciona un formulario. 2. El estudiante responde el formulario. 3. Cuando se haya completado el formulario, el estudiante lo envía ejecutando el caso de uso “<u>Enviar Formulario</u>”. 	
Flujo alternativo	
<ol style="list-style-type: none"> 3.1. Es estudiante decide abandonar el formulario sin enviarlo. Esta opción no estaría disponible en el caso de formularios de tipo examen usados para evaluar a los estudiantes. 	

Pendiente de implementar al 100%

Caso de Uso: Enviar Formulario	
Resumen de la funcionalidad	Se envía el formulario cumplimentado por el estudiante.
Actores	Estudiante
Casos de uso relacionados	<u>Completar Formulario.</u>
Precondición	El estudiante ha ejecutado el caso de uso “ <u>Completar Formularios</u> ” y ha respondido a las diferentes preguntas del formulario.
Postcondición	El formulario completado se guarda en la BD y se marca como realizado por el estudiante.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El Estudiante selecciona la opción “<u>Enviar Formulario</u>”. 2. Es sistema tras pedir confirmación actualiza los datos del formulario en la BD. 	
Flujo alternativo	
Ninguno.	

2.2.4. Actor Administrador.

Usuario registrado y que se ha autenticado en el sistema como administrador. Realiza la gestión de los usuarios registrados en el sistema entre otras funciones, pudiendo obtener diferentes listados e información de los usuarios.

Caso de Uso: Gestión Usuarios	
Resumen de la funcionalidad	Se encarga de mostrar al administrador la interfaz para la gestión de usuarios.
Actores	Administrador.
Casos de uso relacionados	<u>Listar Usuarios</u> , <u>Editar Usuario</u> , <u>Crear Usuario</u> .
Precondición	El usuario se ha identificado como administrador.
Postcondición	Se muestra la interfaz del gestor de usuarios.
Flujo normal principal	
1. El Consultor/Profesor selecciona " <u>Gestión Usuarios</u> ".	
Flujo alternativo	
Ninguno.	

Caso de Uso: Editar Datos Usuario
Este caso de uso es el mismo que el denominado Editar Usuario con la salvedad de que en este caso el administrador puede editar los datos de cualquier usuario del sistema que seleccione.

Caso de Uso: Listar Usuarios	
Resumen de la funcionalidad	Muestra un listado de los usuarios registrados
Actores	Administrador.
Casos de uso relacionados	<u>Gestión Usuarios</u> , <u>Editar Usuario</u> .
Precondición	El usuario se ha identificado como administrador. Existen usuarios registrados en el sistema.
Postcondición	Se muestra una lista de los usuarios registrados. El administrador tendrá la opción de dar de editar los datos de los usuarios listados.
Flujo normal principal	
1. El administrador selecciona " <u>Listar Usuarios</u> ". 2. El sistema muestra una lista de los usuarios existentes en la BD. 3. El administrador podrá seleccionar y editar los datos de los usuarios.	
Flujo alternativo	
3.1. El administrador selecciona un usuario para editarlo, ejecutándose el caso de uso " <u>Editar Usuario</u> ".	

Caso de Uso: Crear Usuario	
Resumen de la funcionalidad	Se encarga de añadir un usuario a la base de datos.
Actores	Administrador.
Casos de uso relacionados	<u>Gestión Usuarios.</u>
Precondición	El administrador ha ejecutado el caso de uso " <u>Crear Formulario</u> ". El usuario no existe en la base de datos.
Postcondición	Se añade un nuevo usuario al sistema.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El administrador selecciona "<u>Crear Usuario</u>". 2. El sistema muestra un formulario para introducir los datos del usuario. 3. El administrador introduce los datos y los confirma. 4. El usuario pasará a estar registrado en el sistema y sus datos disponibles en la BD. 	
Flujo alternativo	
<ol style="list-style-type: none"> 3.1. Faltan datos o son incorrectos por lo que el sistema muestra un aviso pidiendo que se completen correctamente estos datos. 3.2. Existe ya un usuario con ese nombre en el sistema y se informa al administrador de esta incidencia. 	

Caso de Uso: Herramientas	
Resumen de la funcionalidad	Permite acceder al administrador a una serie de herramientas administrativas para la gestión del sitio web.
Actores	Administrador.
Casos de uso relacionados	<u>Ninguno.</u>
Precondición	Se ha accedido al sistema como administrador.
Postcondición	Se accede a un menú con diferentes herramientas administrativas.
Flujo normal principal	
<ol style="list-style-type: none"> 1. El administrador selecciona "<u>Herramientas</u>". 2. El sistema muestra una página con distintas herramientas administrativas. 	
Flujo alternativo	
Ninguno	

2.3.Requisitos no funcionales.

Para el diseño de la aplicación se han tenido en cuenta una serie de requisitos no funcionales que relacionamos a continuación:

2.3.1. Rendimiento.

La aplicación debe permitir la concurrencia de usuarios sin que el tiempo de respuesta del sistema se vea afectado negativamente. El tiempo de carga deberá ser lo más corto posible

por lo que se intentará optimizar al máximo las imágenes y el resto de elementos que utilizamos en el sitio.

Como ya habíamos indicado se usará MySQL como sistema de almacenamiento de datos por lo que deberemos tener presente todos los aspectos relacionados con la BD que afectan al rendimiento global de la aplicación y la integridad de los datos que manejaremos en ella.

2.3.2. Usabilidad.

Se pretende que la interfaz de usuario sea lo más sencilla e intuitiva posible, con fácil localización y acceso a los diferentes recursos que se pretenden ofrecer a cada usuario en sitio Web, dando una información precisa a cada usuario que está interactuando con el sitio Web de donde se encuentra en cada momento.

2.3.3. Accesibilidad.

En esta fase del desarrollo del proyecto no se han tenido en cuenta los requisitos específicos de accesibilidad. Se pretende sin embargo incluirlos en desarrollos futuros que se realizarán de la aplicación sin obviarlos en el desarrollo inicial.

2.3.4. Seguridad.

El acceso a la aplicación está limitado a aquellos usuarios registrados en el sistema, por lo que el acceso solo puede realizarse mediante el uso de la combinación Usuario/Contraseña. Las diferentes páginas son accesibles pues tras la autenticación de los usuarios y solo accesibles para aquellos usuarios a las que se les permita el acceso.

2.3.5. Requisitos tecnológicos.

Estos se tendrán en consideración en la siguiente etapa del desarrollo del presente proyecto, concretamente, en la fase de diseño en su punto "Estudio de la tecnología a utilizar". Sin embargo como ya se avanzó algunos de los elementos o tecnologías serían:

- Uso de **frameworks** (Struts 2, Hibernate,...) y estándares que faciliten el desarrollo del proyecto en su etapa de implementación.
- Uso de patrones de diseño MVC.
- Uso del servidor de aplicaciones Apache TOMCAT, para gestionar servlets, JSP, ...
- Uso y configuración del entorno de desarrollo Eclipse.
- Uso de un SGBD relacional MySQL.

3. Diseño.

3.1. Análisis de las principales tecnologías.

3.1.1. Justificación de la elección de tecnologías.

Para el desarrollo del presente proyecto, como ya indicamos anteriormente, se ha usado la tecnología JSP. En concreto se ha decidido utilizar la plataforma Java EE (J2EE) de la cual forman parte JSP.

PHP, JSP o .NET son los lenguajes más utilizados por la mayoría de las páginas Web con procesos del lado del servidor. Realizando cualquier consulta en internet relativa a estos hemos podido obtener muchas justificaciones en apoyo del uso de uno u otro para el desarrollo de aplicaciones Web. Hemos podido ver que cada uno de ellos tiene sus ventajas e inconvenientes y que en la elección del uso de uno u otro depende del proyecto a desarrollar.

De entrada se descartó el uso de las tecnologías .NET por no ser una tecnología Open Source, y se optó por la tecnología Java EE por varios motivos:

- Utiliza Java como lenguaje de programación el cual presenta un conjunto de ventajas sobre las demás alternativas en lenguajes de desarrollo:
 - Simple y poderoso
 - Seguro
 - Orientado por Objetos
 - Robusto
 - Interactivo
 - Independiente de arquitectura de hardware
 - Interpretado y rápido
 - Fácil de aprender
- Hay gran cantidad de bibliotecas de terceros para el desarrollo web en Java.
- Soporte: API's, manuales, gran cantidad de ejemplos,...
- Muchas utilidades ya creadas fácilmente integrables en el proyecto.

PHP también tiene muchas características a su favor, y es una de las plataformas Web más utilizada, pero dado que en los últimos meses la mayor parte de mis trabajos han estado relacionados con Java tanto en mis estudios en la UOC como a nivel profesional, donde necesito ampliar mis conocimientos (aun escasos) en esta plataforma han hecho que decidiera realizar el proyecto usando JSP.

3.1.2. Plataforma J2EE (Java EE).

Para el desarrollo de la aplicación se ha utilizado la tecnología J2EE que nos proporciona las herramientas necesarias para la creación de aplicaciones distribuidas, además de ser una tecnología Open Source.

J2EE utiliza el lenguaje Java, es independiente de la plataforma, dispone de páginas web dinámicas (JSP) y conectividad con Bases de Datos (JDBC -Java Data Base Connectivity-). Se tiene libertad de elección entre distintos servidores de aplicaciones y entornos de desarrollo libres como Eclipse o NetBeans. Existen frameworks que facilitan el desarrollo de las aplicaciones (Struts 2, Hibernate, Spring, ...) y además hay disponible gran cantidad de soporte en la red.

Esta plataforma permite mejorar el coste y la complejidad de desarrollo multi-tier, facilitando que las aplicaciones web tengan una alta disponibilidad, seguridad, fiabilidad y sean fácilmente escalables además de reutilizables. J2EE está formada básicamente por tres las capas:

- **Capa Cliente**
Los componentes de esta capa se ejecutan en la máquina del cliente. En ella el usuario interactúa con la aplicación.
- **Capa intermedia o capa de negocio.**
Los componentes de esta capa se ejecutan en el servidor J2EE
 - Capa web.
Esta capa proporciona a la aplicación funcionalidad para internet y permite proporcionar servicios a la capa cliente.
 - Capa de negocio (EJB- Enterprise Java Beans-).
Contiene la lógica del negocio
- **Capa de información empresarial (EIS) - datos -.**
Conecta la aplicación J2EE con el sistema y los recursos de la red empresarial como la base de datos.

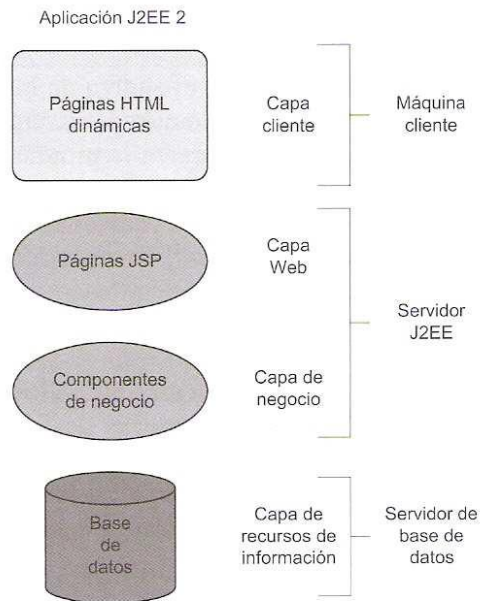


Figura 3. Arquitectura J2EE.

3.1.3. Arquitectura de la aplicación.

El diseño del proyecto, basado en la plataforma J2EE, sigue el modelo multicapa e integrará varias tecnologías. El esquema general de la aplicación es:

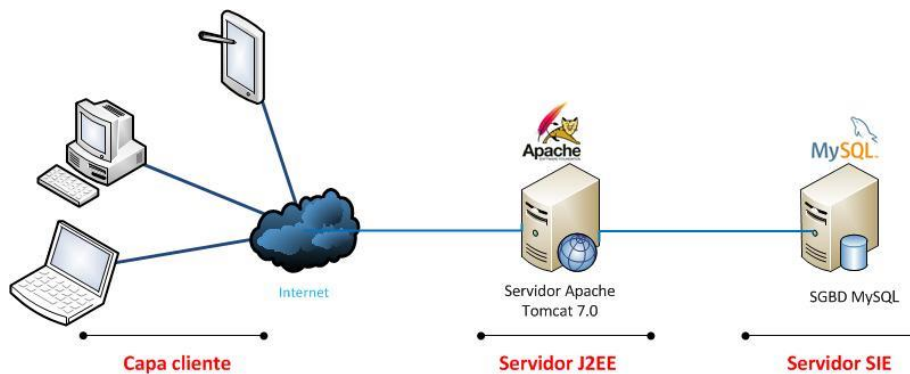


Figura 4. Arquitectura aplicación.

- Capa cliente:** Se implementa mediante páginas Web, tiene como misión la captura de datos de usuario y su envío a la capa intermedia y la presentación posterior de los resultados devueltos por esta capa intermedia. Para las interfaces de usuario utilizaremos JSP así como una serie de bibliotecas de tags que facilitan la implementación. Struts 2, la biblioteca estándar de tags JSP, ...

- **Capa intermedia, servidor J2EE:** Esta capa es la que constituye el verdadero núcleo de la aplicación. En esta se procesan los datos de usuario y es donde se generan y envían las respuesta a este (capa cliente). Está constituida por el servidor de aplicaciones web donde estará almacenada nuestra aplicación para hacerla accesible a través de una conexión a internet. En nuestro caso se utilizará el servidor de aplicaciones Apache Tomcat 7.0.41, plataforma de libre distribución que se integra perfectamente con Java.
- **Servidor SIE (SGBD):** Es la capa de datos con la que interactúa la capa intermedia, almacenando o recuperando datos. Para la gestión de la persistencia se ha optado por la utilización de MySQL, base de datos relacional también dispone en versión de libre distribución, aunque siguiendo el modelo DAO no debería ser problema utilizar cualquier otro SGBD.

Como IDE para el desarrollo de la aplicación se ha optado por Eclipse, plataforma de libre distribución y que ofrece múltiples funcionalidades.

Con el objeto de facilitar el desarrollo de la aplicación se han usado los frameworks Hibernate y Struts 2, siguiendo para su desarrollo los patrones de diseño MVC (Modelo-Vista-Controlador) y DAO (Data Access Object).

El esquema de la aplicación es el siguiente:

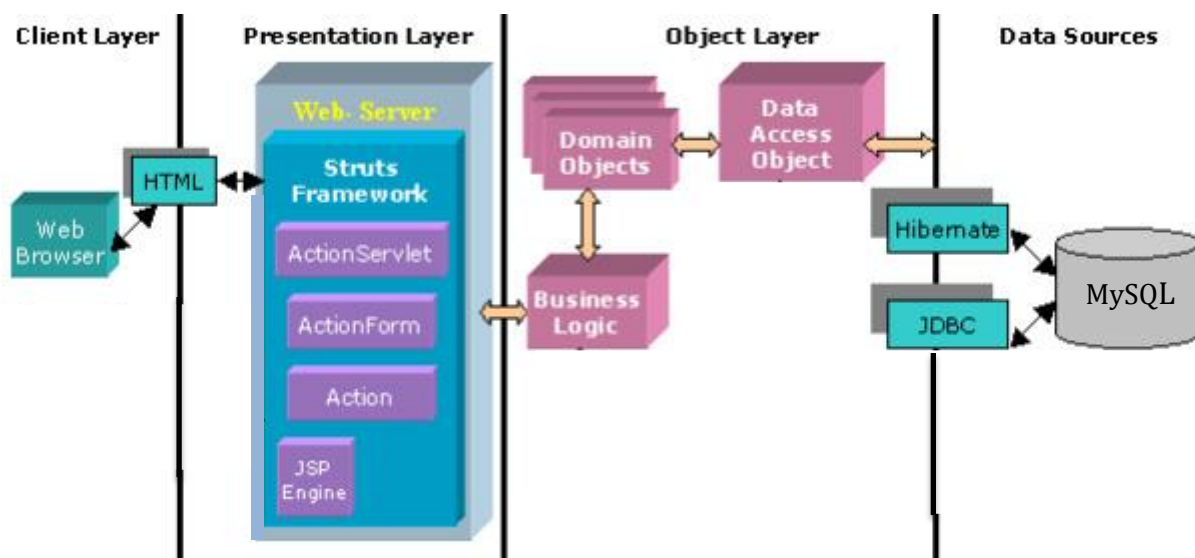


Figura 5. Esquema tecnología de la aplicación.

3.1.4. Patrones de diseño.

3.1.4.1. MVC

Este patrón realiza una separación entre el modelo (lógica de negocio), la vista (interfaz gráfica) y el controlador. Nos proporciona una separación entre las distintas responsabilidades de la aplicación Web.

Especifica cómo debe ser estructurada la aplicación, las capas que la compondrán y la funcionalidad de cada una de ellas. Siguiendo este patrón la capa intermedia de la aplicación puede ser dividida en tres bloques funcionales que aparecen reflejados en la siguiente figura:



Figura 6. Modelo-Vista-Controlador (MVC)

- **Modelo:**
Representa la información con la que trabaja a aplicación, su lógica de negocio. Contiene todos los objetos y reglas de negocio, la implementación de las funcionalidades y los datos con que trata la aplicación.
- **Vista:**
Define la interfaz de usuario. Permite al usuario interactuar con el modelo. En ella el usuario realiza acciones que son procesadas por nuestra aplicación enviando de vuelta los datos necesarios para generar la respuesta a esta acción.
- **Controlador:**
Es la base de la aplicación y en él se centraliza todo el flujo de peticiones, proporcionando significado a las órdenes del usuario, convirtiéndolas en llamadas al modelo y vista. Su misión es determinar las acciones a realizar para cada una de las

peticiones recibidas, invocando al resto de los componentes de la aplicación – modelo y vista – para que se puedan realizar todas las acciones solicitadas. En esta capa se define cuando se debe modificar la base de datos o cuando obtener datos de ella para hacerlos disponibles para la capa vista.

Con ello se consiguen unos desarrollos más sencillos y limpios, facilitándose la detección de errores y el posterior mantenimiento y haciendo las aplicaciones más escalables.

El patrón MVC en la programación web J2EE se conoce como **arquitectura de modelo 2**. Esta arquitectura consiste en la utilización de Servlets para procesar las peticiones, que estarían contenidos en el Controlador del patrón, y páginas JSP para mostrar la interfaz del usuario que representaría la Vista, y finalmente los JavaBeans ubicados en el modelo.

3.1.4.2. DAO

El patrón Data Access Object (DAO) pretende hacer independiente una aplicación respecto de una base de datos concreta.

El conjunto de acciones del sistema trabajan con clases genéricas y no se preocupan de la implementación de las mismas. Esto permite que se pueda variar la fuente de datos dejando intacta la lógica del negocio. Actúa como un adaptador entre el componente y la fuente de datos. Un objeto DAO encapsula la información y se pone en contacto con el sistema de persistencia, sin importar cual se ha elegido, para gestionar la transferencia de información.

La capa DAO contiene un conjunto de clases DAO que serán las encargadas de realizar los accesos a la base de datos. El DAO manipula y controla las conexiones con la BD y define las diferentes formas de manipular los datos. El patrón DAO define los Data Transfer Objects que simulan la conversión de un objeto Java a una tabla de base de datos.

Es en este punto donde se ha utilizado el framework Hibernate que es el encargado de llevar a cabo este cometido.

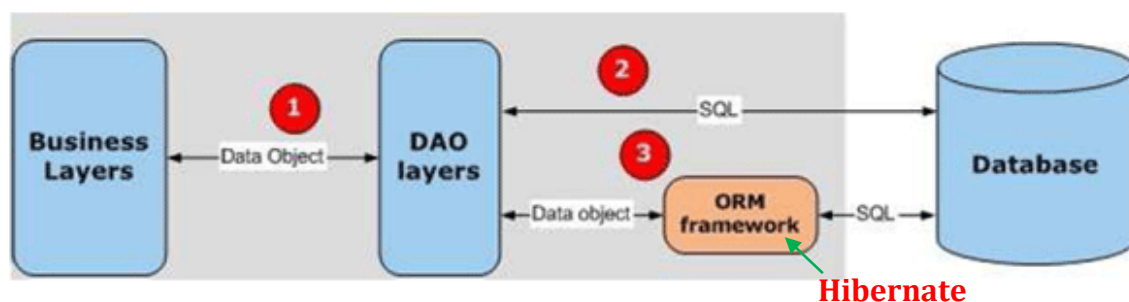


Figura 7. Arquitectura patrón DAO

3.1.5. Frameworks.

“En el desarrollo de software, un **framework** o **infraestructura digital**, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto...” (Wikipedia, <http://es.wikipedia.org/wiki/Framework>)

Durante el desarrollo del proyecto hacemos uso de estos elementos para agilizar el desarrollo de la aplicación Web, ya que nos permiten automatizar tareas comunes, acelerando e incrementando el proceso de desarrollo, promoviendo buenas prácticas de desarrollo con el uso de los patrones y facilitando la reutilización de código ya existente.

Inicialmente su uso puede aumentar la carga de trabajo, como ha sido el caso en este proyecto, debido al tiempo que debe emplearse en su aprendizaje pero las ventajas que su uso nos pueden aportar futuros desarrollos, han hecho que hayamos optado por su uso.

3.1.5.1. Struts 2.

Framework para el desarrollo de aplicaciones web realizadas bajo la plataforma J2EE que agiliza la implementación siguiendo el patrón MVC. Dentro del patrón de diseño MVC, Struts 2 implementa el controlador y además proporciona algunos componentes para la capa vista.

Entre sus características distinguiríamos la configuración del control centralizada, las librerías de entidades para facilitar operaciones que realizan las paginas JSP como las relativas a seguridad SSL y contiene herramientas para la validación de campos. Además permite agregar funcionalidades mediante el uso de plugins como Tiles que facilitan la implementación de las páginas JSP.

Struts 2 procesa las peticiones usando una serie de componentes principales:

- **FilterDispatcher:**

Representa el punto de entrada a la aplicación, dirigiéndose a él todas las peticiones que llegan del cliente.

- **Interceptores:**

Estos permiten que implementen funcionalidades que afectan a uno o varios Actions, pero ejecutándose fuera de estos. Realizan tareas antes y después de los Actions y pueden evitar que estos se ejecuten. Esto permite añadir funcionalidades no relacionadas directamente con el modelo. Como ejemplo de estos se encontrarían los conversores de tipos, los destinados a la validación de datos, etc.

- **Actions:**

Son las encargadas de realizar la lógica para servir una petición. Definen cual será la clase que procesará la petición y cuál será la respuesta en función del resultado. Estas acciones las podemos definir mediante anotaciones en las clases o dentro de struts.xml si realizamos la configuración mediante XML.

Entre otros aspectos se pueden resaltar que:

- Los objetos Action forman parte del modelo.
- Se implementan como clases POJO.
- Incluyen los métodos set/get.

Con lo que simplifican las aplicaciones y se dispone directamente de los datos en la misma clase donde se manipulan.

- **Results:**

Estos determinan la respuesta apropiada, dependiendo del resultado obtenido después de procesar el Action. Determina que es lo que se debe hacer, enviar al usuario una JSP, redirigirlo, ejecutar otro Action,...

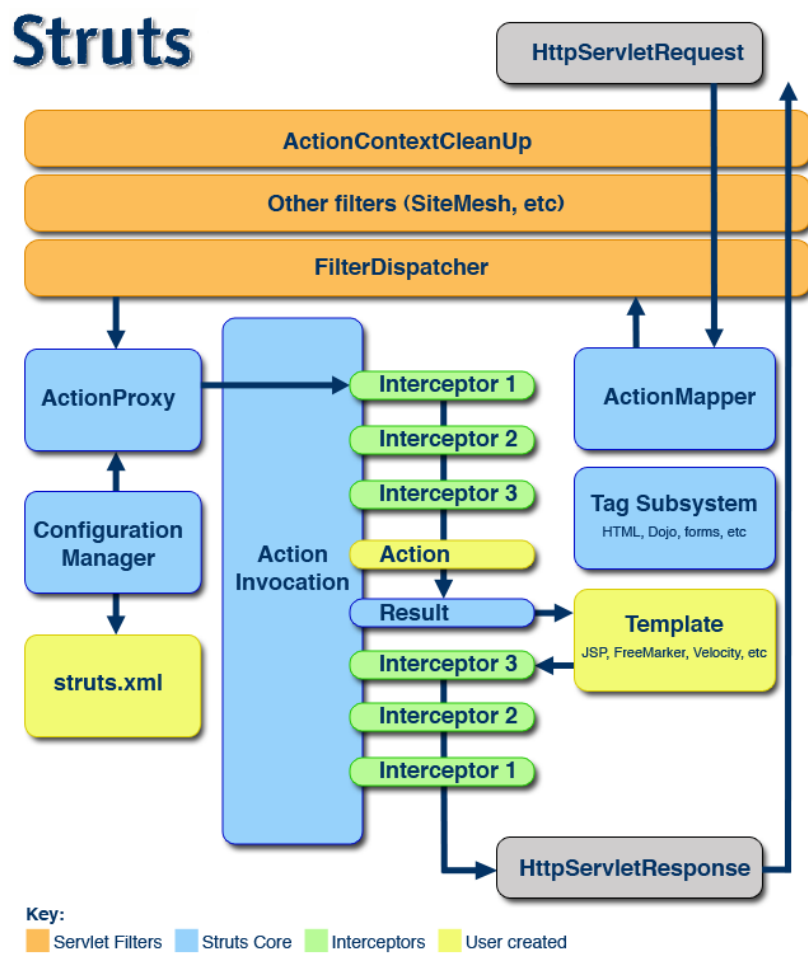


Figura 8. Esquema lógico Struts 2

Los principales ficheros de configuración que encontramos en Struts 2 son:

- **Struts.xml**: Es el archivo donde se define el marco de entrada a los diferentes archivos XML si optamos por realizar la declaración de la arquitectura mediante XML y no mediante anotaciones Java.
Define la lógica de presentación y navegación de la aplicación. Reglas de navegación de las diferentes páginas. Reglas de validación de formularios. Reglas de filtrado de peticiones. Acciones de formulario, que son las acciones que pueden ser llamadas desde la página del cliente.
- **web.xml**: Indica los elementos de la aplicación no relacionados directamente con Struts. Hace referencia al fichero inicial de la aplicación (index.jsp). En él se añade el filtro que sirve a Struts 2 como punto de entrada a la aplicación, el StrutsPrepareAndExecuteFilter, pasando por él todas las llamadas que se hagan a la aplicación.

3.1.5.2. Hibernate.

Es un framework, también Open Source, que facilita el mapeo de los atributos de las clases del modelo de negocio con los de una base de datos relacional.

Hibernate es un ORM (Object Relational Modeler) que permite crear una capa de abstracción para solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en la aplicación.

Para ello proporciona archivos declarativos **xml** (NombreClase.hbm.xml) donde se declaran las relaciones entre los dos dominios. Los datos de configuración de Hibernate los encontramos en el archivo **hibernate.cfg.xml**, que contiene información referente a SGBD utilizado, el driver a utilizar para la comunicación, puerto, usuario y contraseña para el acceso a la base de datos.

Hibernate lleva integrado un lenguaje de consultas HQL (Hibernate Query Language) aunque también reconoce las consultas de SQL.

3.1.5.3. Java Server Pages (JSP).

Es una tecnología usada para construir aplicaciones Web que presentan contenido de manera dinámica, páginas web dinámicas basadas en HTML, XML entre otros tipos de documentos. JSP es similar a PHP pero usa el lenguaje de programación Java.

Para desplegar las Java Server Pages, es necesario un servidor web compatible con contenedores servlet como es el caso de Apache Tomcat que usamos en nuestro desarrollo.

3.2. Diseño de clases.

El diagrama de clases tiene como objetivo identificar los objetos que forman parte del sistema a implementar.

Analizando los requisitos y los casos de uso descritos anteriormente, podemos extraer información para identificar las clases necesarias para la aplicación. En la figura 7 se muestra un diagrama de clases donde aparecen las clases que hemos identificado. Este diagrama es el que se realizó en las fases iniciales del proyecto y que no ha sufrido más que alguna pequeña modificación durante la fase de implementación para cubrir algunas de las necesidades que han ido surgiendo durante la fase de implementación, donde se a identificado alguna nueva clase.

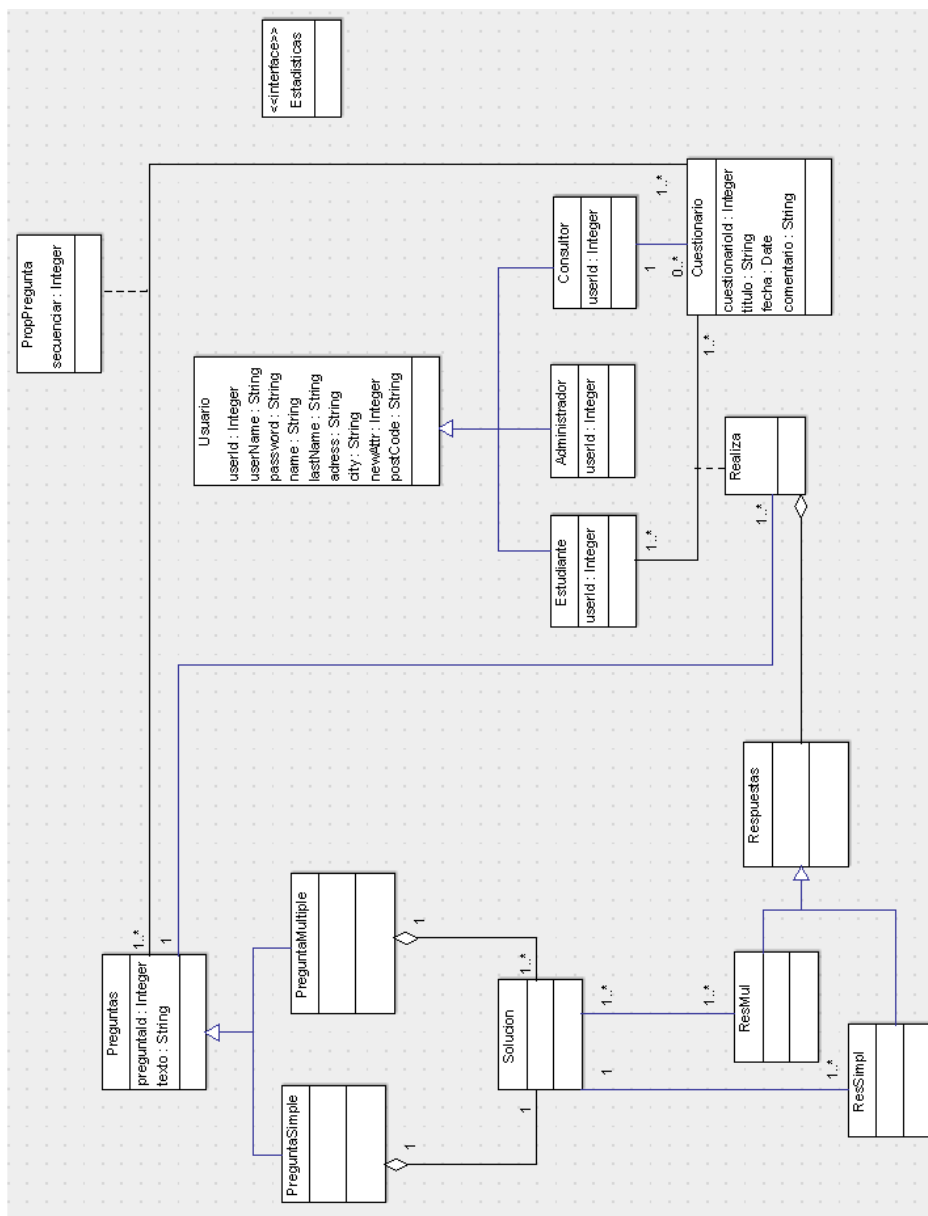


Figura 9. Diagrama de clases.

3.3. Diseño de la base de datos.

En este apartado se presenta el diseño de la base de datos que se ha utilizado en nuestra aplicación. Como sistema de gestión de base de datos se ha decidió escoger el gestor de base de datos MySQL ya que se trata de un software muy potente y de libre distribución.

Para el diseño de la base de datos se hizo uso del diagrama ER (Entidad-Relación) que permite definir la estructura de la base de datos independientemente de que sistema se haya seleccionado para su implementación.

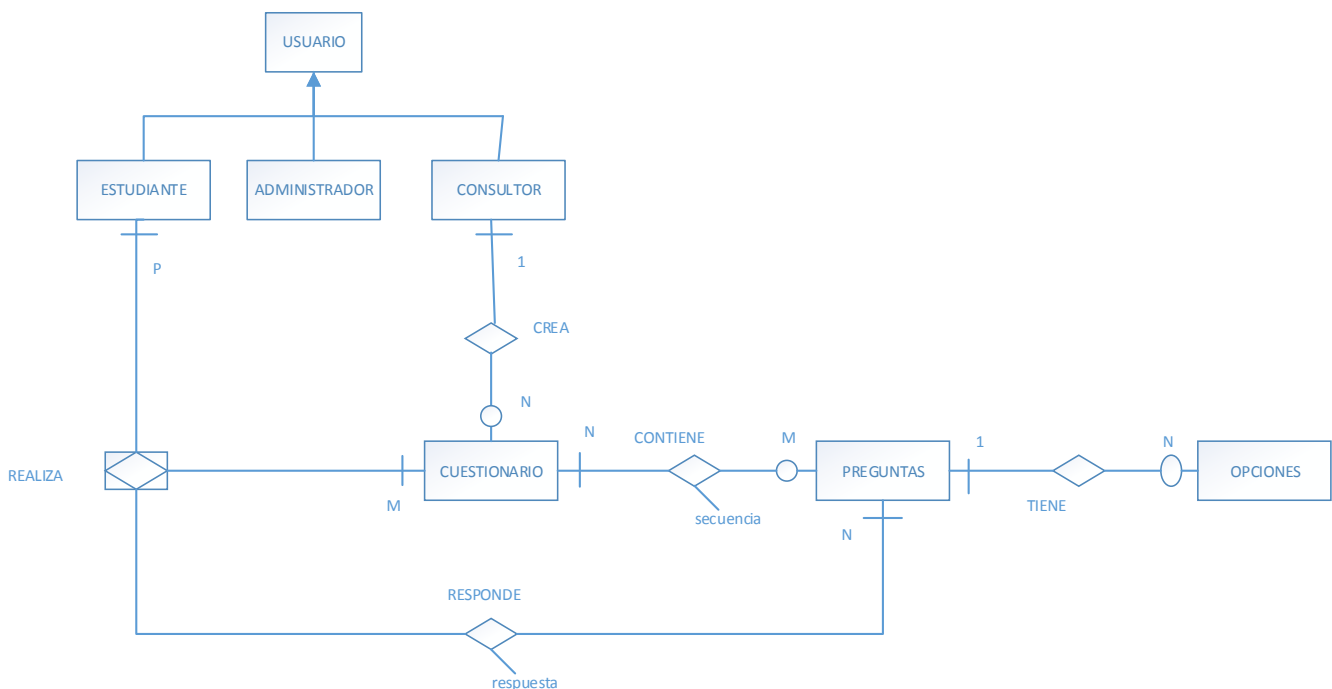


Figura 10. Modelo ER

Partiendo de este modelo ER se ha construido la base de datos en MySQL que se muestra en la figura siguiente.

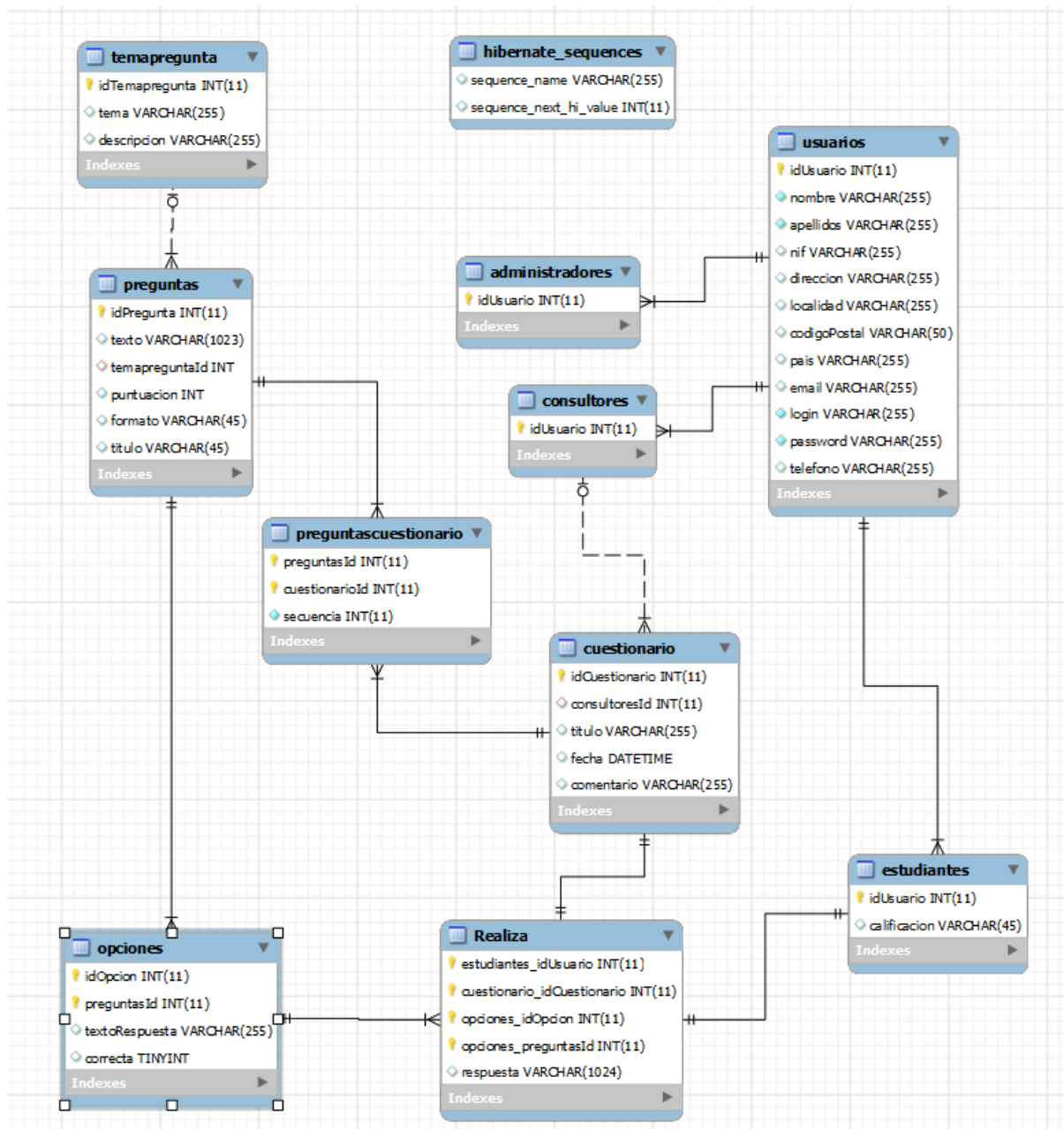


Figura 11. Esquema ERR MySQL

3.4. Modelización de la interfaz.

3.4.1. Mapa navegacional.

Durante las fases iniciales se diseñó un mapa navegacional mediante el cual mostramos cual era el flujo de pantallas que se preveía y que no ha variado sustancialmente dando una visión general de las acciones que se podrían realizar en cada pantalla y el paso de una a otra.

Para ello usábamos un diagrama de actividades mostrando el flujo de procesos que ofrece la aplicación.

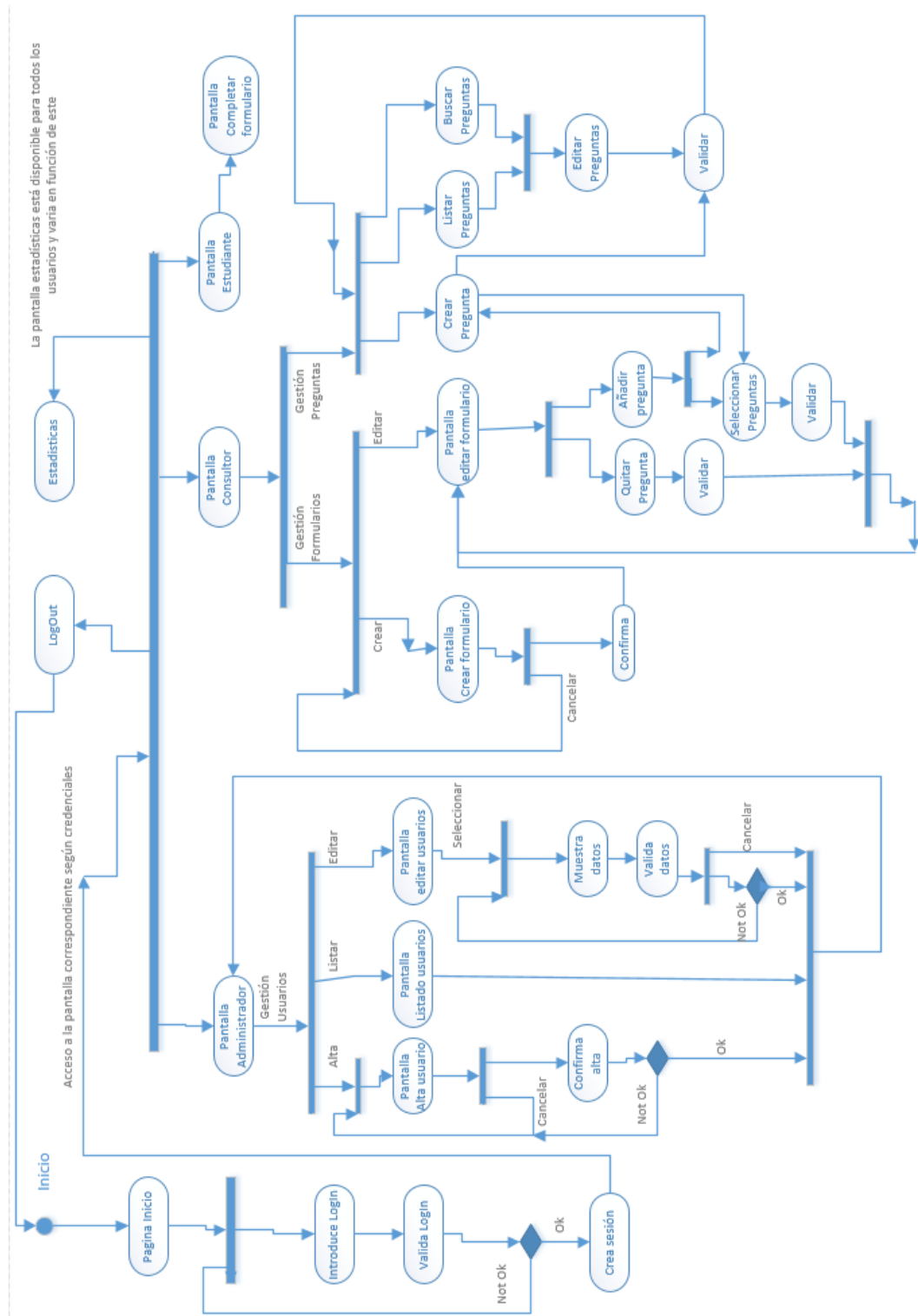


Figura 12. Mapa navegacional.

3.4.2. Prototipo de la interface gráfica de usuario.

A continuación se muestra un prototipo aproximado de cómo se previó que sería la interface gráfica mostrando a continuación de este algunas bocetos de las pantallas.

El esquema del formato de las páginas es el siguiente:

- **Imagen general**: En primer lugar tenemos una imagen con el logotipo de la aplicación web. En esta zona siempre tendremos un menú fijo para toda la aplicación. También en la parte derecha tendremos una zona variable en la que se mostrará información relativa al usuario que ha realizado el LogIn.
- **Menús secundarios**: Zona de menús diferentes que varían en función del usuario y de las opciones que haya seleccionado previamente
- **Pie de página**: Esta zona también será estática y estándar para todas las páginas jsp.
- **Body**: Esta es la zona donde la aplicación presentará todos los resultados de las acciones requeridas por el usuario (sea del tipo que sea).

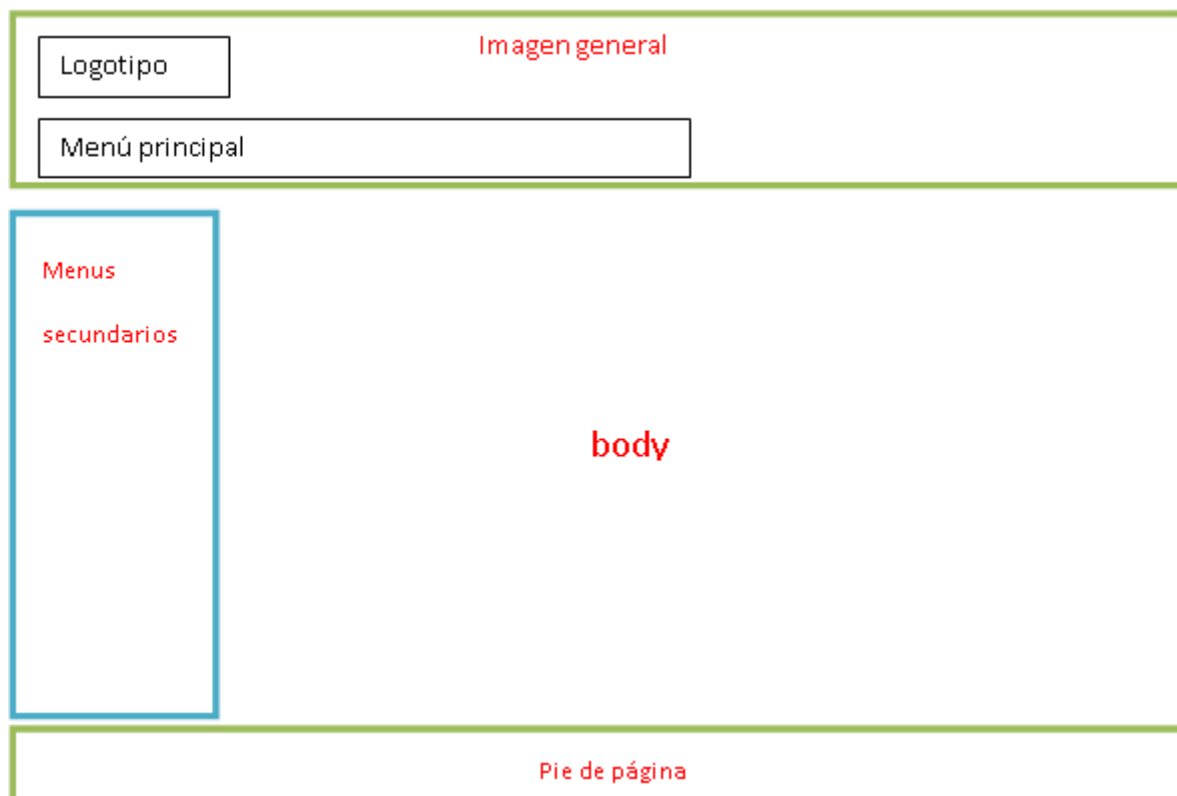


Figura 13. Prototipo interface.

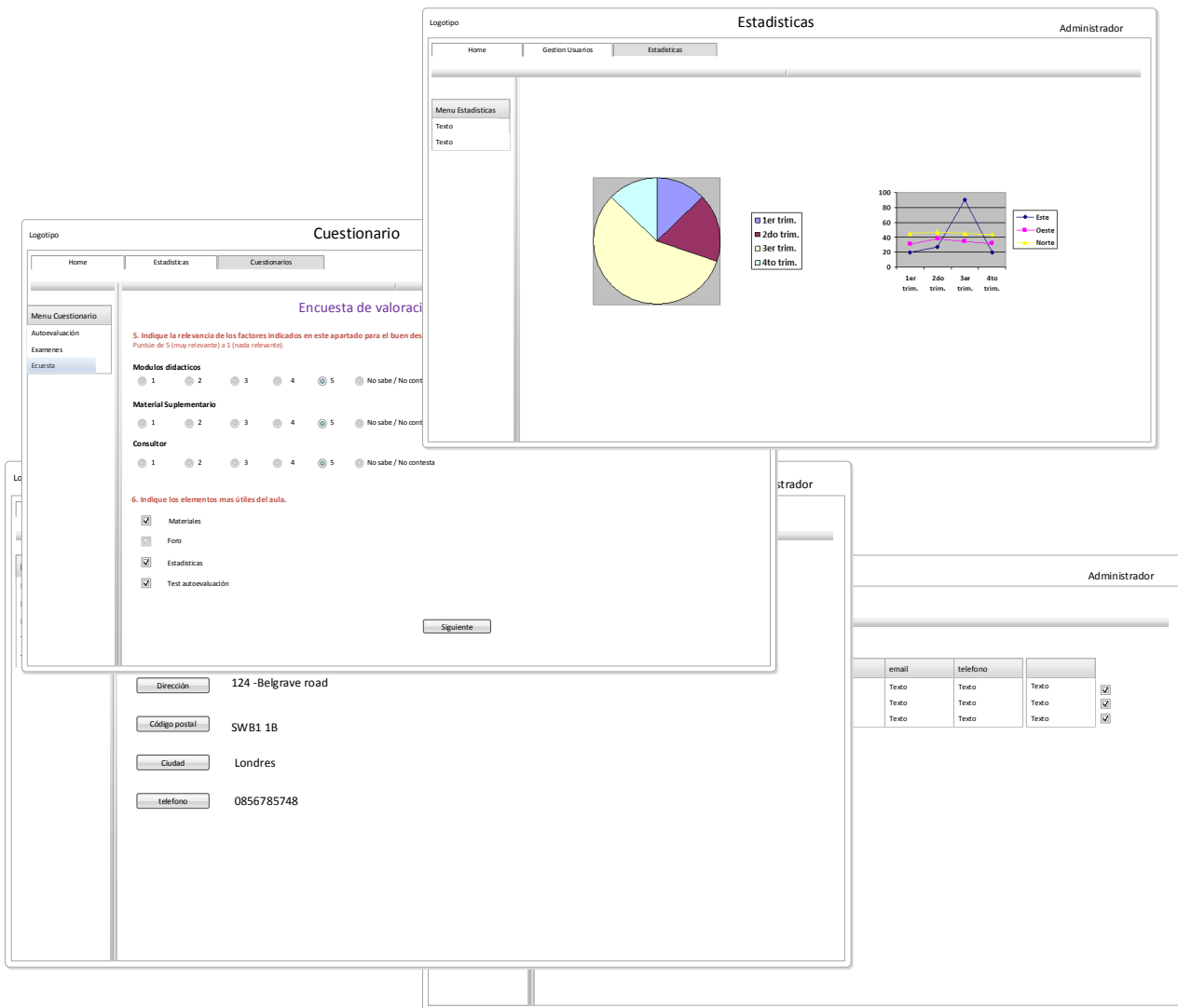


Figura 14. Prototipos pantallas.

4. Ampliación y propuestas de mejora.

Las ampliaciones y mejoras que se pueden realizar en un futuro en la aplicación abarcarían desde la inclusión de nuevas funcionalidades o la realización de cambios en la interface que mejoren su diseño y permitan optimizar la presentación de la información a los usuarios y la interacción de estos con las páginas del sitio.

En un primer lugar, dado las limitaciones de tiempo que han provocado que no se pudiera completar la implementación de los casos de uso relativos a la cumplimentación de los cuestionarios y a la presentación de informes relativa a la información que se obtiene con los mismos, se finalizaría la implementación de estas funcionalidades.

Posteriormente las opciones de ampliación posibles que he considerado serian entre otras:

- Añadir las opciones necesarias para realizar la corrección automática de los test.
- Añadir mejoras para la obtención de estadísticas e informes relativos a todos los datos que se manejan en los diferentes cuestionarios. Permitted la generación dinámica de estos informes a los consultores. Para ello y siguiendo la línea que ya se había empezado se utilizarían herramientas como JasperReport que nos permiten generar distintos informes y graficas e integrarlas en la aplicación.
- Añadir un sistema de control de tiempos para la realización de cuestionarios de tipo examen para establecer el periodo disponible para completarlos.
- Incluir mejoras en la edición de los datos personales de los usuarios que les permitan entre otras opciones subir una fotografía al sistema para incluirla en su ficha personal.
- Mejorar la navegación entre páginas así como la visualización que se obtiene de las mismas en los distintos navegadores.
- Añadir nuevas funcionalidades que permita que los estudiantes puedan realizar distintos cuestionarios, generados aleatoriamente a partir de las preguntas existentes en la base de datos del sistema.

5. Valoración económica.

En la realización del plan de trabajo y para la estimación de los costes del proyecto se han tenido en cuenta 3 roles diferentes del personal encargado del desarrollo del proyecto aunque estos serán desempeñados únicamente por una persona.

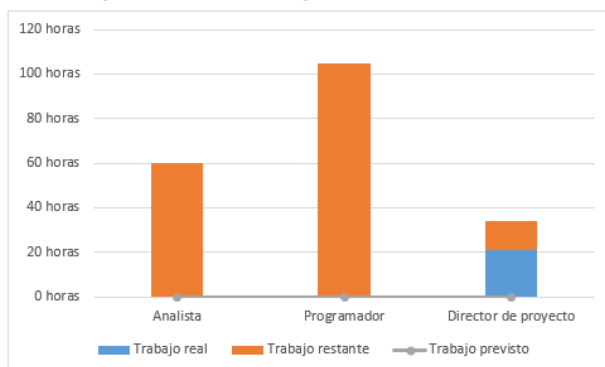
En la tabla siguiente se especifican estos roles y los honorarios asignados a cada uno de ellos y que son los utilizados en la estimación de costes para el desarrollo del proyecto:

Nombre	Tasa estándar
Analista	35,00 €/hora
Programador	21,00 €/hora
Director de proyecto	55,00 €/hora

VISIÓN GENERAL DE LOS RECURSOS

ESTADÍSTICAS DE RECURSOS

Estado de trabajo de todos los recursos de trabajo.



ESTADO DE LOS RECURSOS

Resta trabajo para todos los recursos de trabajo

Nombre	Comienzo	Fin	Trabajo restante
Analista	mar 11/03/14	lun 14/04/14	60 horas
Programador	mar 08/04/14	sáb 07/06/14	105 horas
Director de proyecto	mié 26/02/14	dom 08/06/14	13 horas

En las siguientes tablas y graficas se recogen los costes para el desarrollo del proyecto. Como se puede apreciar ya aparecen computados los costes generados en la fase inicial del proyecto y que se corresponde con la elaboración del presente plan de trabajo.

INFORMACIÓN GENERAL COSTOS

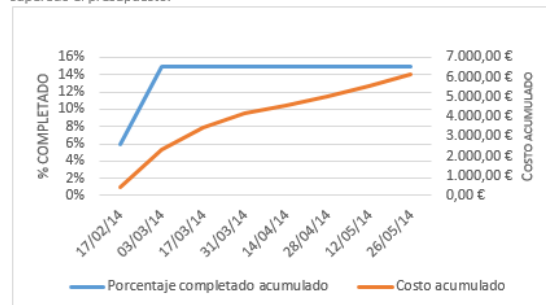
MIÉ 26/02/14 - DOM 08/06/14

COSTO

6.175,00 €

PROGRESO FRENTE A COSTO

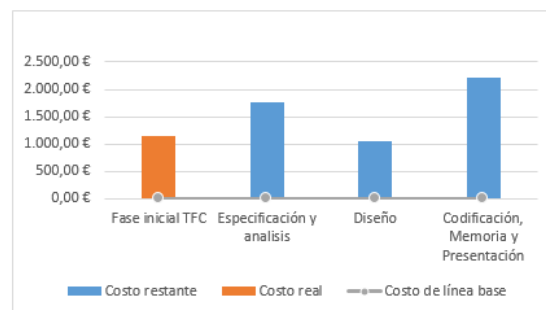
Progreso realizado en comparación con el coste durante el proceso. Si el valor de la línea % completado está por debajo de la línea de coste acumulado, es posible que su proyecto haya superado el presupuesto.



ESTADO DE COSTO

Estado de costo de todas las tareas de nivel superior. ¿La línea base es cero?

[Intente establecer una línea base](#)



ESTADO DEL COSTO

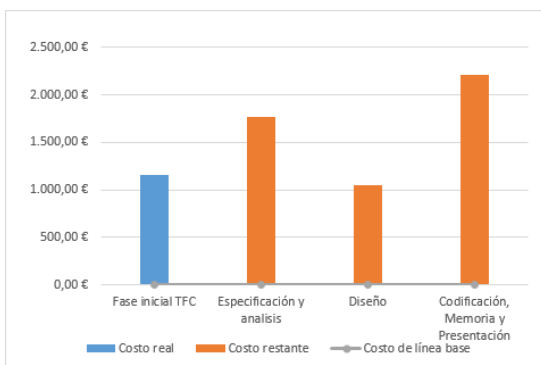
Estado de costo de tareas de nivel superior.

Nombre	Costo real	Costo restante	Costo de línea base	Costo	Variación de costo
Fase inicial TFC	1.155,00 €	0,00 €	0,00 €	1.155,00 €	1.155,00 €
Especificación y análisis	0,00 €	1.765,00 €	0,00 €	1.765,00 €	1.765,00 €
Diseño	0,00 €	1.050,00 €	0,00 €	1.050,00 €	1.050,00 €
Codificación, Memoria y Presentación	0,00 €	2.205,00 €	0,00 €	2.205,00 €	2.205,00 €

INFORMACIÓN GENERAL DE COSTOS DE LA TAREA

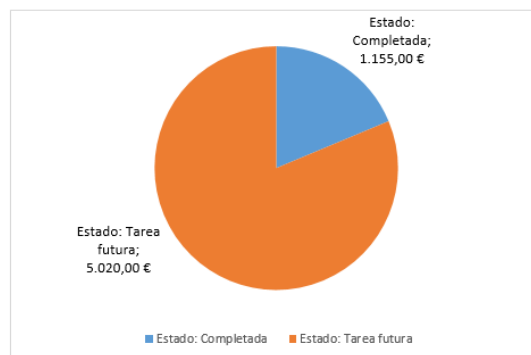
ESTADO DE COSTO

Estado de costo para las tareas de nivel superior.



DISTRIBUCIÓN DE COSTOS

Cómo los costos están distribuidos entre las tareas en función de su estado.



DETALLES DE COSTOS

Detalles de costos para todas las tareas de nivel superior.

Nombre	Costo fijo	Costo real	Costo restante	Costo	Costo de línea base	Variación de costo
Fase inicial TFC	0,00 €	1.155,00 €	0,00 €	1.155,00 €	0,00 €	1.155,00 €
Especificación y análisis	0,00 €	0,00 €	1.765,00 €	1.765,00 €	0,00 €	1.765,00 €
Diseño	0,00 €	0,00 €	1.050,00 €	1.050,00 €	0,00 €	1.050,00 €
Codificación, Memoria y Presentación	0,00 €	0,00 €	2.205,00 €	2.205,00 €	0,00 €	2.205,00 €

6. Conclusiones.

En este punto debo señalar que la aplicación no cumple al cien por cien los objetivos que se habían definido en las primeras fases del proyecto. Aunque la mayoría de las funcionalidades que se establecieron se han completado satisfactoriamente y su funcionamiento es correcto, han quedado por completar dos funcionalidades importantes como es la de contestar y almacenar las respuestas de los cuestionarios generados y a partir de estas obtener los distintos informes y graficas estadísticos. Estas funcionalidades representarían menos del 10% del trabajo total que ha implica el desarrollo de todo el proyecto, ya que consistirían en la implementación únicamente de una serie de JSP que permitiera presentar las preguntas a los estudiantes y almacenar sus respuestas en la base de datos y a partir de estas y de forma prácticamente automática generar unos informes utilizando JasperReports que presentaran de forma estadística estos resultados.

Por otro lado, y durante todo el desarrollo del proyecto se han tomado una serie de decisiones, como el uso de una determinada tecnología de desarrollo, seguir unos determinados modelos, el uso de unos determinados frameworks que en ocasiones han complicado el desarrollo de la aplicación.

Como he podido constatar, la curva de aprendizaje de algunos de estos frameworks es alta, pero a medida que se avanza en su estudio y se profundiza en su uso los resultados que se obtienen compensan el esfuerzo que se ha tenido que realizar inicialmente.

Uno de los objetivos que me había planteado inicialmente, aparte de conseguir acabar el desarrollo del sitio web, fue el de adquirir experiencia en las técnicas de desarrollo de software, desde las fases iniciales de identificación de los requisitos hasta la entrega e implantación del producto final, trabajando diferentes áreas que implicaran el estudio de nuevas herramientas y recursos o profundizar en los conocimientos ya adquiridos a lo largo de la carrera de diferentes asignaturas.

Durante todo el proceso han ido surgiendo algunos problemas relacionados con las herramientas utilizadas, pero el principal problema ha estado relacionado con el trabajo cotidiano que debía realizar fuera de lo que ha sido el proyecto, que han provocado que no pudiera cumplir la planificación que había planteado. En concreto gran parte de la planificación realizada para la fase de implementación y realización de la memoria se han visto alterada. No se han podido dedicar todas las horas que se establecieron en la planificación y que eran necesarias para alcanzar los objetivos planteados.

He podido constatar lo importante que resulta realizar una buena planificación ya que de ella depende en gran medida que cualquier proyecto culmine satisfactoriamente dentro de los plazos establecidos. No menos importante es realizar un buen análisis ya que con ello se facilita el posterior desarrollo del proyecto.

7 ■ Bibliografía y referencias.

Se ha utilizado material docente de la UOC de diferentes asignaturas (Ingeniería del software, Trabajo fin carrera,...) relacionadas con los diferentes puntos desarrollados en el presente documento. También

Bibliografía

Ian Roughley. Starting Struts 2. <http://infoq.com/minibooks/starting-struts2>

Donald Brown, Chad Michael Davis, Scott Stanlick (2008). Struts 2 in Action. Manning Publications Co.

Srinivas Guruzu and Gary Mak (2010). Hibernate Recipes . Apress

Antonio J. Martín Sierra (2010). Struts. Ra-Ma.

Jon Duckett (2011). Html & CSS . Wiley

Software utilizado.

- ArgoUML (Diagrama de clases de uso).
<http://argouml.tigris.org/>
- Microsoft Word 2010 (Desarrollo del documento presentado).
<http://office.microsoft.com>
 - Microsoft Project 2013 (Plan de trabajo)
 - Eclipse (IDE para el desarrollo)
www.eclipse.org

Referencias requisitos tecnológicos.

<http://www.mysql.com/>

<http://struts.apache.org/>

<http://tomcat.apache.org/>

<http://hibernate.org/>

Referencias en internet.

<http://www.mkyong.com/>

http://www.tutorialspoint.com/struts_2/index.htm

<http://www.javatutoriales.com/>

<http://docs.jboss.org/hibernate/orm/3.5/reference/en-US/html/index.html>

<http://code.google.com/p/cemicursoj2ee/wiki/Struts2>

<http://www.dzone.com/tutorials/java/struts-2/struts-2-tutorial/struts-2-tutorial.html>

<http://code.google.com/p/socialauth/>

ANEXOS

A. Acceso a la aplicación.

La aplicación desarrollada está disponible a través del siguiente enlace:

<http://uocsurveys-tfcuoc.rhcloud.com/>

Se ha utilizado OPENSIFT (www.openshift.com) integrado con eclipse para el desarrollo del proyecto.

Todo el código y el acceso a la base de datos están disponibles utilizando la siguiente cuenta en OpenShift:

User: consultortfc@gmail.com
Password: Consultor12

A través de esta cuenta se accede a toda la información importante del proyecto y a las claves de acceso a la base de datos, que también se indican a continuación junto a las claves utilizadas en la aplicación para los usuarios ya registrados en la BD.

Database: **uocsurveys** User: **adminRdzkefu** Password: **PD6Xl2F5Fsba**

	User	Password
Administrador:	Admin	Admin
Consultor:	Consul	Consul
Estudiante:	Estud	Estud

B. Capturas de pantalla de la aplicación.

