	Máster Oficial en Software Libre
	Trabajo Final de Máster
	Memoria
	17/1/2016

Movilidad en la consulta de exámenes

Máster Universitario en Software Libre

Administración Web y Comercio Electrónico

Autor: Roberto Calvo Mendoza

Consultor: Francisco Javier Noguera Otero
Tutor externo: Andrea Contu

17-01-2016

RESUMEN DEL PROYECTO

El presente proyecto se desarrolla como trabajo fin de máster del programa de estudios de Máster Universitario en Software Libre de la UOC junto con la colaboración de la empresa Implemental Systems S.L.

El proyecto consiste en el análisis, diseño y desarrollo de una aplicación para dispositivos móviles con la que los estudiantes de un centro de estudios pueden autenticarse y visualizar la información relativa a sus próximos exámenes: fecha y lugar de realización con posicionamiento en el mapa. También muestra las calificaciones de los exámenes realizados más recientemente. Se incluye una aplicación del lado del servidor o *backend* con la que se comunicará los dispositivos móviles.

Además, se incorpora una aplicación web tradicional para la administración de este backend que permite a los administradores o profesores, introducir la información que podrán ver los estudiantes.

Todo el proyecto está desarrollado con tecnologías basadas en software libre. Para la parte de dispositivos móviles, se utiliza Apache Cordova que nos permite portar la aplicación a distintas plataformas sin necesidad de un desarrollo específico para cada una de ellas, manteniendo así unos costes bajos de desarrollo. Apache Cordova utiliza tecnologías web que son estándares ampliamente conocidos: HTML, CSS y JavaScrip, con lo que podemos utilizar librerías propias del desarrollo web, por ejemplo jQuery Mobile para el desarrollo de interfaces. Una vez desarrollada la aplicación, se compila para Android, IOS o Windows Phone, que son los sistemas operativos para móviles que dominan la mayor parte del mercado.

Técnicamente, la parte de dispositivos móviles se comunica con la parte del servidor mediante servicios web tipo RESTful. La información se transmite en formato JSON. Como servidor web de aplicaciones, se ha optado por usar Tomcat junto con el lenguaje de programación Java y el framework de desarrollo Spring. Como sistema gestor de base de datos se utiliza MySQL. También se hace uso de Google Maps para localizar los lugares de realización de los exámenes.

Durante el desarrollo se ha utilizado la plataforma Openshift creada por la empresa Red Hat. Nos ofrece un entorno de integración online con todas las herramientas más importantes hoy en día en el desarrollo de software libre.

En este proyecto se ha intentado aplicar el máximo de conocimientos adquiridos en las asignaturas del máster en una caso empresarial real .

TABLA DE CONTENIDOS

1 INTRODUCCIÓN

Este proyecto se realiza como finalización del programa de estudios del Máster Universitario en Software Libre de la UOC. Se trata de un proyecto profesional que se realiza en un empresa externa, en este caso Implemental Systems S.L.

El proyecto incluye el análisis, diseño y desarrollo de una aplicación para dispositivos móviles con la que los estudiantes de un centro de estudios pueden acceder a la información relativa a sus exámenes. También se incluye una aplicación web de administración. Todo el proyecto se ha realizado utilizando exclusivamente software libre.

No solo tiene como objetivo el desarrollo de la aplicación en sí misma, sino también demostrar todas las fases de desarrollo de aplicaciones usando software libre. En el capítulo 1 se especifican más detalladamente los objetivos así como la motivación para la selección de los tecnologías utilizadas. Entre las actividades realizadas está el estudio de viabilidad, incluido en el capítulo 2 de este proyecto, el análisis previo al desarrollo, incluido en el capítulo 3, y dentro de las tareas de diseño y desarrollo, se incluyen aspectos como la planificación del desarrollo y los planes de pruebas para asegurar la calidad del software.

1.1 Objetivos

Los objetivos principales del proyecto son:

- Desarrollar una aplicación para dispositivos móviles que permita a los estudiantes de un centro de estudios, la consulta de información relativa a sus exámenes.
- Desarrollar una aplicación web de administración, para la aplicación del punto anterior.
- Disponer de una aplicación móvil junto con su parte de administración, que sea extensible de forma que se puedan añadir nuevas funcionalidades en un futuro.

Como objetivos secundarios incluimos:

- Reducir los costes de desarrollo y mantenimiento en aplicaciones móviles. Esto se logra principalmente por el uso de Apache-Cordova[3], que nos permite realizar un solo desarrollo para distintas plataformas móviles.
- Ampliar los conocimientos en desarrollo de aplicaciones móviles multiplataforma. En este proyecto se utilizan tecnologías recientes, de las que no disponíamos de experiencia previa.
- Mejorar el conocimiento del ciclo de vida de un proyecto web. Se ha desarrollado el ciclo completo de vida del proyecto, pasando por todas las fases de un proyecto de software: estudio de viabilidad, análisis, diseño, desarrollo, implantación y mantenimiento .
- Poner en práctica el mayor número de conocimientos adquiridos durante la realización del máster.

1.2 Estado del arte

En los últimos años hemos asistido a un auge explosivo en el uso de dispositivos móviles. Muchas actividades que se realizan usando aplicaciones web a las que se accede con algún navegador, hoy en día se pueden realizar con apps instaladas en nuestros teléfonos. Dentro del mundo de los

dispositivos móviles, coexisten distintos sistemas operativos y distintas plataformas de distribución de software. Hoy en día, las dos plataformas mayoritarias son IOS[26] y Android[1]. Una aplicación que se haya desarrollado de forma nativa para IOS no es ejecutable en un dispositivo Android y viceversa. Esto es un problema, porque si el desarrollador quiere que su aplicación sea compatible con la mayor parte de móviles, debe realizar dos desarrollos distintos utilizando tecnologías distintas, lo cual duplica el trabajo tanto de aprendizaje como en la generación del código. Últimamente han aparecido tecnologías para solventarlo de forma que se genera el código una vez y se ejecuta en distintas plataformas. No es una solución exenta de problemas, ya que cada plataforma dispone de sus funcionalidades y no siempre están disponibles las mismas operaciones en ambos sistemas. Por ejemplo, los dispositivos Android disponen de tres botones principales, mientras que IOS solo tiene uno. También hay que tener en cuenta que al realizar un desarrollo único, la imagen de la aplicación es la misma en todos los casos, y habitualmente los interfaces de IOS o Android difieren.

Una de las soluciones que más aceptación ha tenido es Apache Cordova[3]. Ofrece un conjunto de APIs que permite acceder a las funciones nativas de los dispositivos móviles mediante aplicaciones desarrolladas utilizando las tecnologías web: HTML[25] , CSS[13], Javascript[33]. No es necesario escribir código en el lenguaje de programación nativo de cada plataforma como son Java para Android o Objective-C para Iphone. Una vez finalizado el desarrollo con Cordova, podemos empaquetar la aplicación para cada plataforma usando el SDK específico de cada una de ellas. Finalmente obtenemos una aplicación nativa que se puede instalar en el dispositivo.

Para el desarrollo de aplicaciones web, existen multitud de tecnologías maduras. En este proyecto se ha buscado el uso de tecnologías ampliamente demandadas en la industria del software. Se ha optado por crear una capa de persistencia mediante MySQL[45] y la comunicación con el dispositivo móvil se realiza mediante servicios web de tipo REST[61]. REST se compone de un conjunto de principios de arquitectura que están teniendo gran aceptación como alternativa a los servicios web SOAP. Para el desarrollo en el lado del servidor se ha usado como lenguaje de programación Java, ya que se trata de un lenguaje de programación muy desarrollado y demandado por la industria del software. Además dispone de multitud de frameworks y librerías dentro del mundo del software libre, en este proyecto se usan algunas de ellas.

Otra tecnología que ha surgido en los últimos años ha sido la computación en la nube o “cloud computing” donde cualquier recurso informático se ofrece como servicio accesible a través de la red. Es interesante la utilización de lo que se denomina PaaS “platform as a service”, que nos ofrece, de forma online todo lo necesario para el desarrollo del software. En concreto, se ha hecho uso de Openshift[52]. OpenShift nos ofrece una plataforma online para el desarrollo de aplicaciones basadas en software libre. Nos provee de servicios de hosting online donde poder realizar pruebas de integración, con sistema de control de versiones y herramientas de administración. Está creado por la empresa Red Hat[59] que es unos de los fabricantes de software libre más importantes. Openshift facilita enormemente la creación de servidores de pruebas o integración así como la instalación de sistemas de control de versiones y nos libra de las tareas de administración y mantenimiento.

1.3 Estructura de la memoria del proyecto

El proyecto se realiza siguiendo una metodología clásica en cascada en la que se suceden las distintas fases: estudio de viabilidad, análisis, diseño, desarrollo, implantación y mantenimiento . La estructura de este documento replica esas fases, existiendo un capítulo principal para cada una de ellas.

El último capítulo se dedica a las conclusiones, donde se resume el trabajo realizado, las impresiones personales obtenidas durante el desarrollo y el trabajo futuro a realizar con la aplicación.

Al final del documento, se incluye un capítulo de Bibliografía con todas las referencias utilizadas.

En un documento aparte, se han incluido los anexos, con información relativa a licencia, manual de usuario, resultados de pruebas, presupuesto, errores detectados y listado de imágenes.

2 ESTUDIO DE VIABILIDAD

La empresa Implemental Systems S.L ya ha desarrollado una aplicación móvil para uno de sus clientes que permite a los estudiantes de un centro educativo, visualizar la información relativa a sus exámenes. Esta aplicación se ha creado solamente para los dispositivos Android. Ahora se pretende adaptar la aplicación a distintas plataformas y que sea posible la integración en otros clientes similares. La aplicación pretende mostrar la siguiente información:

- Listado de los próximos exámenes que el estudiante debe realizar.
- Información específica de cada examen: fecha, hora, duración, lugar de realización, materiales necesarios.
- Posicionamiento sobre un mapa del lugar de realización del examen.
- Calificaciones de los últimos exámenes realizados.

Así mismo, se incluye la realización de una aplicación web de administración. Esta aplicación, permitirá a un administrador o profesor que se identifique mediante usuario/contraseña:

- Buscar asignaturas por nombre o código.
- Introducir toda la información relativa a los exámenes de una asignatura: fecha, hora, lugar, duración, materiales, dirección de realización.
- Ubicar la dirección en un mapa.
- Una vez realizados los exámenes, dispondrá de un listado de los alumnos matriculados en cada examen, y el profesor podrá introducir la calificación correspondiente a cada alumno.

Por una parte, esta aplicación debe estar disponible para el mayor número de plataformas móviles posibles, pero manteniendo el coste lo más bajo posible con el fin de asegurar su viabilidad económica.

Técnicamente, se deja libertad para la selección de las tecnologías a utilizar, siempre que se cumplan las condiciones anteriores.

Dentro del alcance del presente proyecto, se incluye el análisis, diseño y desarrollo de la aplicación móvil como la de administración, incluyendo el desarrollo del backend y su base de datos. No entra en el alcance la realización de la implantación, pero sí se incluye el soporte necesario. Tampoco se incluye la compra de equipos ni dominios o certificados de seguridad.

2.1 Estudio de la situación actual

El desarrollo actual presenta algunos problemas.

En primer lugar la aplicación solo funciona en plataformas Android. No existe para el resto de plataformas, por lo que no es útil para una parte importante de estudiantes.

La aplicación se desarrolló de forma exclusiva para un cliente. No es reutilizable para otros centros educativos.

La empresa desarrolladora, no dispone de un backend ni de una aplicación de gestión, ya que originalmente no se incluyó en el alcance del proyecto, la aplicación se comunicaba con los sistemas del cliente mediante unos servicios web tipo SOAP que fueron desarrollados por el propio cliente.

Requisitos del sistema

A continuación se describen los requisitos generales, posteriormente se describirán con mayor detalle en el capítulo de análisis:

Requisitos funcionales
Desarrollar una aplicación para móviles, multiplataforma que permita a un estudiante visualizar la información de sus próximos exámenes. Esta aplicación permitirá al estudiante identificarse mediante usuario y contraseña y visualizar un listado de sus próximos exámenes. Seleccionando uno de ellos, deberá poder ver la información relativa a su realización: fecha, lugar, hora, etc. También debe permitir localizar en un mapa el lugar de realización del mismo, ya que en centros educativos grandes es habitual realizar los exámenes en lugares distintos a aquellos en los que se imparte la docencia. También debe ser útil para localizar la realización de exámenes oficiales, por ejemplo exámenes de acceso a la universidad o de certificación de conocimientos de idiomas. Tras la realización de los exámenes permitirá consultar las calificaciones obtenidas por el alumno.

Desarrollar una aplicación web que sirva de administración. La aplicación será accesible desde un navegador web, y permitirá a un administrador o profesor editar la información indicada en el punto anterior.

Requisitos de implantación
La aplicación para dispositivos móviles, debe poder instalarse en la principales plataformas móviles disponibles actualmente.

Requisitos económicos
Se debe mantener tanto los costes como los plazos de entrega lo más bajo posible

Requisitos legales
Se debe permitir la libre distribución y edición del software.

2.2 Alternativas de la solución

Para resolver las necesidades planteadas con las interfaces para dispositivos móviles, se plantean tres posibles soluciones:

- Crear una aplicación nueva para cada plataforma: Android[1], Iphone[26] o Windows Phone[70]. Técnicamente consiste en replicar la misma aplicación, pero usando el lenguaje de programación propio de cada plataforma. Por ejemplo, para el caso de Android se usaría java, para Iphone se debería realizar otro desarrollo usando el lenguaje Swift[67] o Objective-C y para Windows Phone habría que utilizar la plataforma .Net[50].

- Realizar una aplicación web de forma tradicional, pero adaptando la presentación de los datos al tamaño de la pantallas de dispositivos móviles. Se utilizarían las tecnologías tradicionales de desarrollo web: HTML[25] , CSS[13], Javascript[33]. junto con cualquier lenguaje de desarrollo web: PHP[57], Java[27], .Net[50]. A esta aplicación se accedería utilizando un navegador web del móvil.

- Desarrollar una aplicación con el uso de alguna tecnología que permita realizar un desarrollo híbrido, por ejemplo Phonegap[54], Apache Cordova[3] o Appcelerator Titanium[2]. Para ello se utilizarían las tecnologías de desarrollo web: HTML5, CSS, JavaScript. A partir de ese desarrollo, se portaría a cada una de las plataformas móviles.

Para el desarrollo de la aplicación de administración y la parte del backend encargada de

interactuar con la base de datos, también se ofrecen varias alternativa de las más utilizadas hoy en día. En todos los casos se plantea el desarrollo de una aplicación web, teniendo como principal diferencia el lenguaje de programación. Los lenguajes de programación seleccionados son:

- PHP junto con el servidor de aplicaciones Apache Web Server[4]. PHP es un lenguaje de uso general interpretado, que fue diseñado para aplicaciones web. Es sencillo y requiere pocos recursos.
- Java, junto con el servidor de aplicaciones Apache Tomcat[7]. Java es un lenguaje multitarea, actualmente pertenece a la empresa Oracle. Ha tenido gran aceptación en el desarrollo web, siendo uno de los lenguajes dominantes. Dispone de una versión open source
- .NET (C sharp, o Visual Basic). Es la plataforma de desarrollo de la empresa Microsoft. No es software libre y requiere la instalación de sistema operativo Windows.

Para la parte de persistencia se considera montar un sistema gestor de bases de datos. Existen múltiples alternativas, por lo que se seleccionan algunas de las más utilizadas, de las cuales seleccionaremos una de ellas:

- PostgreSQL[58] es un sistema relacional, open source, desarrollado por la universidad de Berkeley, e incluye características de orientación a objetos y cumple con el estándar SQL92/SQL99.
- MySQL[45]/MariaDB[42] MySQL es otro gestor relacional y open source que ha adquirido gran popularidad en el desarrollo de aplicaciones para internet. MySQL fue adquirida por la empresa Oracle, y raíz de esa adquisición apareció el fork MariaDB con la intención de seguir siendo software libre.
- Oracle[53] es posiblemente el sistema gestor de bases de datos más utilizado a nivel mundial. Es relacional, con características de rendimiento muy altas y muchas herramientas potentes de administración. No es software libre, requiere licencias que pueden llegar a ser bastante cara.
- MongoDB[44]. Es un sistema de base de datos orientada a documentos. Se suele denominar NoSQL. No se trata de una base entidad-relacion, sino que almacena la información en documentos Json[39].

A continuación se analizan las alternativas atendiendo a sus ventajas e inconvenientes

Aplicación móvil

Aplicación móvil nativa para cada plataforma	
Ventajas:	El código estaría optimizado para cada plataforma y al ser un desarrollo nativo se espera un rendimiento superior.
Inconvenientes:	No disponemos de experiencia ni conocimientos en otras plataformas que no sea Android, los lenguajes de programación de algunas de ellas no son software libre, se disparan los gastos de mantenimiento y evolución del software.
Riesgos:	Cada plataforma pertenece a un fabricante distinto y alguno de ellos puede cambiar la estrategia de negocio, desapareciendo el soporte u obligando a una actualización.
Medidas paliativas:	Firma de contrato de soporte y/o actualizaciones

Aplicación web adaptable a móviles	
Ventajas:	Reutilización del conocimiento, se utilizarían tecnologías con las que contamos con amplia experiencia.
Inconvenientes:	Perdida de “experiencia del usuario”, para su utilización se requeriría el uso del navegador y no se podría instalar en el dispositivo móvil como una app más.
Riesgos:	Las modificaciones de los navegadores web de los dispositivos móviles pueden afectar a nuestra aplicación. cambiar la estrategia de negocio, desapareciendo el soporte u obligando a una actualización.
Medidas paliativas:	Realizar el desarrollo utilizando solo funcionalidades y métodos estándar recogidos por el World Wide Web Consortium (W3C).

Aplicación con Apache Cordova / Phonegap	
Ventajas:	Permite desarrollar utilizando tecnologías conocidas: HTML5, CSS, JavaScript y posteriormente portar la aplicación como una app para cada plataforma móvil. Se puede usar software libre sin necesidad de pagar ninguna licencia.
Inconvenientes:	Al no tratarse de un desarrollo nativo para cada aplicación se espera un rendimiento inferior al caso nativo
Riesgos:	Al ser software libre, podría abandonarse el proyecto o desaparecer el equipo de desarrollo.
Medidas paliativas:	Se puede contratar una empresa que de soporte al código existente.

Aplicación con Appcelerator Titanium	
Ventajas:	Permite desarrollar utilizando como lenguaje de programación JavaScript. Tienes funciones comunes para todas la plataformas
Inconvenientes:	Aunque permite compartir código para las distintas plataformas, pero para requiere escribir código específico para utilizar algunas funciones de cada plataforma.
Riesgos:	Al ser software libre, podría abandonarse el proyecto o desaparecer el equipo de desarrollo.
Medidas paliativas:	Se puede contratar una empresa que de soporte al código existente.

Aplicación Web y backend

PHP + Apache Web Server	
Ventajas:	Es un lenguaje open-source. Sencillo de utilizar. El hosting requiere pocos recursos de hardware y se puede subcontratar servicios de hosting para PHP muy baratos.
Inconvenientes:	Tiene una comunidad de usuarios más pequeña que otras plataformas,

	especialmente para proyectos empresariales grandes.
Riesgos:	Es software libre, podría abandonarse el proyecto. Si se quiere contratar un mantenimiento existe mayor dificultad para encontrar personal formado.
Medidas paliativas:	Contratar una formación en PHP

Java + Apache Tomcat	
Ventajas:	Dispone de muchos frameworks como Struts o Spring, librerías y herramientas creadas por terceros. Además tiene una comunidad de usuarios muy grande que facilita la resolución de cualquier problema.
Inconvenientes:	Requiere de servidores más potentes y consume más recursos que PHP. El desarrollo también es más pesado.
Riesgos:	La empresa propietaria, puede cambiar en un futuro el tipo de licencia o la estrategia de desarrollo
Medidas paliativas:	Se puede contratar empresas que den soporte con las versiones Open source

Plataforma .NET	
Ventajas:	Todas las herramientas relacionadas con el desarrollo de .NET están unificadas en un solo producto que facilita la instalación y configuración.
Inconvenientes:	Las licencias son de pago. La comunidad de usuarios es menor que en java y dificulta la resolución de problemas.
Riesgos:	El pago de licencias de la plataforma o sistemas operativos, puede superar el presupuesto del proyecto, cancelando su realización.
Medidas paliativas:	Ninguna. Al no tratarse de software libre 100% no se considera su utilización.

Base de datos

PostgreSQL	
Ventajas:	Es un sistema relacional, open-source que cumple con el estandar SQL y ofrece características de orientación a objetos.
Inconvenientes:	El rendimiento es inferior a otros sistemas como MySQL. Tampoco es tan popular como otros sistemas y no dispone de muchas herramientas de administración.
Riesgos:	Es software libre, podría abandonarse el desarrollo. Si en un futuro se desea migrar a otro sistema gestor de base de datos, pueden aparecer dificultades porque dispone de alguna características únicas que no son fácilmente trasladables a otros sistemas, como la herencia entre objetos.

Medidas paliativas:	Se podría contratar una empresa especializada en PostgreSQL. Se puede desarrollar la base de datos sin usar características especiales. Se puede

MySQL / MariaDB	
Ventajas:	Es un sistema relacional open-source, de gran aceptación en proyectos web y con comunidad de usuarios muy amplia. Tiene un gran rendimiento.
Inconvenientes:	Carece de algunas funcionalidades de otros sistemas.
Riesgos:	La empresa propietaria, puede cambiar en un futuro el tipo de licencia o la estrategia de desarrollo
Medidas paliativas:	Se puede utilizar el fork MariaDB, que sigue siendo software libre dirigido por la comunidad.

Oracle	
Ventajas:	Es uno de los sistemas relacionales de mayor rendimiento, con buenas herramientas de administración. Dispone de características como lenguaje de programación para el desarrollo de procedimientos. Es uno de los más utilizados.
Inconvenientes:	No es open-source y las licencias de uso son caras.
Riesgos:	El coste de las licencias desaconseja la realización del proyecto.
Medidas paliativas:	Ninguna. El coste y el hecho de no tratarse de software libre, la dejan fuera de consideración.

MongoDB	
Ventajas:	Es open-source. De gran crecimiento y aceptación en la comunidad de software libre.
Inconvenientes:	No se adapta bien a este proyecto, porque no es relacional. No disponemos de experiencia previa en su utilización
Riesgos:	Es software libre, podría abandonarse el desarrollo. Al no tener experiencia, es posible que no fuésemos capaces de adaptarla correctamente al proyecto.
Medidas paliativas:	Se podría contratar una empresa especializada para su desarrollo. También se podría contratar una formación específica en MongoDB

2.3 Selección de la solución

Los principales factores para seleccionar una solución han sido:

- Mantener los costes y tiempos de desarrollo lo más bajos posibles cumpliendo con los requisitos funcionales.
- Utilizar tecnologías de uso generalizado por las empresas de forma que sea sencillo contratar empresas que realicen el mantenimiento o resolver incidencias a través de internet.

- Permitir que la aplicación para dispositivos móviles estuviese disponible en distintas plataformas.
- Permitir la libre distribución y modificación del código de la aplicación.

A continuación se muestra gráficos comparativos de las distintas tecnologías preseleccionadas. Los gráficas están generadas por Google Trends[21] y nos dan una idea de la aceptación de cada tecnología.

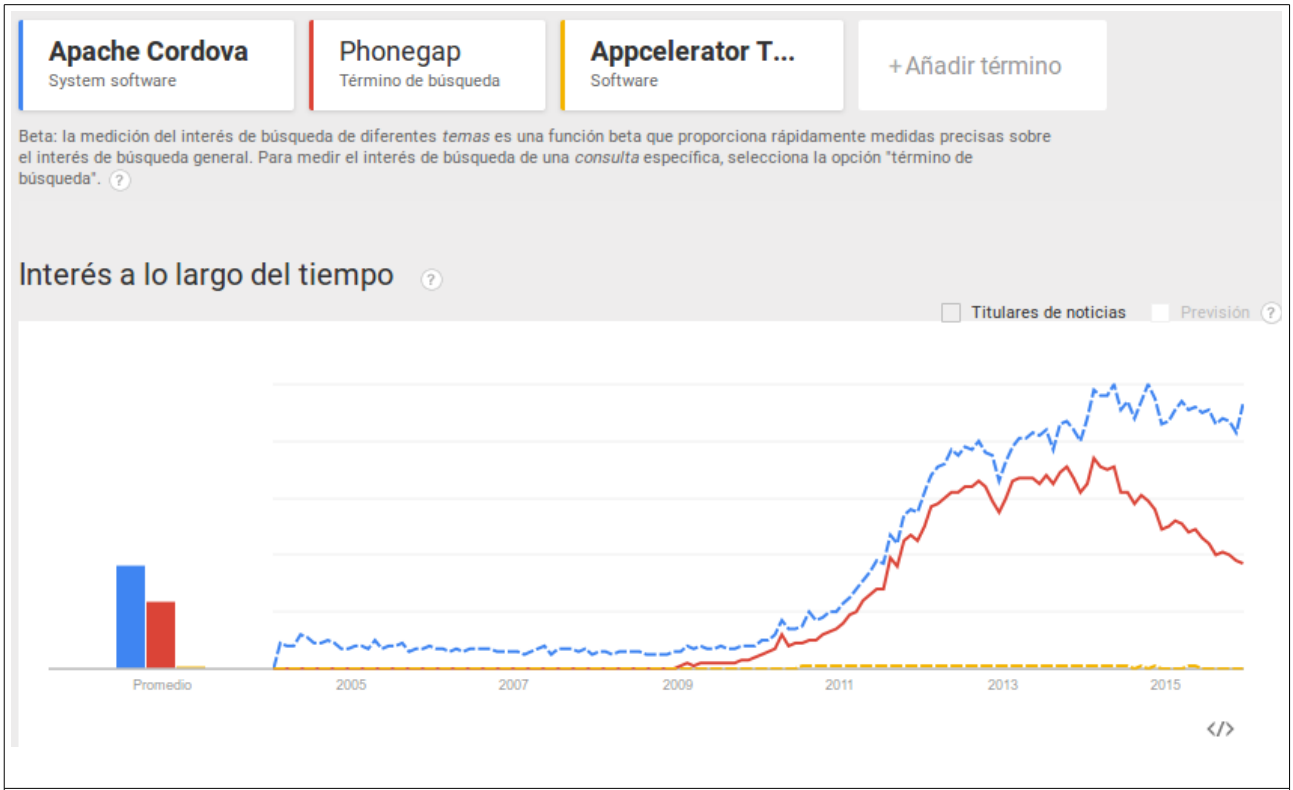


Fig 2.1 Comparativa tecnologías de desarrollo híbrido para móviles (por Google Trends[21])

Fig 2.2 Comparativa lenguajes de programación web (por Google Trends[21])

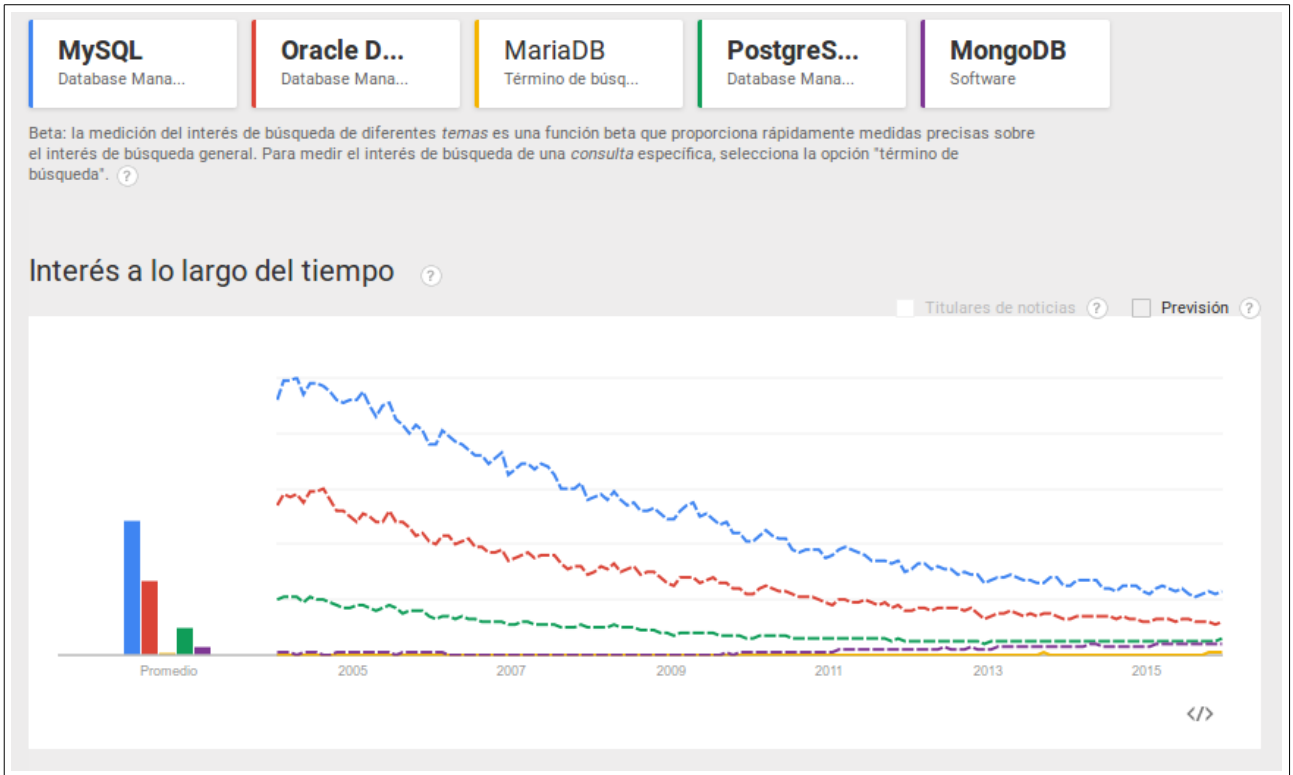


Fig 2.3 Comparativa sistema gestores de bases de datos (por Google Trends[21])

Con esos factores, la solución pasa por el uso de software libre, además, para permitir que la aplicación móvil sea ejecutable en distintas plataformas sin perder la experiencia de usuario de las aplicaciones nativas, hemos decidido utilizar **Apache Cordova** para su desarrollo.

Para el backend, la selección de la propuesta se ha realizado buscando las tecnologías más utilizadas hoy en día y de las que dispusiésemos de experiencia y conocimientos para realizar un desarrollo de forma eficiente. En base a esto, se ha seleccionado **MySQL** como sistema gestor de bases de datos y **Java** como lenguaje de programación de la aplicación. Como sistema operativo para el Backend se utilizará **Linux**.

3 ANÁLISIS

3.1 Definición del sistema

Como se ha explicado anteriormente, este proyecto se compone por una parte de una aplicación para dispositivos móviles desarrollada con Cordova que permite a un estudiante visualizar información relativa a sus exámenes. Esta aplicación se conectará al servidor a través de servicios web tipo REST, con los que intercambiará información. La capa de persistencia se materializa con una base de datos MySQL. Por otra parte, existe una aplicación web, desarrollada en java, que permite las labores de administración. Se seguirán todas la normas y estándares de desarrollo web, utilizando frameworks y librerías de uso generalizado en programación web.

A continuación se especifica detalladamente los requisitos que debe cumplir la aplicación.

3.2 Requisitos

Requisitos funcionales

Los siguientes requisitos funcionales deberán estar disponibles desde dispositivos móviles:

RQF101	El sistema permitirá al alumno o profesor identificarse como usuario de la aplicación mediante usuario y contraseña.
RQF102	Los alumnos podrán seleccionar mediante un menú si acceden a los próximos exámenes o ven sus últimas calificaciones.
RQF103	El profesor podrá ver un listado de los próximos exámenes a los que está asignado.
RQF104	El alumno o profesor, podrá filtrar el listado de exámenes según un campo de búsqueda que buscará entre los resultados del listado.
RQF105	El alumno o profesor, al seleccionar un examen del listado, podrá visualizar la información detallada del mismo.
RQF106	El alumno o profesor, podrá visualizar en un mapa el lugar de realización del examen.
RQF107	El alumno, podrá ver las calificaciones de sus últimos exámenes.
RQF108	El alumno y profesor, podrán en todo momento desautenticarse de la aplicación, volviendo a mostrar la pantalla de login y siendo necesario introducir otra vez la contraseña para acceder.

Los siguientes requisitos funcionales deberán estar disponibles desde la aplicación web accesible con un navegador:

RQF201	El sistema permitirá al profesor o administrador, acceder a la aplicación web de administración tras identificarse con su usuario y contraseña.
RQF202	La aplicación permitirá al administrador o profesor, realizar una búsqueda de asignaturas contra la base de datos mediante el nombre o código de asignatura y visualizar un listado con el resultado de la búsqueda.
RQF203	El administrador, tras seleccionar una asignatura podrá introducir o modificar los datos de los exámenes: fecha, hora, lugar, dirección y posición en el mapa.

RQF204	El profesor podrá ver los detalles del examen pero no modificarlos.
RQF205	El sistema permitirá al administrador o profesor, visualizar en un mapa la dirección de realización del examen.
RQF206	Tras la realización del examen, la aplicación web permitirá al profesor, ver un listado de los alumnos convocados al examen e introducir sus calificaciones
RQF207	El administrador y profesor, podrán en todo momento desautenticarse de la aplicación, volviendo a mostrar la pantalla de login y siendo necesario introducir otra vez la contraseña para acceder.

Requisitos de implantación

RQI101	La aplicación móvil deberá ser portable a las distintas plataformas móviles.
RQI102	La aplicación web, deberá adaptarse a distintas pantallas, por ejemplo tablets.

Requisitos económicos

RQE101	Se deberán mantener los costes de desarrollo lo más bajo posible.
RQE102	Se deberán mantener el plazo de entrega lo más corte posible.

Requisitos legales

RQL101	Se deberá escoger una licencia de software libre que permita la libre distribución y modificación del software

3.3 Perfiles de usuario

A partir de los requisitos funcionales descritos anteriormente, podemos definir los siguientes perfiles de usuario que utilizarán la aplicación.:

- **Alumno**. Es la persona a la que va dirigida la aplicación móvil. Sólo tiene acceso a la aplicación móvil y solo podrá ver la información relativa a sus exámenes.

- **Profesor**. Tendrá acceso tanto a la aplicación móvil cómo a la de administración. En la aplicación móvil podrá ver la información relativa a los exámenes de sus asignaturas. Desde la aplicación de administración podrá visualizar la información y tendrá permisos para introducir las calificaciones.

- Administrador. Tendrá acceso a la aplicación de administración y podrá introducir la información relativa a los exámenes, excepto las calificaciones, que es tarea del profesor responsable de la asignatura.

3.4 Casos de uso

Los siguiente casos de uso hacen referencia a la aplicación para los dispositivos móviles.

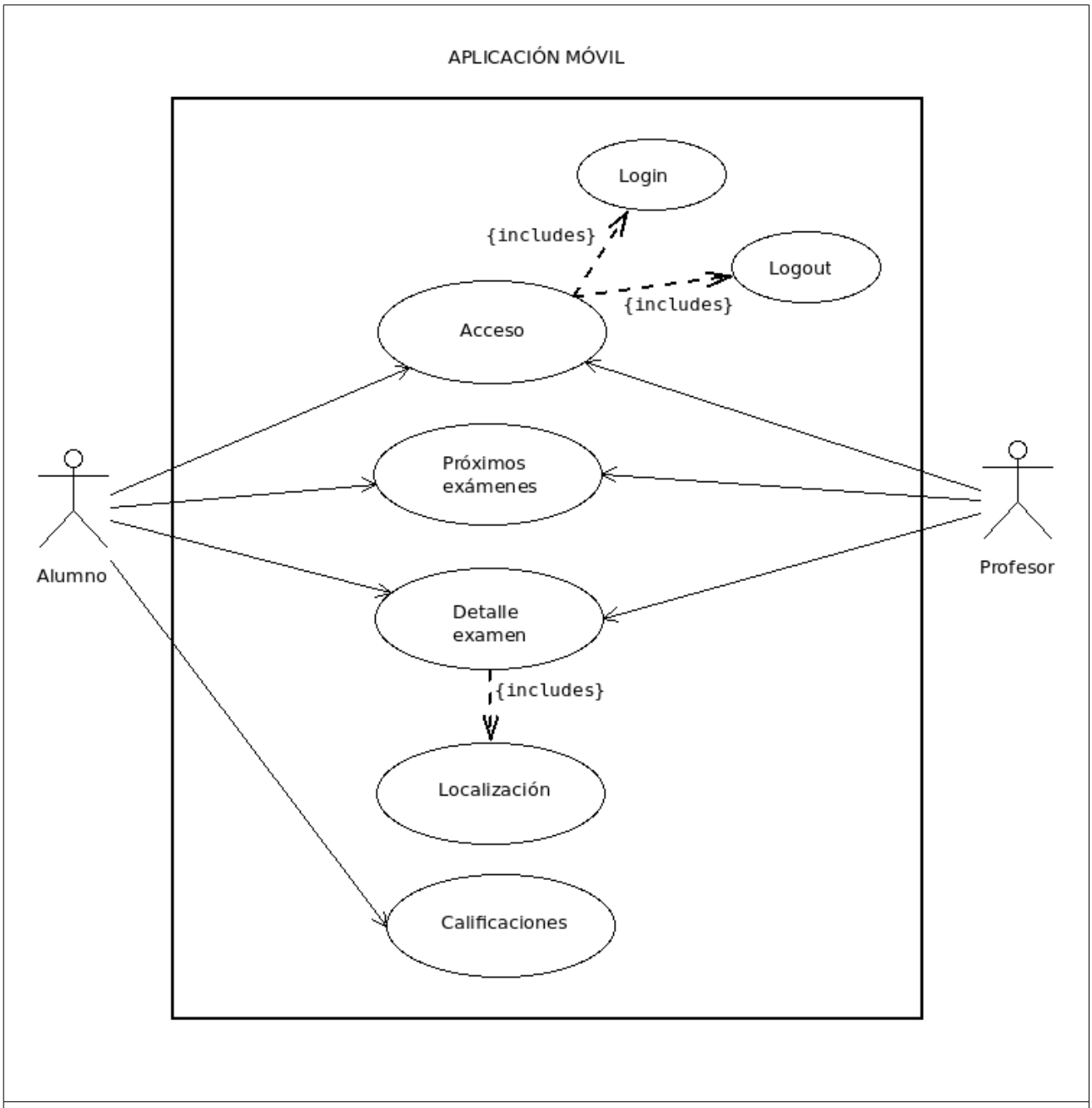


Fig. 3.1 Casos de uso de la aplicación móvil

Gráfico generado con Dia: http://dia-installer.de/

CU101.1 Acceso-Login	
Actores:	Alumno, profesor
Descripción:	Representa la operación de hacer login en la aplicación del dispositivo móvil, validando que el usuario y contraseña sean correctos, si tiene perfil de estudiante se le mostrará una pantalla para que escoja si quiere ver un listado de los próximos exámenes o las calificaciones de los últimos exámenes. Si es profesor, le llevará directamente al listado de los próximos exámenes. Se corresponde con el requisito funcional
Requisitos	RQF101, RQF102.

funcionales:	

CU101.2 Acceso-Logout	
Actores:	Alumno, profesor
Descripción:	Representa la operación de hacer logout en la aplicación del dispositivo móvil. Se volverá a mostrar la pantalla de login inicial y se borrará cualquier dato relativo a la sesión en el servidor. Si el usuario quiere volver a acceder a cualquier operación, deberá volver a hacer login.
Requisitos funcionales:	RQF108

CU102 Listado de exámenes	
Actores:	Alumno, profesor
Descripción:	Muestra en la pantalla del dispositivo móvil un listado con los próximo exámenes a los que está convocado el usuario. Los resultados pueden ser filtrados a través de un campo de texto editable.
Requisitos funcionales:	RQF103, RQF104

CU103 Detalle de exámenes	
Actores:	Alumno, profesor
Descripción:	Muestra el detalle de un examen en la pantalla de un dispositivo móvil.
Requisitos funcionales:	RQF105

CU104 Localización del examen.	
Actores:	Alumno, profesor
Descripción:	Permite visualizar sobre un mapa en el dispositivo móvil, la dirección donde se realiza el examen.
Requisitos funcionales:	RQF106

CU105 Últimas calificaciones	
Actores:	Alumno.
Descripción:	Permite al alumno autenticado visualizar en el dispositivo móvil las calificaciones de sus últimos exámenes
Requisitos funcionales:	RQF107

Los siguiente casos de uso hacen referencia a la aplicación web de administración:

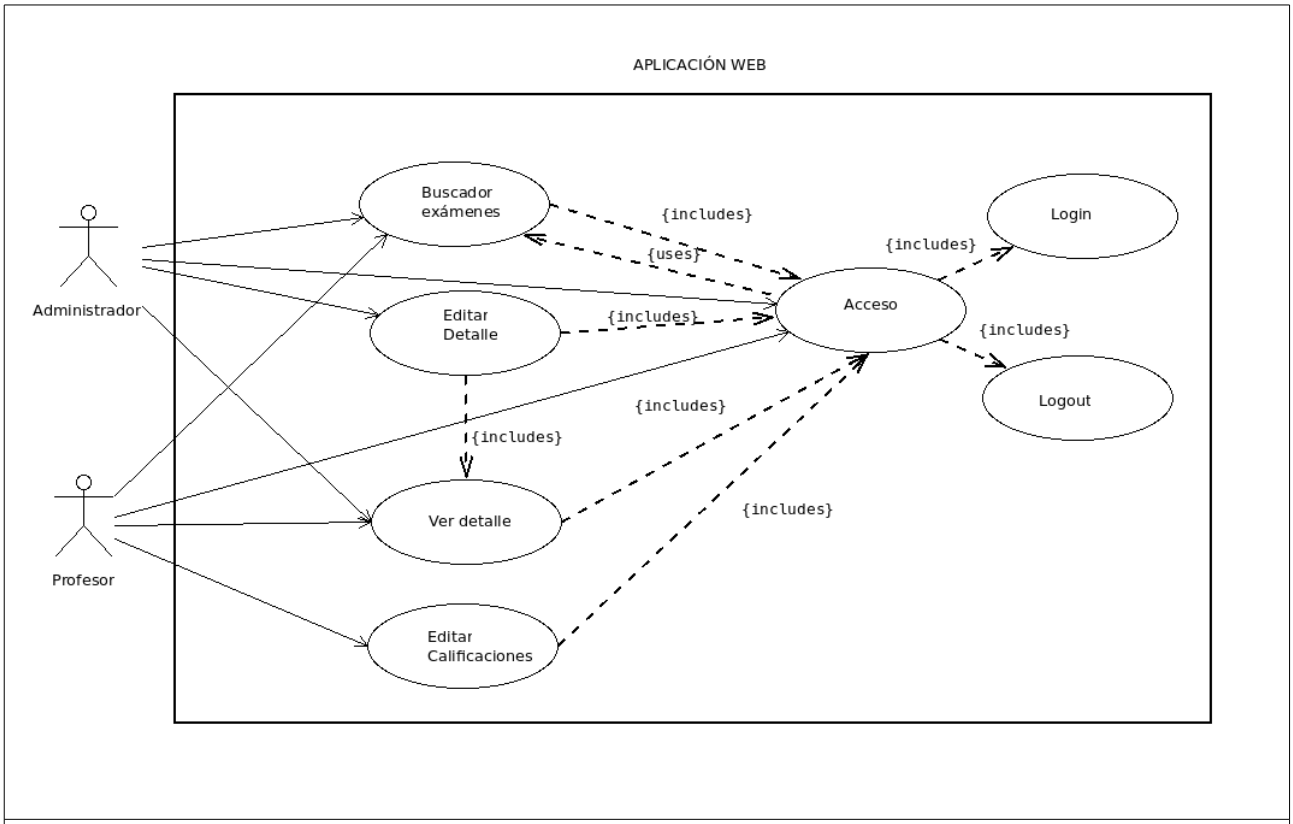


Fig. 3.2 Casos de uso de la aplicación web de administración

Gráfico generado con Dia: http://dia-installer.de/

CU201.1 Acceso-Login	
Actores:	Administrador, profesor
Descripción:	Representa la operación de hacer login en la aplicación web de administración, validando que el usuario y contraseña sean correctos. Permite el acceso a los usuarios que tengan perfil de administrador o profesor
Requisitos funcionales:	RQF201

CU201.2 Acceso-Logout	
Actores:	Administrador, profesor
Descripción:	Representa la operación de hacer logout en la aplicación web de administración, cerrando la sesión del servidor web. A partir de ese momento, si intenta acceder a cualquier pantalla, volverá a mostrarse la pantalla de login.
Requisitos funcionales:	RQF207

CU202 Listado de exámenes	
Actores:	Administrador, profesor
Descripción:	Permite a un usuario autenticado, con perfil de administrador o profesor, obtener un listado de próximos exámenes de acuerdo a un criterio de búsqueda. El administrador podrá visualizar todos los exámenes, mientras que el profesor solo verá en la lista de aquellos exámenes en los que está asignado.
Requisitos funcionales:	RQF202

CU203 Editar detalle examen	
Actores:	Administrador
Descripción:	Permite a un usuario autenticado, introducir los detalles de un examen: fecha, hora, lugar, dirección. La dirección se visualizará sobre un mapa en la misma pantalla.
Requisitos funcionales:	RQF203

CU204 Ver detalle examen	
Actores:	Administrador, profesor
Descripción:	Permite a un usuario autenticado, visualizar los detalles de un examen: fecha, hora, lugar, dirección. La dirección se visualizará sobre un mapa en la misma pantalla.
Requisitos funcionales:	RQF204 y RQF205

CU205 Editar calificaciones	
Actores:	Profesor
Descripción:	Permite a un usuario autenticado, ver el listado de alumnos convocados a un examen e introducir o modificar la calificación del mismo.
Requisitos funcionales:	RQF206

La siguiente tabla muestra relación entre los casos de uso y los perfiles que pueden ejecutar ese caso de uso:

	Alumno	**Profesor**	**Administrador**
CU101.1	X	X	
CU101.2	X	X	
CU102	X	X	
CU103	X	X	
CU104	X	X	

CU105	X		
CU201.1		X	X
CU201.2		X	X
CU202		X	X
CU203			X
CU204		X	X
CU205		X	

3.5 Interfaces de usuario

A continuación se muestran el diseño de los interfaces que deberá tener la aplicación:

Interfaces para dispositivos móviles

Fig. 3.3
Pantalla de Login aplicación móvil
CU101

Fig 3.4
Pantalla de selección de información.
CU101

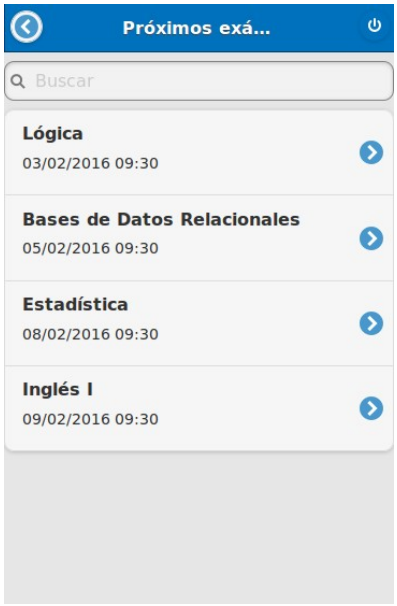


Fig. 3.5
Listado de exámenes. CU102

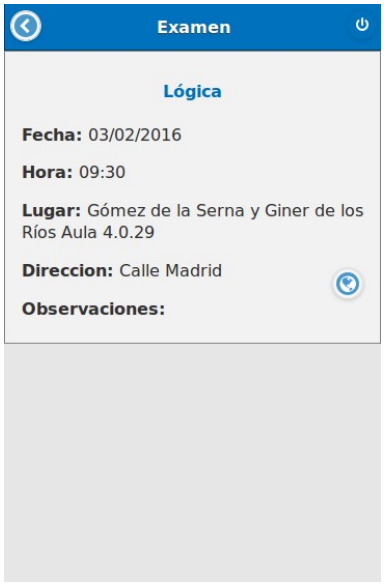


Fig. 3.6
Detalle de examen CU103

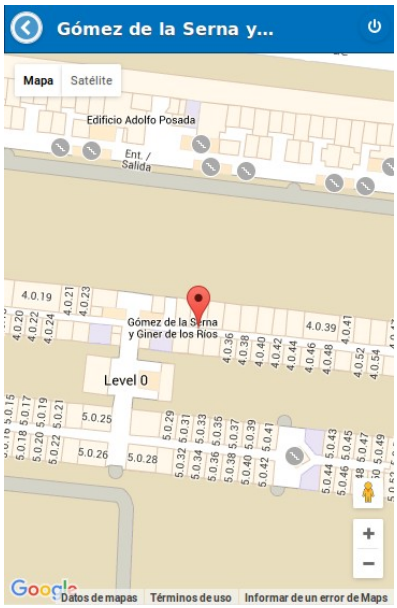


Fig. 3.7
Localizción del examen CU104

Fig 3.8
Últimas calificaciones CU105

Interfaces para aplicación web

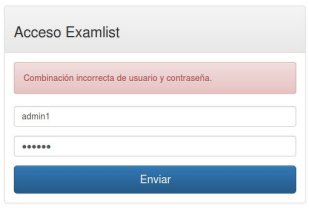


Fig. 3.9
Login aplicación web. CU201

Fig. 3.10
Listado de exámenes CU202

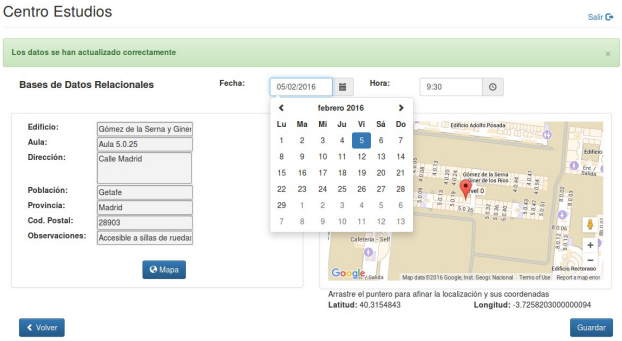


Fig. 3.11
Editar detalle de examen CU203

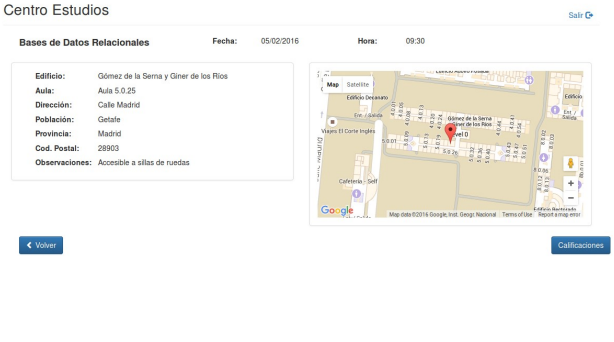


Fig. 3.12
Ver detalle de examen CU204

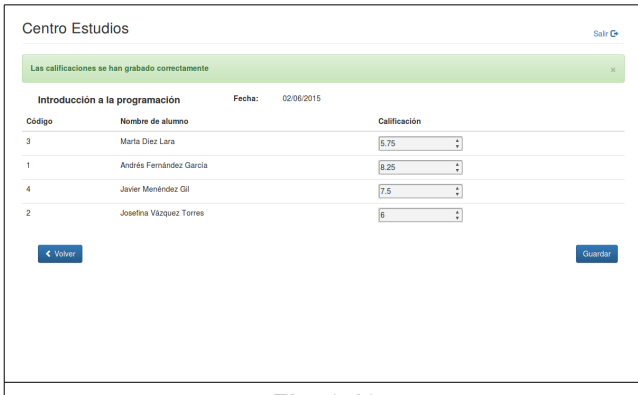


Fig. 3.13
Editar calificaciones CU205

3.6 Plan de pruebas

Pruebas unitarias

Se realizarán durante el desarrollo. Se prevé la realización de pruebas unitarias para los métodos encargados de recuperar la información de BBDD. Cada consulta a base de datos estará en un método independiente sobre el que aplicaremos una prueba unitaria. Con esta prueba, comprobaremos por una parte que las consultas con correctas y por otra que el tiempo de respuesta es aceptable, las pruebas se automatizarán para poder repetirlas siempre que se considere oportuno, para ello se usará alguna herramienta de pruebas unitarias como **Junit**[40] .

Pruebas de integración

Se realizarán distintas pruebas de integración para cada caso de uso, que podrán realizarse manualmente o programarlas con alguna herramienta como por ejemplo **Selenium**[62].

CU101.1 – Acceso Login aplicación móvil	
PU101.1-1	Se probará que para un usuario con perfil de alumno, el sistema valida correctamente la contraseña y le muestra la pantalla para seleccionar los próximos exámenes o las últimas calificaciones.
PU101.1-2	Se probará que para un usuario con perfil de profesor, el sistema valida correctamente la contraseña y le muestra la pantalla con el listado de exámenes.
PU101.1-3	Para cualquier tipo de usuario, si la contraseña o el usuario son incorrectos, no se debe permitir acceder y se mostrará un mensaje de error.

CU101.2 – Acceso Logout aplicación móvil	
PU101.2-1	Se probará que para un usuario autenticado, al pulsar sobre el botón de logout desde cualquier punto de la aplicación, se muestra la pantalla de login y en la base de datos del servidor, se borran la información de la sesión.

CU102 - Listado de exámenes en aplicación móvil	
PU102-1	Se probará que tanto para el alumno como el profesor se le muestran únicamente sus exámenes. Se comprobará que la información se corresponde con la existente en la BBDD.
PU102-1	Si no dispone de exámenes, de debe mostrar un mensaje indicativo.

CU103 – Detalle de examen en aplicación móvil	
PU103-1	Se probará que tanto para el alumno como el profesor se muestra el detalle del examen: fecha, hora, lugar de realización, dirección. Se comprobará que la información se corresponde con la existente en la BBDD. Deberá mostrarse un botón para acceder a la localización sobre el mapa del lugar de realización del examen.

CU104 – Localización del examen en aplicación móvil	
PU104-1	Se probará que al pulsar el botón de localización en el mapa del lugar del examen, se

	abre una pantalla mostrando la posición sobre Google Maps. Se probará que el mapa se adapte a distintas tamaños de pantalla y distintas orientaciones del dispositivo, cambiando entre horizontal y vertical.

CU105 – Últimas calificaciones en aplicación móvil	
PU105-1	Se probará que tanto para al alumno se le muestran únicamente las calificaciones de sus exámenes. Se comprobará que la información se corresponde con la existente en la BBDD.
PU105-1	Si no dispone de calificaciones, de debe mostrar un mensaje indicativo

CU201.1 – Acceso Login aplicación web de administración	
PU201.1-1	Se probará que para un usuario con perfil de profesor o administrador el sistema valida correctamente la contraseña y se accede a la pantalla del buscador de asignaturas.
PU201.1-2	Para cualquier tipo de usuario, si la contraseña o el usuario son incorrectos, no se debe permitir acceder y se mostrará un mensaje de error.

CU202.2 – Acceso Logout aplicación web de administración	
PU201.2-1	Se probará que en cualquier punto de la aplicación, un usuario autenticada podrá cerrar la sesión y se le mostrará la pantalla de login.
PU201.2-2	Se comprobará que tras hacer logout, si intenta acceder a otra pantalla de la aplicación, le aparece la pantalla de login.

CU202 - Listado de exámenes en aplicación web	
PU202-1	Se probará que introduciendo unos criterios de búsqueda de asignaturas el sistema devuelve un listado con los criterios introducidos. Se comprobará que la información se corresponde con la existente en la BBDD.
PU202-2	Si se trata de un perfil de profesor, solo podrá visualizar sus asignaturas. El administrador podrá visualizar todas asignaturas que se correspondan con los criterios de búsqueda.
PU202-3	Si no hay resultados que se correspondan con los criterios de búsqueda, de debe mostrar un mensaje indicativo.

CU203 – Editar detalle de examen en aplicación web	
PU203-1	Se probará que el administrador puede introducir y modificar el detalle del examen: fecha, hora, lugar de realización, dirección y mapa con la localización del examen. Se comprobará que la información se graba en la BBDD y se muestra la pantalla con los datos actualizados.

CU204 – Ver detalle de examen en aplicación web	
PU204-1	Se probará que el perfil de profesor puede visualizar la información del detalle del examen: fecha, hora, lugar de realización, dirección y mapa con la localización del examen.
PU204-2	Se probará que si todavía no se han introducido los detalles de un examen, se muestra un mensaje indicativo.

CU205 – Editar calificaciones en aplicación web	
PU205-1	Se probará que el perfil de profesor visualizar el listado de los alumnos asignados a un examen y puede modificar las asignaturas, quedando grabadas en la BBDD
PU205-2	Se probará que no se puede introducir las calificaciones antes de que el examen se haya realizado.

Pruebas de sistema

Se realizarán pruebas de carga sobre el sistema. Se probará que el sistema es capaz de soportar al menos 50 usuarios simultáneos, simulando para cada usuario una navegación completa por la aplicación móvil en un tiempo no superior a 5 segundos. Cada petición al servidor debe ser resuelta en un tiempo inferior a un segundo. Para realizar estas pruebas, se utilizará herramientas open-source, p.ej. **Jmeter**[5].

Pruebas de implantación

Se planificarán las siguientes pruebas en la fase de implantación:

- Volcado y recuperación de la BBDD, comprobando que no afecta al funcionamiento de la aplicación.
- Restauración de la aplicación alojada en el servidor. Deberá eliminarse y volver a instalar la aplicación web comprobando que sigue funcionando correctamente.
- Reinicio del servidor. Se realizará una parada y arranque del servidor comprobando el funcionamiento vuelve a ser correcto.

Pruebas de aceptación

Las pruebas de aceptación, deben ser realizadas por personal ajeno al desarrollo. Debe probarse la aplicación en su totalidad, comprobando que se cumple con los requisitos planteados. Lo ideal es que las realice el usuario o cliente final. En este caso, al tratarse de un proyecto par el trabajo fin de máster, al final del desarrollo se realizan unas pruebas completas para simular las pruebas de aceptación.

4 DISEÑO

4.1 Arquitectura

El proyecto se va a diseñar siguiendo una arquitectura multicapa de tres capas. Por una parte, existirá una capa de presentación encargada de mostrar los datos al usuario y permitir la interacción con el mismo. A continuación se encuentra la capa de lógica de negocio encargada de procesar las peticiones del usuario, realizar las operaciones lógicas, solicitar los datos a la capa de persistencias y devolver una respuesta al usuario. Por último, tenemos la capa de persistencia, encargada del almacenamiento físico de los datos.

Además, en este proyecto, la capa de presentación se divide en dos partes independientes entre sí. La parte de dispositivos móviles está dirigida a los alumnos del centro de estudios, que accederán vía dispositivos móviles. La aplicación web de administración solo es accesible por los administradores y profesores. Se trata de una aplicación web diseñada para ser usada desde un navegador.

Esquemáticamente, se representa en la figura 4.1. mediante el siguiente diagrama UML[68] de despliegue:

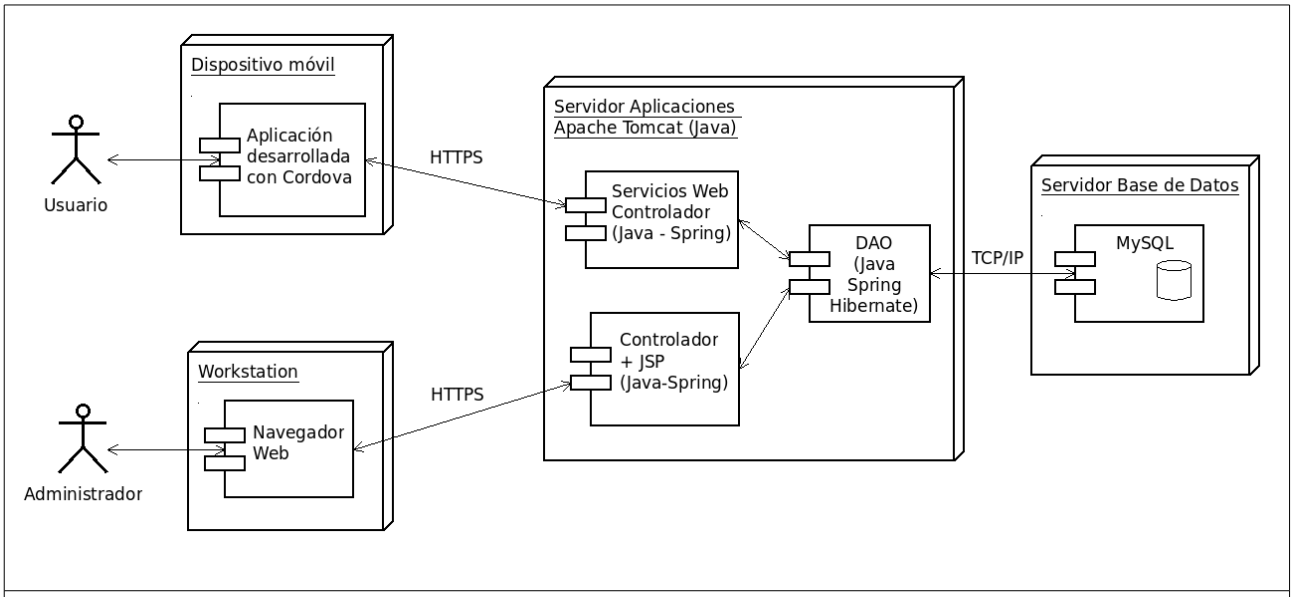


Fig. 4.1 Diagrama de despliegue

Gráfico generado con Dia: http://dia-installer.de/

Aplicación móvil Cordova

La capa de presentación de los dispositivos móviles, se generará mediante Apache Cordova[3]. Apache Cordova originalmente fue un proyecto que compró Adobe Systems y que denominó PhoneGap[54]. Posteriormente liberó una versión de código abierto a la fundación de software Apache, el proyecto resultante pasó a llamarse Apache Cordova. Phonegap sigue existiendo como una distribución de Apache Cordova que añade funcionalidades relacionadas con los servicios de Adobe, como la compilación de aplicaciones en la nube [56].

Apache Cordova permite crear aplicaciones para dispositivos móviles utilizando tecnologías web: **HTML**, **CSS** y **JavaScript**. La aplicación así creada, se compila con Cordova para distintas plataformas móviles de destino: Android, iOS o Windows y de esta manera podrá ser publicada e instalada en los dispositivos móviles.

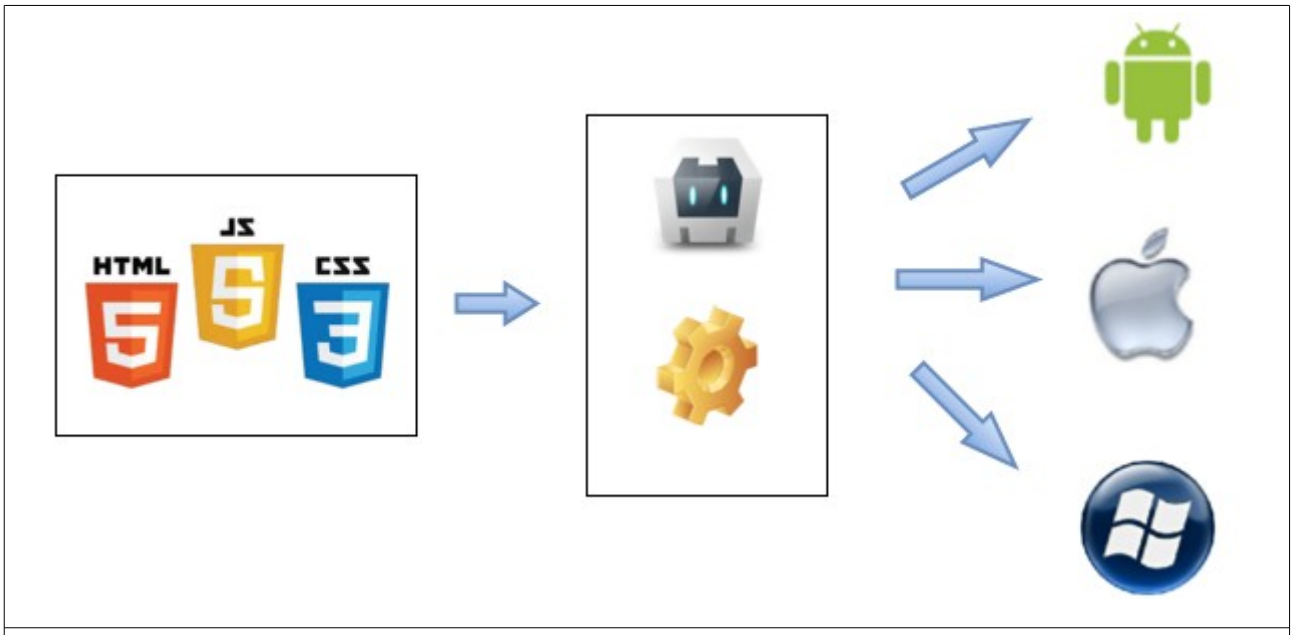


Fig. 4.2 Funcionamiento Cordova

Gráfico generado con: https://www.draw.io/

La aplicación resultante se denomina híbrida porque no se trata de una aplicación nativa para el dispositivo sino que utiliza un motor de renderizado web para mostrar los interfaces, pero tampoco se trata de una aplicación web pura, ya que está compilada para cada tipo de móvil y permite utilizar APIs nativas del dispositivo mediante el uso de plugins[55] .

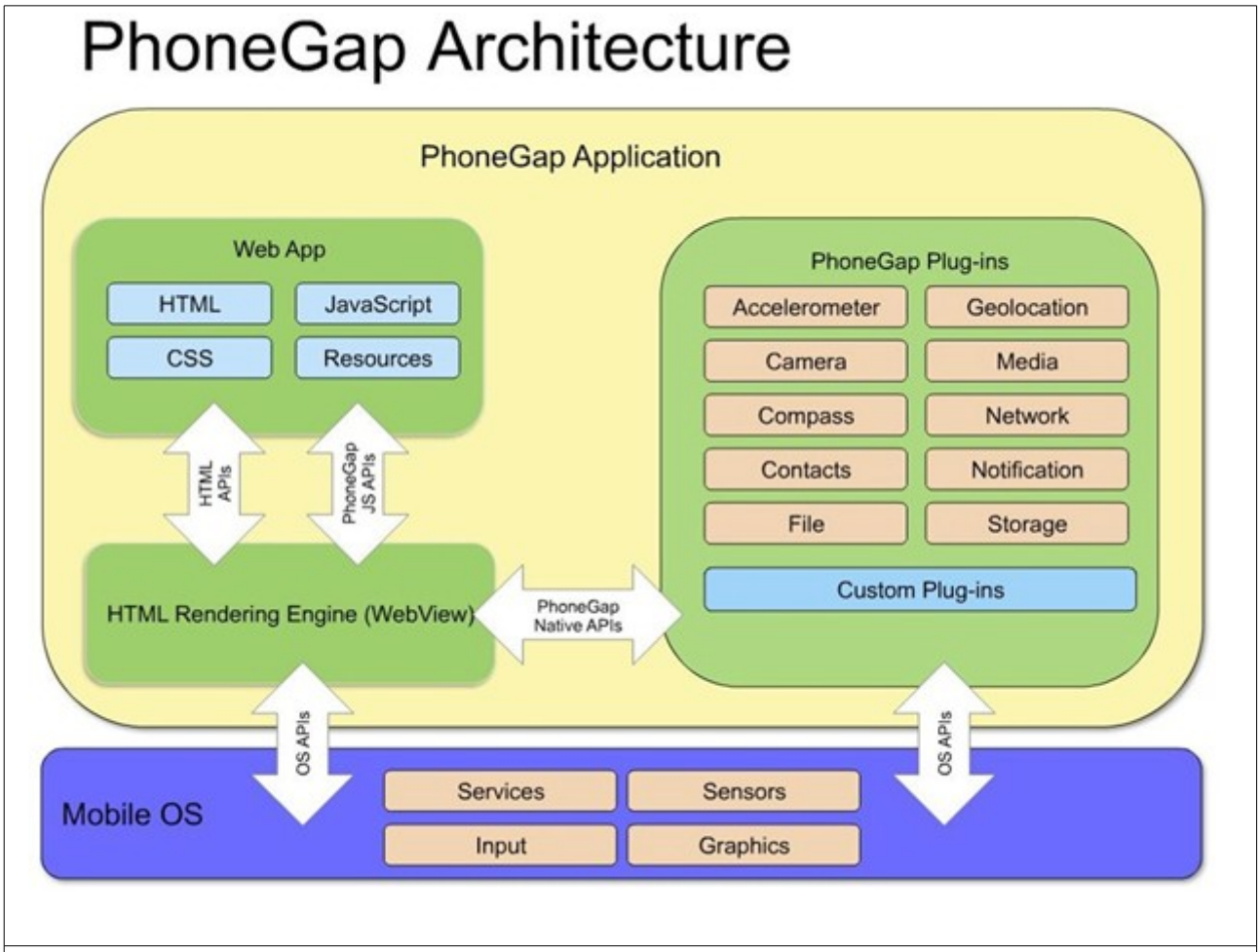


Fig. 4.3 Arquitectura Phonegap/Cordova

Ref.: http://mobile-design-framework.wikispaces.asu.edu/

El funcionamiento básico de nuestra aplicación consistirá en unos interfaces móviles que recuperan información del servidor y la muestran adaptada al dispositivo. La aplicación móvil se comunicará con la capa de lógica de negocio mediante servicios web tipo **Restful**, usando protocolo Https y recibiendo las respuestas en formato **Json**.

Fig. 4.4 Esquema comunicación

Gráfico generado con: https://www.draw.io/

Aplicación java en el servidor

La capa de presentación de la aplicación web de administración, se desarrollará mediante la tecnología **JSP** (Java Server Pages) que estarán alojadas en el servidor de aplicaciones **Apache Tomcat**. Para la utilización de esta aplicación, el usuario accede mediante un navegador web.

La capa de lógica de negocio se realizará mediante una aplicación web escrita en java y ejecutándose en un servidor de aplicaciones Apache Tomcat. Recibirá tanto las peticiones de la

aplicación para dispositivos móviles como las provenientes del navegador para la aplicación web de administración. Al tratarse de una capa común tanto para la aplicación web como para la aplicación web de administración, nos permite reutilizar componentes. Esta capa se desarrollará usando la tecnología **Java EE** en combinación con el framework **Spring**.

El framework Spring se compone de varios módulos que facilitan el desarrollo de todo tipo de aplicaciones, no solo web. En nuestro caso, además del core de Spring, que nos permite configurar la aplicación, hay que destacar la utilización del módulo MVC (modelo-vista-controlador) diseñado para el desarrollo de aplicaciones y servicios web; y el módulo el módulo de acceso a bases de datos y gestión de transacciones, que utilizaremos en combinación con Hibernate como ORM (Object-relational mapping).

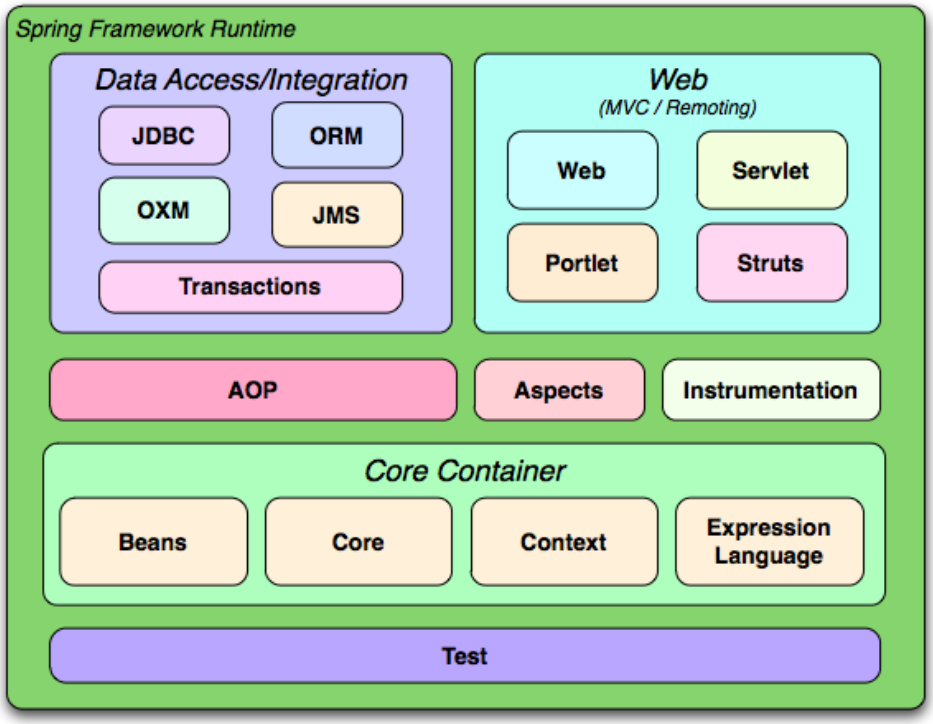


Fig. 4.5 Módulos de Spring

Ref: http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html

Utilizaremos la versión de **Hibernate**[22] **JPA** (Java Persistence Api), se trata de un framework de ORM (Object-relational mapping) con el que traducimos el modelo entidad-relación de una base de datos a un modelo orientado a objetos. A través de Hibernate, nos comunicaremos con la capa de persistencia de forma que no realizaremos consultas SQL directamente sobre la base de datos, sino que las consultas se realizarán usando el modelo orientado a objetos y es éste el encargado de interactuar con la capa de persistencia. La capa de persistencia se realizará mediante la instalación de un sistema gestor de bases de datos MySQL donde implementaremos un modelo entidad-relación.

Aunque el framework Spring junto con Hibernate son frameworks complejos en su interior, supone una gran facilidad a la hora de desarrollar, ya que simplifica la generación de código, permitiendo al desarrollador centrarse en lo temas de negocio. En un diagrama UML de secuencia, podemos representar las peticiones del usuario de la siguiente manera:

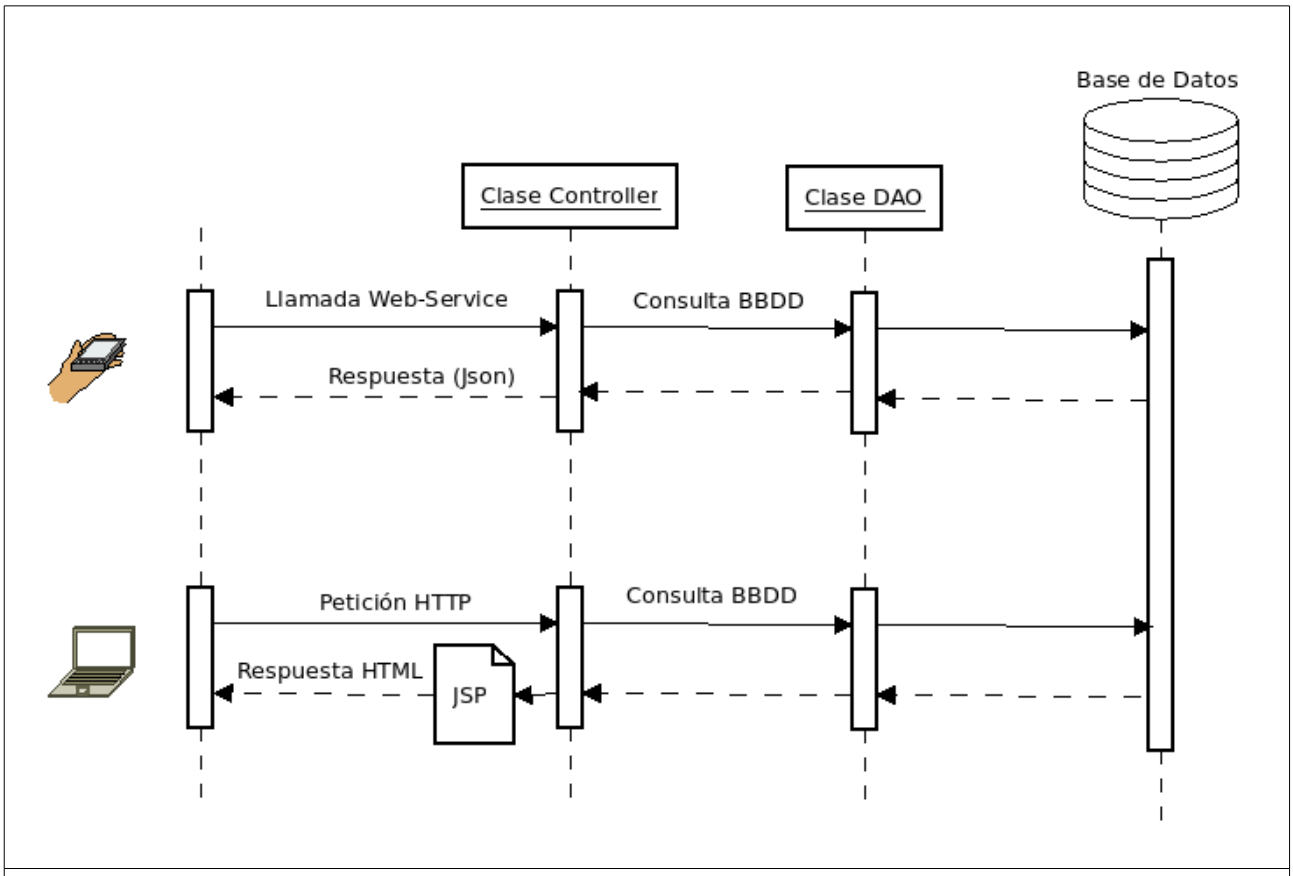


Fig. 4.6 Diagrama de secuencia

Gráfico generado con Dia: http://dia-installer.de/

Vemos que tenemos una clase 'Controller' que recibe la petición del usuario, en ésta se implementa la lógica de negocio asociada a la petición. Para el acceso a la base de datos, se utiliza una clase DAO (Data Access Object), en esta clase implementamos las consultas a base de datos, siguiendo la especificación JPA. Esquemáticamente podemos representarlo de la siguiente manera en un diagrama UML de clases:

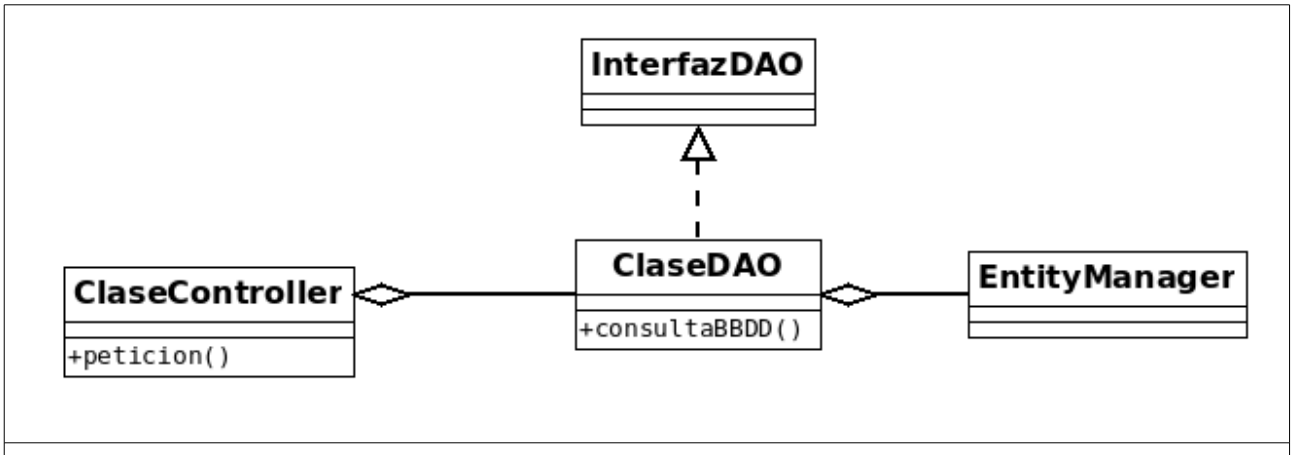


Fig. 4.7 Diagrama de clases

Gráfico generado con Dia: http://dia-installer.de/

Hay que indicar que en el diagrama anterior, la clase EntityManager es suministrada por el framework Spring, y el desarrollado no debe escribir ningún código en ella.

Base de datos

La capa de persistencia se realizará con un sistema gestor de base de datos MySQL v5.5. Se trata de un sistema open-source de gran aceptación en el desarrollo de aplicaciones web. El desarrollo con **MySQL** consistirá en el diseño del modelo entidad relación y la implementación de las tablas con sus relaciones. También introduciremos datos de prueba para facilitar el trabajo. No se incluirán en la base de datos ninguna consulta SQL de la aplicación, ya que de esto se encarga la aplicación escrita en Java mediante el uso de Hibernate y el plugin para conectarse a MySQL.

4.2 APIs y especificaciones de desarrollo

En el proyecto, se van a usar múltiples tecnologías, todas ellas open-source que se enumeran a continuación junto con las referencias más importantes para su consulta.

Capa de presentación

En la capa de presentación se utilizan las siguiente tecnologías:

HTML 5[25]
Es un lenguaje de marcado que se utiliza para presentar la información en la web. Los navegadores web, interpretan este lenguaje para presentar las páginas web por pantalla. También se va utilizar en la generación de los interfaces para dispositivos móviles.
Página oficial :
http://www.w3.org/TR/html5/

CSS[13]
Es un lenguaje de hojas de estilo que sirve para describir la presentación de un documento HTML.
Página oficial :
http://www.w3.org/Style/CSS/

JavaScript[33]
Es un lenguaje de programación interpretado por los navegadores web que sirve para dinamizar las páginas web. Junto con las HTML y CSS forman las tres tecnologías básicas de la web.
Referencia JavaScript Mozilla :
https://developer.mozilla.org/en-US/docs/Web/JavaScript

JQuery[36]
Es una librería JavaScript diseñada para simplificar el desarrollo en este lenguaje.:
https://jquery.com/

Jquery Mobile[37]
Es un framework para el desarrollo de aplicaciones para dispositivos móviles. Lo desarrolla el equipo de JQuery y utiliza sus librerías.
https://jquerymobile.com/

Stanford Javascript Crypto Library (SJCL)[66]
Es una librería para el uso de criptografía desde JavaScript
https://crypto.stanford.edu/sjcl/

Apache Cordova[3]
Apache Cordova, es el core de PhoneGap que a su vez es un framework para desarrollar aplicaciones híbridas para dispositivos móviles. El desarrollo se realiza usando HTML, CSS Y JavaScript. Posteriormente se compila la aplicación para cada plataforma específica.

https://cordova.apache.org/

Bootstrap[10]
Es un framework basado en HTML, CSS y JavaScript para el diseño de páginas web de forma que éstas se adapten automáticamente a los distintos tipos y tamaños de pantallas existentes hoy en día. En nuestro proyecto, será utilizado para el desarrollo de la aplicación web de administración.
http://getbootstrap.com/

Capa de lógica de negocio

Esta es la capa que se ejecuta en el servidor de aplicaciones. Incluye las siguiente tecnologías:

Java[27]
Java es un lenguaje de programación de propósito general, multiplataforma, orientado a objetos y concurrente. Ha tenido gran aceptación en el desarrollo de aplicaciones para internet y existe una extensión específica para el desarrollo de aplicaciones empresariales llamada Java EE[30]. También dispone de multitud de frameworks, librerías y entornos desarrollados por terceros. Para el desarrollo de aplicaciones java, existe el JDK (Java Development Kit) cuya versión open source OpenJDK[32], ha sido elegida como versión de referencia para el desarrollo de las versiones comerciales de Java desarrolladas por Oracle

OpenJDK[32]
http://openjdk.java.net/

API de referencia Java 7[31] :
http://docs.oracle.com/javase/7/docs/api/

Java Enterprise Edition (Java EE)[30] :
http://www.oracle.com/technetwork/java/javaee/overview/index.html

API de referencia java EE 7[31] :
https://docs.oracle.com/javaee/7/api/

Apache Tomcat[7]
Apache Tomcat es un servidor de aplicaciones web desarrollado por la Apache Software Foundation que cumple con varias de las especificaciones incluidas en Java EE. Dentro de este servidor se ejecutará nuestra aplicación escrita en java.

Web oficial[7]:
http://tomcat.apache.org/

Api de referencia Tomcat 7[9]:
https://tomcat.apache.org/tomcat-7.0-doc/api/

Spring Framework[63]
Es un framework open source para aplicaciones java. Dispone de varios módulos, algunos de los cuales han sido diseñados específicamente para el desarrollo de aplicaciones web, de forma que ofrece una arquitectura base sobre la que construir las aplicaciones.

Página oficial de Spring Framework[63]:

https://spring.io/

Referencia[64]:
http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/

Hibernate[22]
Es un framework ORM (object-relational mapping) escrito en java, que sirve para comunicar una base de datos de tipo entidad-relación, con un sistema orientado a objetos de forma que el desarrollador no debe generar consultas SQL directamente sobre la BBDD sino que consulta el modelo orientado a objetos. El desarrollador se centra en la programación. Permite integrarse con Spring y dispone de una versión que cumple con la especificación JPA (Java Persistence API) que forma parte de Java EE.

Documentación Hibernate ORM 4.3[24] :
http://hibernate.org/orm/documentation/4.3/

Api de Referencia Hibernate JPA [23]:
http://docs.jboss.org/hibernate/jpa/2.1/api/

Json[39]
Es un formato estándar y abierto para intercambiar datos. Los datos se formatean de forma legible para el ser humano. Es más ligero que XML por lo que ha ganado gran aceptación, especialmente en el desarrollo de aplicaciones para dispositivos móviles. Se utilizará para dar formato a los datos que se envían como respuesta de los servicios web y que serán interpretados por la aplicación móvil.
http://www.json.org/

Capa de persistencia

Esta capa es la encargada del almacenamiento físico de los datos. Se compone de un sistema gestor de base de datos MySQL

MySQL 5.5[45]
Es uno de los sistemas gestores de bases de datos más utilizados en el mundo. Además es open-source.
http://dev.mysql.com/doc/refman/5.5/en/

4.3 Seguridad

Para garantizar la seguridad de nuestra aplicación, todas las conexiones al servidor, tanto de los dispositivos móviles como de los navegadores web, se realizarán mediante el uso de protocolo https. Esto implica la utilización de certificados de seguridad en el servidor de producción. Durante el desarrollo, utilizaremos como servidor de integración el suministrado por Openshift, con el que ya disponemos de conexiones https. Además la forma de autenticar a los dispositivos móviles y la aplicación web es diferente.

Autenticación aplicación móvil

La aplicación para dispositivos móviles se comunicará con el servidor de aplicaciones mediante servicios web tipo REST. Estos servicios web, no tienen estado, por lo que dos llamadas al servicio no comparten ningún dato. Por tanto, no podemos esperar, que tras la operación de login

y el envío de credenciales, el sistema recuerde al usuario. La aplicación cliente es responsable de pasar el estado, bien enviando el usuario y contraseña cada vez, o mediante un token de seguridad o cualquier otro tipo de credencial que nos sirva para autenticar y autorizar al usuario.

En este proyecto, cuando el usuario hace login, en el servidor comprobaremos que el login es correcto comparándolo con la función hash del login almacenada en base de datos. Si es correcto, generaremos un token que enviaremos de vuelta al cliente y a su vez, lo guardaremos en base de datos para posteriores comprobaciones .

En la siguientes peticiones del cliente, el cliente montará una cadena de seguridad compuesta por:

cadena = url + hora del servidor + usuario +token;

De esa cadena calcularemos la función SHA256 y obtendremos el auth = SHA256(cadena)

En las peticiones que se envían al servidor, en la cabecera de la petición HTTP, se incluirán los siguientes datos:

- X-date: hora del servidor
- X-user: usuario,
- X-auth: auth

Cuando la petición llega al servidor haremos las siguientes comprobaciones:

- La hora enviada, no debe sobrepasar los 2 minutos de diferencia, ya que podría indicar que están intentando repetir la petición. Si supera los 2 minutos, devolvemos un error 403 (forbidden)
- El último acceso al servidor fue en menos de 15 minutos. Si supera los 15 minutos, actuamos como si hubiese caducado la sesión y volvemos a solicitar el login.
- Volvemos a crear la cadena del cliente y su función SHA256, esta vez, recuperamos el token de la base de datos, la cadena auth, enviada por el cliente debe ser igual a la calculada desde el servidor. En caso contrario, devolvemos un error 403 (forbidden).

Cuando el usuario realice logout, borraremos el token de la base de datos.

Autenticación aplicación web

La seguridad de la aplicación web de administración se implementará mediante el framework Spring. Este framework mantiene la información del usuario en el sesión, la cual utiliza cookies para guardar un identificador de sesión.

Spring permite la autenticación contra distintos sistemas: LDAP, Base de datos, memoria, incluso permite crear un sistema de autenticación propio. En este caso validaremos contra nuestra base de datos.

Ataques SQL Injection y Cross-site scripting[12][65]

Para evitar ataques de tipo inyección de SQL, utilizaremos de forma apropiada el lenguaje de acceso a base de datos de Hibernate HQL, de forma que evitaremos la concatenación de valores de entrada en las cadenas que componen las consultas HQL.

Otro ataque similar es el XSS o cross-site scripting, en este caso utilizaremos las etiquetas y atributos que ofrecen JSP y Spring para evitar la inyección de código JavaScript.

4.4 Guía de estilos

Aplicación móvil

Las interfaces de usuario para los dispositivos móviles, se desarrollaran usando los estilos de jquery-mobile, personalizado mediante el uso de la herramienta online:
https://themeroller.jquerymobile.com/

Con esa herramienta se han generado los siguientes imágenes de elementos para los interfaces móviles y las hojas de estilo CSS correspondientes:

Fig. 4.8 Ejemplos de interfaces con Jquery Mobile

Ref: https://themeroller.jquerymobile.com/

Aplicación web

Las interfaces de la aplicación web de administración, se desarrollarán utilizando las hojas de estilos CSS de **Bootstrap**: http://getbootstrap.com/

Los hojas de stilo de Boostrap, podrán ser personalizadas de acuerdo a la siguiente URL: http://getbootstrap.com/examples/theme/

A continuación mostramos algunos ejemplos de interfaces creados con bootstrap para la aplicación web:

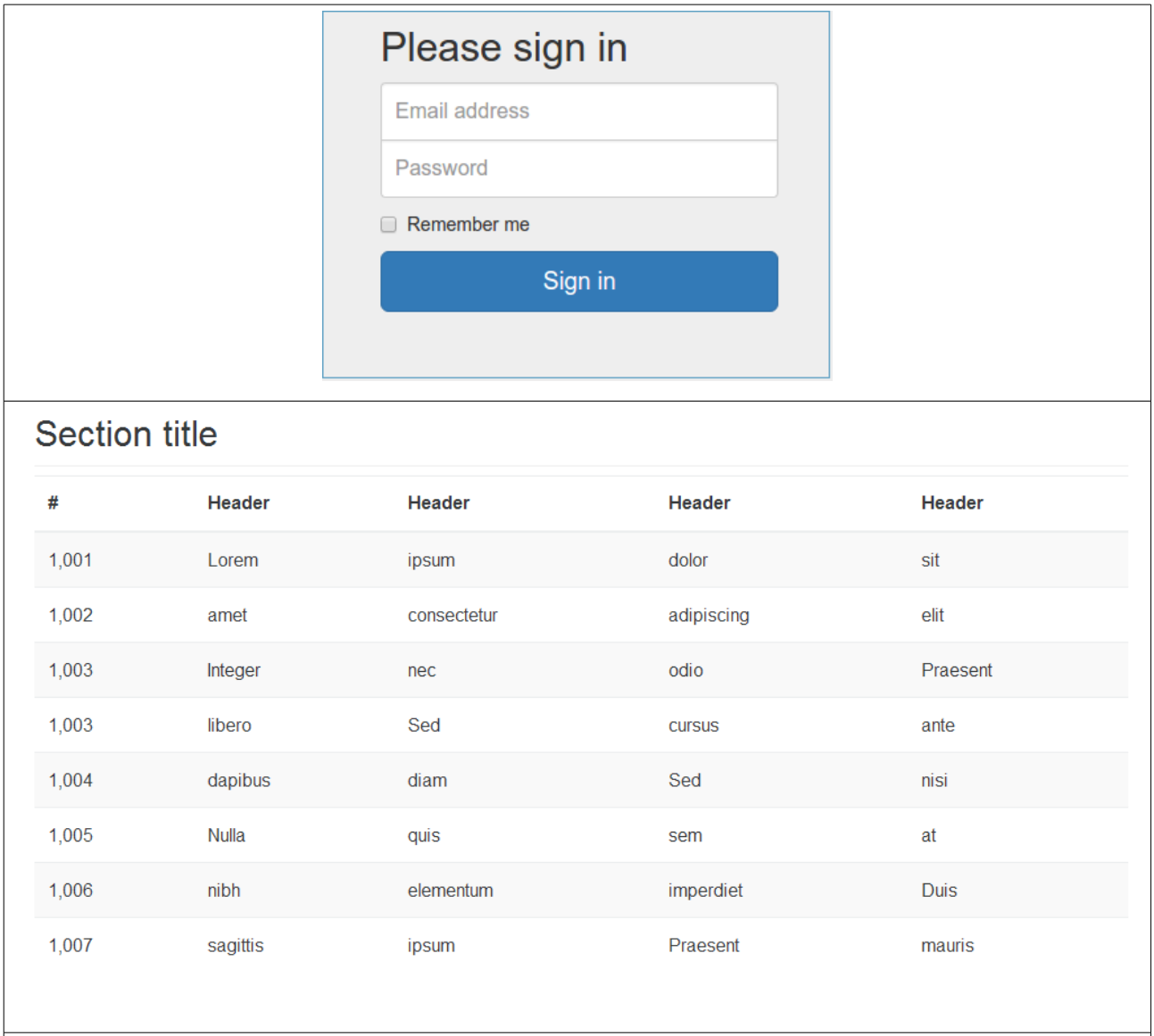


Fig. 4.9 Ejemplos de interfaces con Bootstrap

Ref: http://getbootstrap.com/examples/theme/

4.5 Especificaciones de pruebas

Pruebas unitarias

Las pruebas unitarias especificadas en el capítulo anterior, se realizarán mediante el uso de **Junit**[40] . Junit es un framework de pruebas para aplicaciones hechas en java y viene ya integrado en el entorno de desarrollo **Jboss Developer Studio**[34].

http://junit.org/

Pruebas de integración

Las pruebas de integración se realizarán de forma manual, siguiendo las especificaciones del capítulo de diseño. Se rellenará una tabla indicando el caso de uso a probar, la descripción, los pasos a seguir, el resultado esperado y el resultado obtenido. Deberá indicarse la persona que realizó las pruebas y la fecha de las mismas.

Aunque las pruebas se realizan de forma manual, también se podrán automatizar mediante el uso de herramientas como **Selenium IDE**[62] que junto con el navegador Firefox permite grabar y volver a reproducir una prueba repetidamente en un futuro.

http://www.seleniumhq.org/projects/ide/

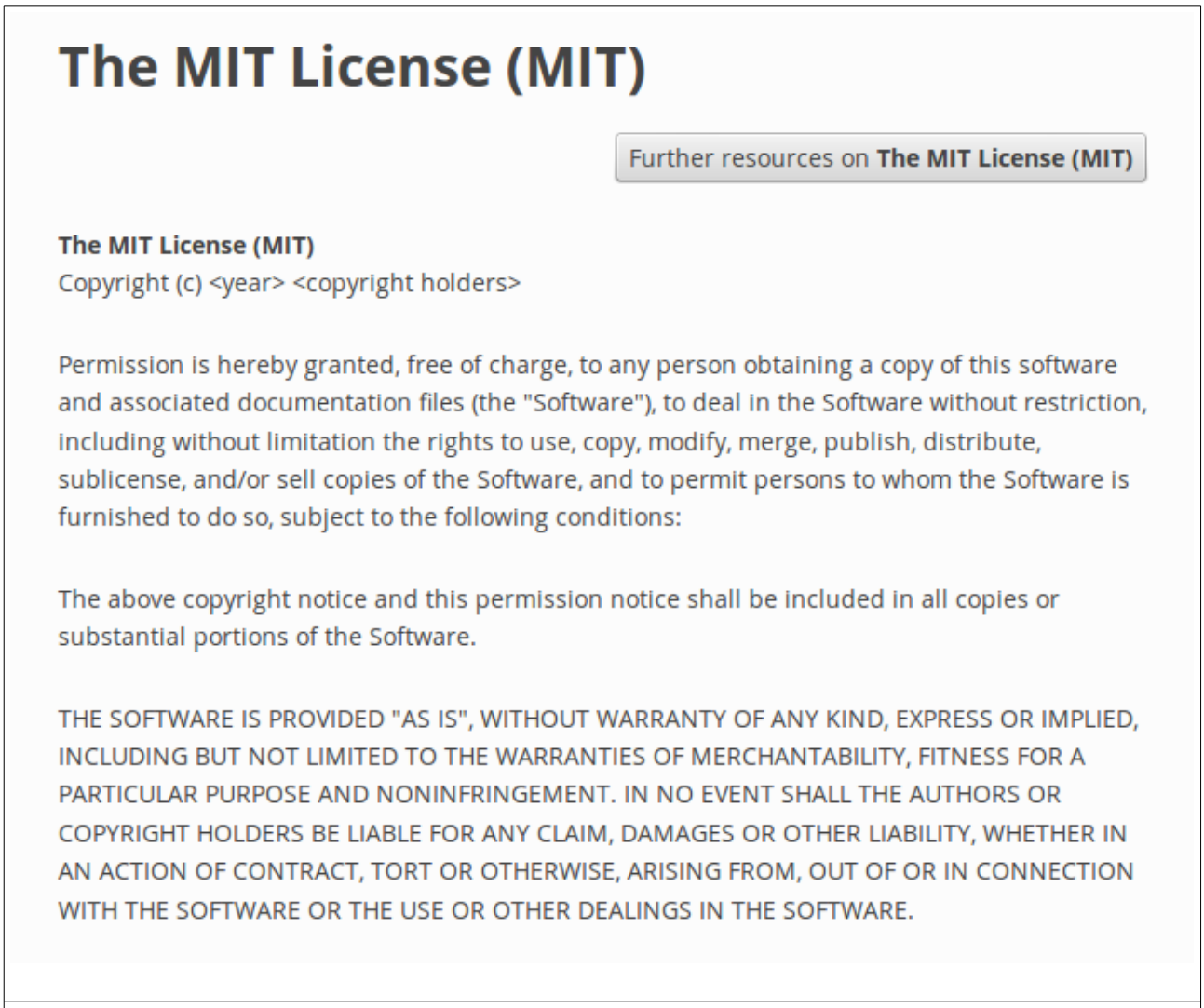
Pruebas de sistema

Las pruebas de sistema se realizarán mediante el uso de la herramienta **Apache Jmeter**[5] y cumpliendo con lo especificado en el capitulo de diseño:

http://jmeter.apache.org/

4.6 Licencia de desarrollo

Como licencia de este proyecto se ha elegido la **licencia MIT** o licencia X11[43]. Se trata de una licencia de código libre permisiva que se caracteriza por su simplicidad. A continuación, se incluye una imagen de la misma. El texto se puede encontrar en el documento de anexos.



The MIT License (MIT)

Further resources on **The MIT License (MIT)**

The MIT License (MIT)
Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Licencia MIT* **https://opensource.org/licenses/MIT**

* Texto disponible en el documento de anexos.

5 DESARROLLO

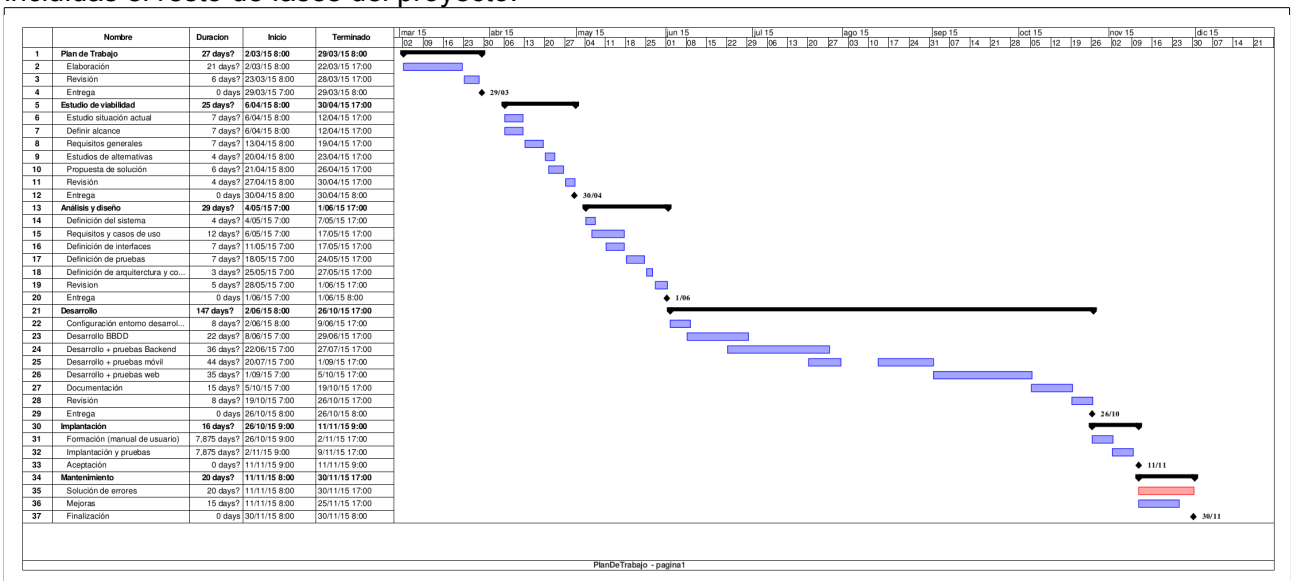
5.1 Planificación

Todo el proyecto se desarrolla utilizando una metodología clásica en cascada. Se prefiere esta metodología frente o otras de tipo ágil porque se trata de un proyecto de fin de máster donde solo hay un desarrollador y una única entrega final. Además es un proyecto de tamaño pequeño.

La fase de desarrollo se divide en las siguientes tareas:

- **Configuración del entorno de desarrollo.** Se instalará y configurará todo el software necesario para el desarrollo en un equipo de escritorio, así como la configuración de una cuenta Openshift[52] que hará la funciones de servidor de integración y sistema de control de versiones.
- **Desarrollo BBDD.** Incluye el diseño del modelo entidad-relación y su implementación en MySQL. También incluye la generación de datos iniciales de prueba.
- **Desarrollo y pruebas del backend.** Incluye el desarrollo de servicios web escritos en java para recuperar la información de la BBDD. Se harán en java usando el framework Spring.
- **Desarrollo y pruebas de la aplicación móvil.** Se desarrollarán los interfaces para dispositivos móviles y la programación para comunicarse con el backend. Todo ello usando Apache-Cordova.
- **Desarrollo y pruebas de la aplicación web de administración**. Se desarrollará los interfaces de la aplicación web de administración así como la programación en java del lado del servidor.

Estas tareas pueden verse en el siguiente diagrama de Gantt, donde además del desarrollo están incluidas el resto de fases del proyecto.



	Nombre	Duracion	Inicio	Terminado
1	**Plan de Trabajo**	**27 days?**	**2/03/15 8:00**	**29/03/15 8:00**
2	Elaboración	21 days?	2/03/15 8:00	22/03/15 17:00
3	Revisión	6 days?	23/03/15 8:00	28/03/15 17:00
4	Entrega	0 days	29/03/15 7:00	29/03/15 8:00
5	**Estudio de viabilidad**	**25 days?**	**6/04/15 8:00**	**30/04/15 17:00**
6	Estudio situación actual	7 days?	6/04/15 8:00	12/04/15 17:00
7	Definir alcance	7 days?	6/04/15 8:00	12/04/15 17:00
8	Requisitos generales	7 days?	13/04/15 8:00	19/04/15 17:00
9	Estudios de alternativas	4 days?	20/04/15 8:00	23/04/15 17:00
10	Propuesta de solución	6 days?	21/04/15 8:00	26/04/15 17:00
11	Revisión	4 days?	27/04/15 8:00	30/04/15 17:00
12	Entrega	0 days	30/04/15 8:00	30/04/15 8:00
13	**Análisis y diseño**	**29 days?**	**4/05/15 7:00**	**1/06/15 17:00**
14	Definición del sistema	4 days?	4/05/15 7:00	7/05/15 17:00
15	Requisitos y casos de uso	12 days?	6/05/15 7:00	17/05/15 17:00
16	Definición de interfaces	7 days?	11/05/15 7:00	17/05/15 17:00
17	Definición de pruebas	7 days?	18/05/15 7:00	24/05/15 17:00
18	Definición de arquitectura y co...	3 days?	25/05/15 7:00	27/05/15 17:00
19	Revision	5 days?	28/05/15 7:00	1/06/15 17:00
20	Entrega	0 days	1/06/15 7:00	1/06/15 8:00
21	**Desarrollo**	**147 days?**	**2/06/15 8:00**	**26/10/15 17:00**
22	Configuración entorno desarrol...	8 days?	2/06/15 8:00	9/06/15 17:00
23	Desarrollo BBDD	22 days?	8/06/15 7:00	29/06/15 17:00
24	Desarrollo + pruebas Backend	36 days?	22/06/15 7:00	27/07/15 17:00
25	Desarrollo + pruebas móvil	44 days?	20/07/15 7:00	1/09/15 17:00
26	Desarrollo + pruebas web	35 days?	1/09/15 7:00	5/10/15 17:00
27	Documentación	15 days?	5/10/15 7:00	19/10/15 17:00
28	Revisión	8 days?	19/10/15 7:00	26/10/15 17:00
29	Entrega	0 days	26/10/15 8:00	26/10/15 8:00
30	**Implantación**	**16 days?**	**26/10/15 9:00**	**11/11/15 9:00**
31	Formación (manual de usuario)	7,875 days?	26/10/15 9:00	2/11/15 17:00
32	Implantación y pruebas	7,875 days?	2/11/15 9:00	9/11/15 17:00
33	Aceptación	0 days?	11/11/15 9:00	11/11/15 9:00
34	**Mantenimiento**	**20 days?**	**11/11/15 8:00**	**30/11/15 17:00**
35	Solución de errores	20 days?	11/11/15 8:00	30/11/15 17:00
36	Mejoras	15 days?	11/11/15 8:00	25/11/15 17:00
37	Finalización	0 days	30/11/15 8:00	30/11/15 8:00

Fig. 5.1 Diagrama de Gantt, planificación del proyecto

Gráfico generado con ProjectLibre: https://www.projectlibre.org/

5.2 Configuración entorno de desarrollo

Para el desarrollo del proyecto, es necesario la instalación de distintos componentes de software. Se utilizará una distribución **Linux** como sistema operativo del equipo de escritorio con el que se va a trabajar. Según se indicó en el capítulo dedicado al diseño, para el desarrollo de la aplicación móvil necesitaremos **Apache Cordova**, además, para poder realizar pruebas reales sobre un dispositivo móvil, instalaremos el lenguaje de programación java y el **Android SDK**.

Para el Backend, usaremos una base de datos **MySql** y un servidor de aplicaciones **Tomcat**. Como entorno de desarrollo hemos seleccionado el **Jboss Developer Studio**[34] el cual que está basado en Eclipse[16] y se integra con la plataforma de desarrollo **OpenShift**[52].

Instalación del sistema operativo

Para el desarrollo del proyecto, se ha seleccionado el sistema operativo **Linux Mint**[41]. Se trata de una distribución Linux diseñada especialmente para equipos de escritorio. Está basada en la distribución Ubuntu, que a su vez está basada en Debian. La instalación resulta sencilla y durante la misma se incluyen drivers y plugins necesarios para el funcionamiento de la mayor parte de componentes de los equipos de escritorio. Dispone a su vez de varios tipos de escritorio. En nuestro caso hemos seleccionado el escritorio Mate, ya que se trata de un escritorio clásico, basado en GNOME 2, muy estable y compatible con la mayor parte de aplicaciones.

En la página oficial se puede descargar tanto la ISO de la distribución como instrucciones de instalación y utilización:
http://www.linuxmint.com/

Instalación de Apache Cordova

Para la instalación de Apache Cordova, en primer lugar debemos instalar Node.js. Para ello podemos ejecutar la siguiente instrucción:

```
sudo apt-get install nodejs
```

Además creamos el alias “node” para que Cordova lo encuentre:

```
sudo ln -s /usr/bin/nodejs /usr/bin/node
```

También hay que instalar npm (Node Package Manager) mediante:

```
sudo apt-get install npm
```

Para instalar Cordova usamos npm:

```
npm install -g cordova
```

Podemos comprobar la versión instalada mediante:

```
cordova --version
```

Instalación de Java SDK y Android SDK

Como nuestra distribución de linux está basada en Ubuntu, es necesario descargarse la librería libgl1-mesa-dev mediante:

```
sudo apt-get install libgl1-mesa-dev
```

Instalamos el SDK de java mediante:

```
sudo apt-get install openjdk-7-jdk
```

Descargamos el Android Studio completo para Linux desde:

http://developer.android.com/sdk/index.html#top

Lo descomprimimos en nuestra carpeta home. Desde consola, entramos en la carpeta que acabamos de crear y ejecutamos el script de instalación

```
cd ~/android-studio/bin
./studio.sh
```

Se abre un instalador gráfico para la instalación y descarga de componentes necesarios, entre ellos el android SDK tools .

También es necesario añadir la herramienta ant para poder compilar o emular en Android . Hay que descargar la última versión desde: http://ant.apache.org/bindownload.cgi y descomprimirla en la carpeta del usuario.

Finalmente, hay que añadir la variable ANDROID_HOME, para ello se crea el fichero .bashrc en nuestra carpeta home con las siguientes líneas:

```
ANDROID_HOME=/home/user/Android/Sdk
export ANDROID_HOME
PATH=$PATH:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools
export PATH
ANT_HOME=~/apache-ant-1.9.4
export ANT_HOME
PATH=$PATH:~/apache-ant-1.9.4/bin
export PATH
```

Instalación de MySQL

La instalación del sistema gestor de base de datos MySQL, puede realizarse desde el Gestor de Software de Linux Mint instalando el paquete mysql-server, o bien desde consola ejecutando el comando

```
sudo apt-get install mysql-server
```

Durante el proceso de instalación, nos pedirá que introduzcamos una contraseña para el usuario root (administrador) de MySQL .

El entorno de desarrollo para MySQL Workbench, puede ser instalado igualmente desde el Gestor de Software o desde consola. En este caso se trata del paquete mysql-workbench

```
sudo apt-get install mysql-workbench
```

El entorno MySQL Workbench aparecerá disponible en el menú de Linux Mint, y podremos conectarnos al servidor MySQL introduciendo el usario 'root' y la contraseña indicada en la instalación de MySQL.

Una vez instalado MySQL y con la ayuda MySQL Workbench, crearemos la base de datos 'examlist' haciendo click en el botón 'Create new schema in the connected server' e introduciendo

el nombre 'examlist'. Sobre este esquema podemos lanzar los siguientes comando SQL para para crear un usuario específico para este esquema:

```
CREATE USER 'adminKxrmvmD'@'localhost' IDENTIFIED BY 'XXXXXXX';
GRANT ALL PRIVILEGES ON examlist.* TO 'adminsB2IW8h'@'localhost';
FLUSH PRIVILEGES;
```

Este usuario será utilizado posteriormente en nuestra aplicación.

Configuración de OpenShift

Para usar OpenShift como plataforma online de desarrollo, primero debemos abrir una cuenta gratuita en la dirección: https://www.openshift.com/ Accedemos a la opción 'Sign up' y rellenamos el formulario que nos aparece. Tras el alta de la cuenta, podemos configurar las herramientas que utilizaremos, para ello accedemos a nuestra cuenta y hacemos click en 'Add Application...'. Nos aparecera un listado con todos los tipos de aplicaciones disponibles. En el grupo de 'java' seleccionamos Tomcat 7 (Fig. 5.2)

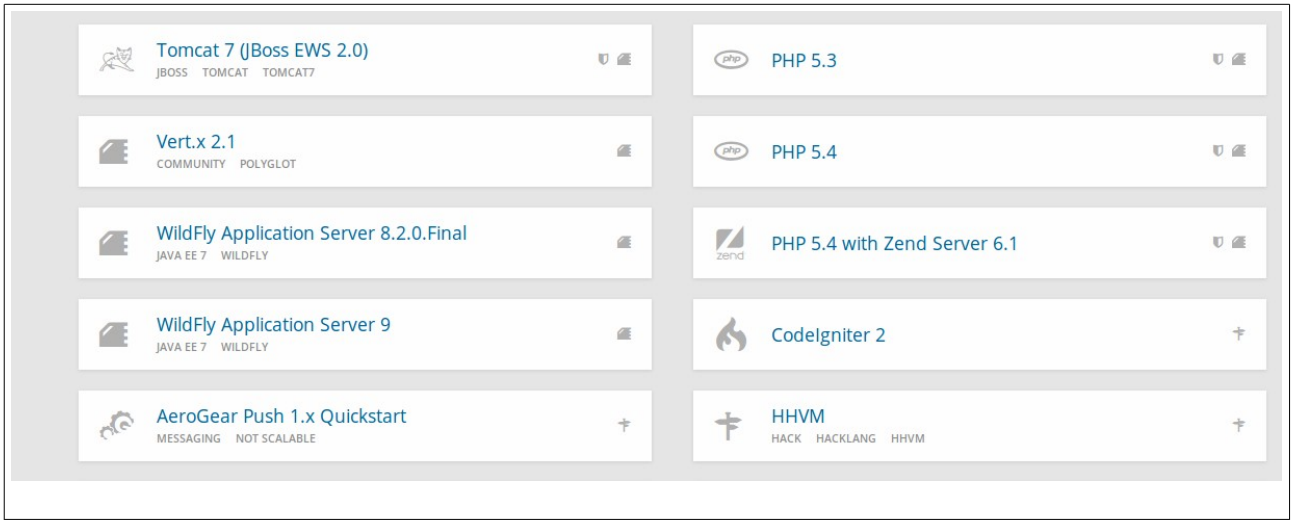


Fig. 5.2 Herramientas en OpenShift

A continuación, deberemos seleccionar el la url pública de nuestra aplicación. En nuestro caso hemos nombrado a la aplicación 'examlist' de forma Openshift nos asigna la URL:

http://http://examlist-robertocalvouoc.rhcloud.com

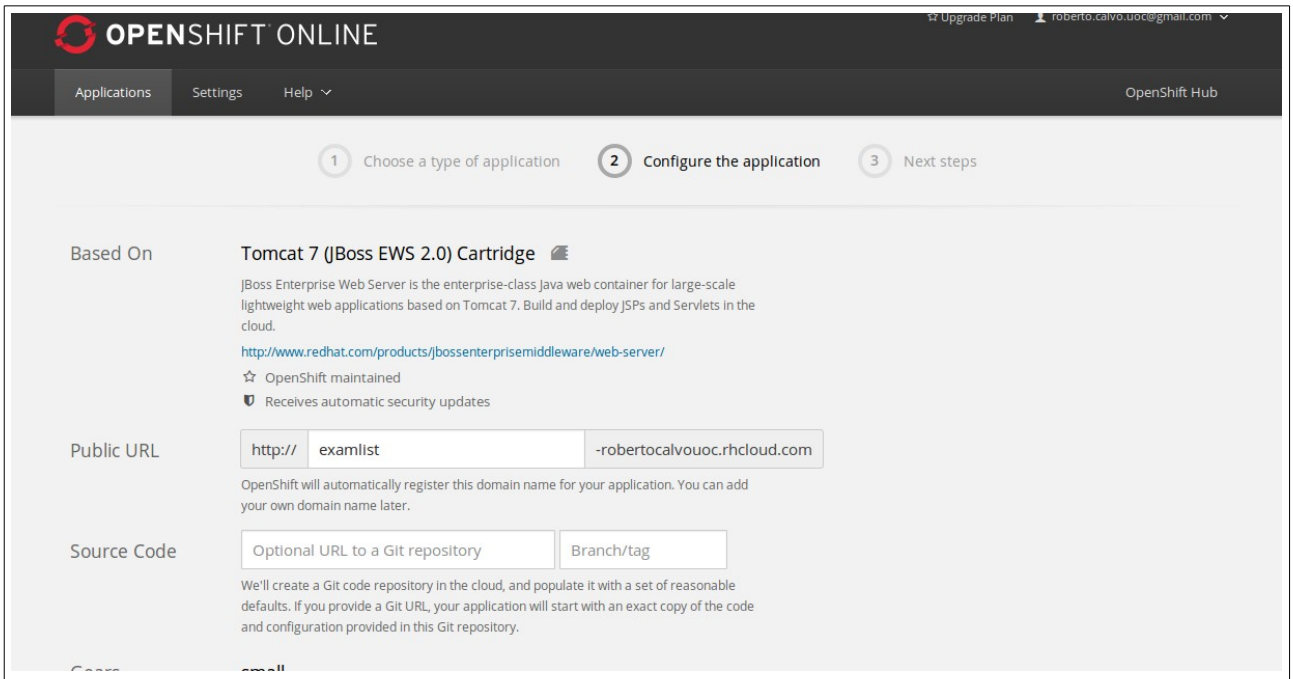


Fig. 5.3 Selección de URL de la aplicación

A continuación, OpenShift nos ofrece la opción de instalar una base de datos para nuestra aplicación. Seleccionamos MySQL 5.5. Tras la instalación MySQL también se nos ofrece la instalación de phpMyAdmin, la cual aceptamos. Finalmente se nos mostrará una pantalla de confirmación (Fig. 5.3) donde se nos indica un resumen de la instalación realizada, incluyendo usuario y password para gestionar nuestra base de datos en OpenShift así como la url de conexión vía SSH.

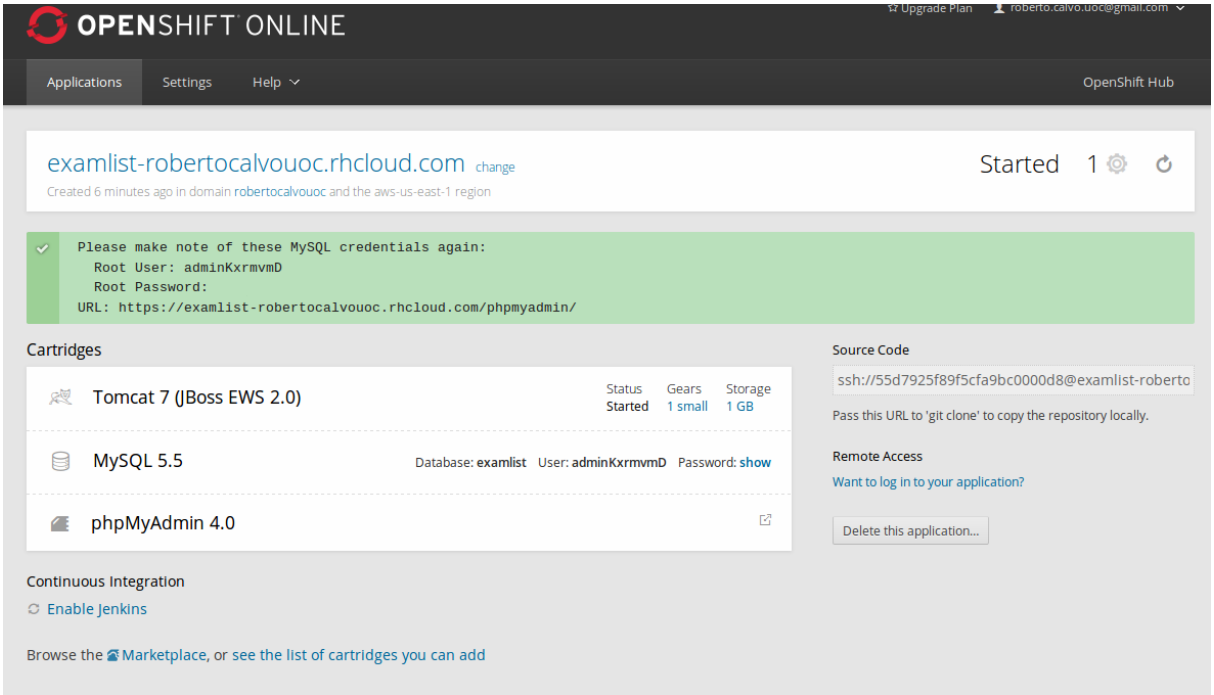


Fig. 5.4 Pantalla de confirmación

Instalación y configuración de Jboss Developer Studio

Para descargarnos el Jboss Developer Studio, tenemos que acceder a la URL:

http://www.jboss.org/products/devstudio/overview/

Al pulsar sobre el botón 'Download' nos pide que nos demos de alta, y nos descarga el fichero: jboss-devstudio-8.1.0.GA-installer-eap.jar. Para ejecutar el fichero, con Linux Mint podemos hacer doble click sobre el mismo, o bien desde consola, acceder al directorio que lo contiene y ejecutar:

```
java -jar  jboss-devstudio-8.1.0.GA-installer-eap.jar
```

Se nos muestra un instalador gráfico que nos guiará durante el resto del proceso.

Una vez instalado, procedemos a configurarlo para que se integre con nuestra cuenta OpenShift creada anteriormente. Para ello accedemos a `File/Import/` y seleccionamos `OpenShift/Existing OpenShift Application`

La primera vez, configuraremos una clave pública ssh-rsa que subiremos a nuestra cuenta OpenShift y nos permitirá la comunicación directa entre nuestro entorno de desarrollo y OpenShift. Primeramente introduciremos nuestro usuario/contraseña de la cuenta OpenShift (Fig. 5.5) Posteriormente, en la pantalla de la figura 5.5 podremos crear una clave SSH pública y subirla al servidor de OpenShift, para ello en la pantalla de la imagen seleccionaremos 'New' , tras crear las claves, podremos seleccionar la aplicación creada en OpenShift en el punto anterior y descargarla. De esta manera la aplicación estará incluida en nuestro IDE y podremos subir los cambios y desplegar en el servidor Tomcat en OpenShift mediante el sistema de control de versiones GIT[17].

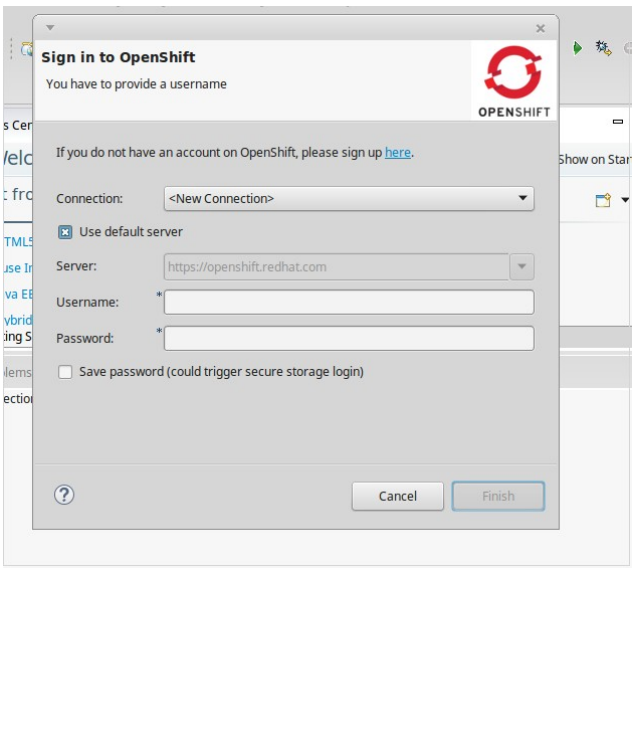


Fig. 5.5 Configuración entorno

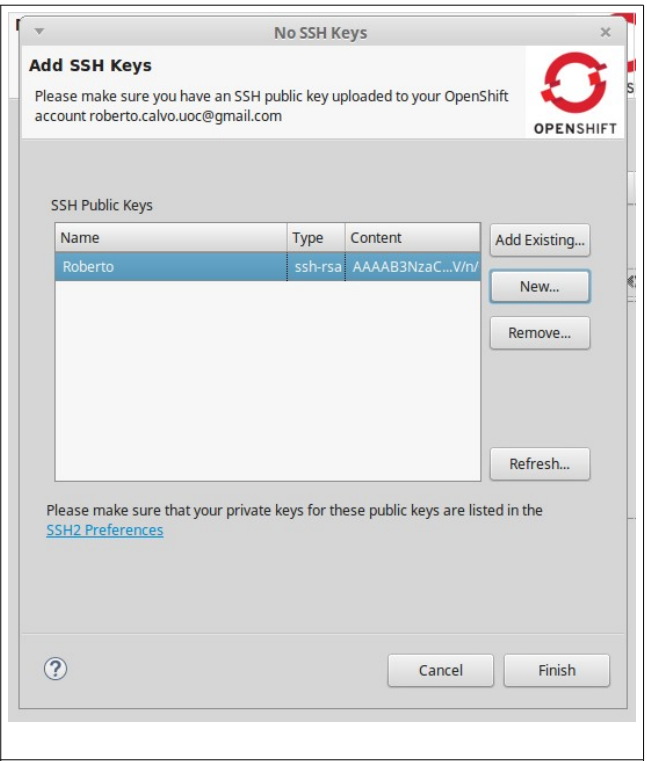


Fig. 5.6 Configuración entorno

Además, vamos a configurar un servidor Tomcat en nuestra máquina local. De esta forma trabajaremos con más fluidez e incorporaremos el desarrollo a OpenShift solo cuando tengamos alguna funcionalidad terminada o código que queramos incorporar al repositorio. En primer lugar descargamos Apache Tomcat desde la URL: http://tomcat.apache.org/. Descargamos la versión 7

para asegurarnos la compatibilidad con OpenShift. Para la instalación, basta con descomprimir el fichero en alguna carpeta. Dentro del Jboss Developer Studio, en la pestaña 'servers' haciendo click con el botón secundario, selecionamos `new/server` y a continuación 'Apache Tomcat v7.0'. En la opción `'Server runtime environment'` añadimos el servidor que acabamos de descargar y nos aparecerá disponible bajo la pestaña '`Servers`'

Una vez instalado Tomcat en nuestra máquina local, lo configuramos para que utilice también nuestra base de datos local MySQL. En la pestaña 'Project Explorer' en 'Servers' abrimos el fichero `context.xml` e introducimos la información relativa a la conexión a la base de datos local. En nuestro caso sería:

```
<Resource name="jdbc/MySQLDS" auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="adminKxrmvmD" password="XXXXXXX" driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/examlist"/>
```

5.3 Técnicas de desarrollo

A continuación, se va a describir el proceso de desarrollo de la aplicación. Junto con esta memoria se entrega todo el código fuente de la aplicación.

Desarrollo de la base de datos

La base de datos, se desarrolla utilizando la herramienta MySQL Workbench. Para crear las bases de datos, se puede escribir directamente el código que genera las tablas y todas sus propiedades. Por ejemplo, para la tabla usuario podemos lanzar la una sentencia sobre el esquema examlist de MySQL de este tipo:

```
CREATE TABLE `usuario` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `login` varchar(50) NOT NULL,
  `password` varchar(150) NOT NULL,
(...)
  `fecha_modif_estado` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `login_UNIQUE` (`login`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
```

En lugar de escribir el código para cada tabla, también se pueden utilizar las herramientas visuales de MySQL Workbench. Para ello, sobre el esquema examlist, con el botón derecho seleccionamos '`Create Table...`' e introducimos los datos del script anterior de forma visual:

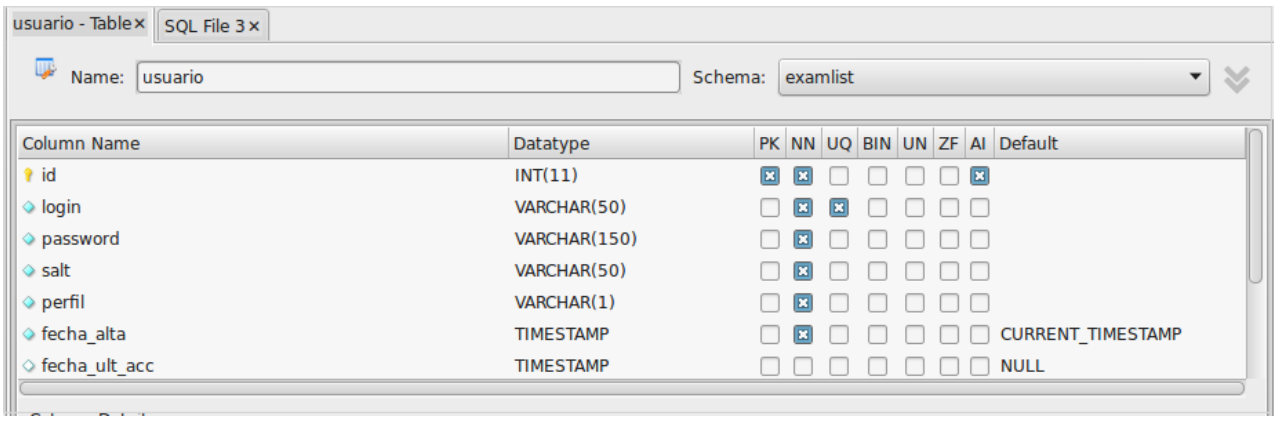


Fig. 5.7 Creación de tablas en MySQL Workbench

Tras la creación de todas las tablas, MySQL Workbench nos permite obtener el siguiente diagrama E-R que representa todo el modelo de datos creado:

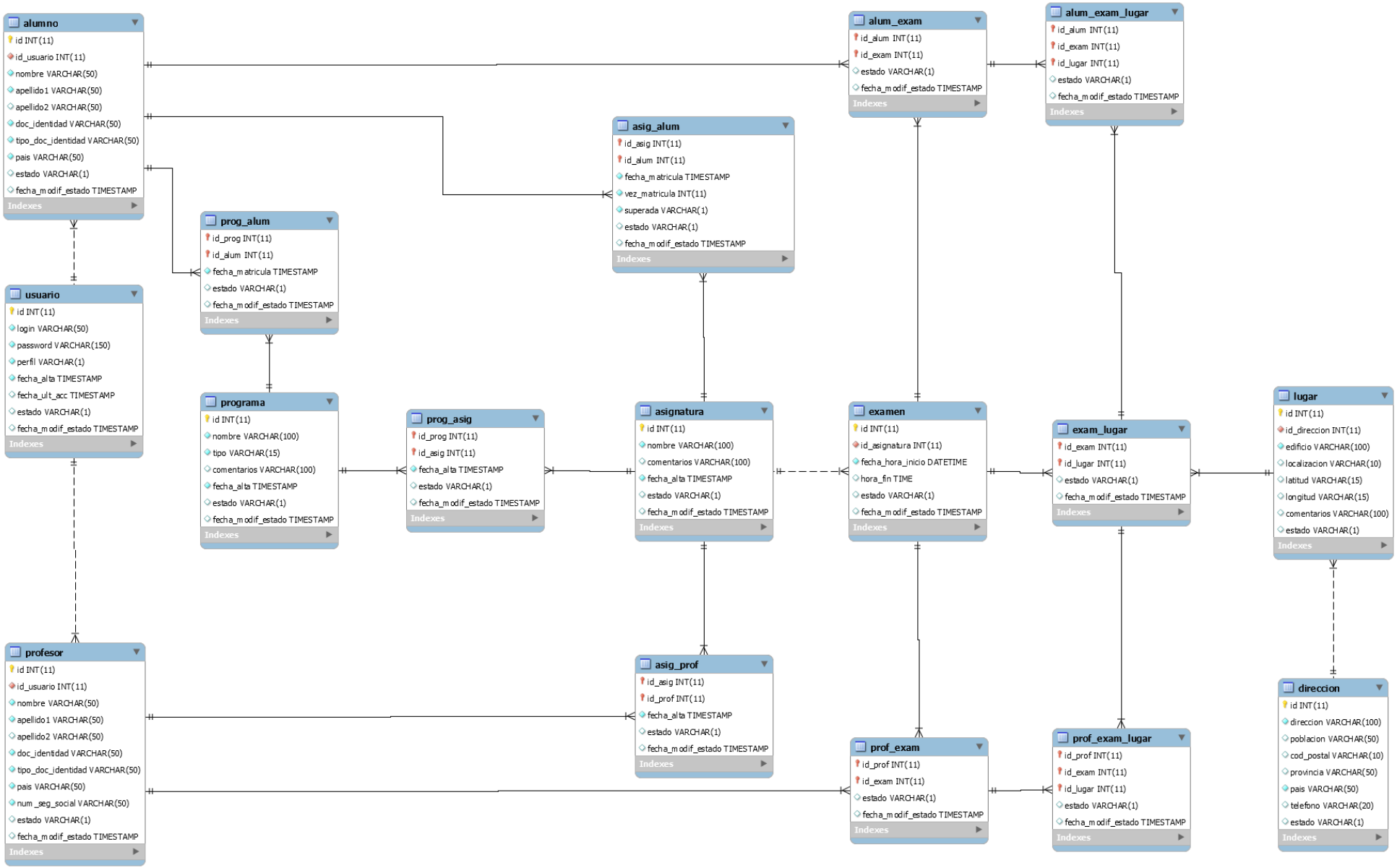


Fig. 5.8 Diagrama Entidad-Relación. Generado con MySQL WorkBench

Desarrollo de la aplicación del servidor

La aplicación web se desarrolla usando el framework Java Spring. Para incluir tanto las librerías de spring como cualquier otra necesaria, se utiliza la herramienta 'maven'. Para ello, modificamos el fichero `pom.xml`, donde se añaden todas la dependencias necesarias. El fichero completo de dependencias se encuentra en: `examlist/pom.xml`.

Con el botón derecho, sobre el fichero `pom.xml`, seleccionamos `Run As/Maven install` y maven se conectará a sus repositorios y nos descargará todas las librería allí definidas.

En nuestra aplicación, configuramos la conexión a la BBDD mediante el fichero :

`examlist/src/main/resources/META-INF/persistence.xml`

En ese fichero se puede ver la configuración de Hibernate como proveedor de JPA que se conecta a una base de datos MySQL.

Ahora debemos configurar el framework Spring. En primer lugar, en el fichero:

`examlist/src/main/webapp/WEB-INF/web.xml`,

se agregan los datos para configurar el framework Spring.

El framework Spring, a su vez dispone de sus ficheros de configuración. Para el módulo MVC, existe el fichero:

`examlist/src/main/webapp/WEB-INF/examlist-spring-mvc-context.xml`

En ese fichero, entre otras cosas, aparece la ruta de las páginas JSP, en nuestro caso:

`examlist/src/main/webapp/WEB-INF/views/`

Finalmente, el contexto de Spring, lo configuramos mediante el fichero:

`META-INF/Spring/applicationContext.xml`:

En ese fichero, hemos indicado que la conexión a BBDD la realizamos mediante el fichero anterior `pesistence.xml`, además, indicamos las clases que la aplicación usará para gestionar aspectos de la conexión a BBDD.

Una vez configurado Spring, se puede comenzar con el desarrollo. A continuación se indican las carpetas generadas con el código fuente:

Clases controladoras que aceptan las peticiones:

`examlist/src/main/java/org/uoc/examlist/controller`

Clases DAO encargadas del acceso a base de datos:

`examlist/src/main/java/org/uoc/examlist/data`

Clases que mapean las tablas de nuestro modelo de datos :

`examlist/src/main/java/org/uoc/examlist/model`

El código de estas clases se pueden generar de forma manual, no obstante, nuestro entorno de desarrollo dispone de herramientas que permiten la generación automática de estas clases una vez configurada la conexión a la base de datos.

Clases para mapear la salida en formato json de las peticiones desde el móvil:

`examlist/src/main/java/org/uoc/examlist/responsePojo`

Clases encargadas de la seguridad relativa a autenticación y autorización:

`examlist/src/main/java/org/uoc/examlist/security`

Clases para mapear la salida en formato json de las peticiones desde el móvil:

`examlist/src/main/java/org/uoc/examlist/responsePojo`

Clases de utilidades:

`examlist/src/main/java/org/uoc/examlist/utils`

Clases que mapean los formularios de las páginas web:

`examlist/src/main/java/org/uoc/examlist/view`

Desarrollo de la aplicación móvil

Para desarrollar la aplicación móvil, después de instalarnos Cordova, debemos abrir un terminal sobre la carpeta en la que queramos crear la aplicación. Lanzamos el siguiente comando:

```
cordova create nombreAplicacion
```

En nuestro caso, hemos escogido como nombre de la aplicación “examlist”. Con ese comando, se creará un directorio “nombreAplicacion” y dentro de él, tendremos la estructura básica de nuestra aplicación cordova. Si accedemos al directorio, veremos una carpeta con nombre `www`, dentro de ella están los ficheros sobre los que trabajaremos. Existe un fichero `index.html`, donde pondremos el código html que da forma a los interfaces móviles. También hay otra capeta: js, dentro de la cual hay un fichero `index.js`, en ese fichero escribiremos el código JavaScript de nuestra aplicación.

También podemos añadir ya las plataformas para las que queremos portar la aplicación. En nuestro caso añadiremos Android. Pero además, vamos a añadir la plataforma 'browser' que nos va a ser de utilidad para probar nuestra aplicación directamente con un navegador. Para añadir las plataformas hacemos:

```
cordova platform add android
cordova platform add browser
```

Con esto, ya podremos probar a compilar la aplicación para las plataformas añadidas. Eso se hace usando el comando:

```
cordova build
```

Con este comando se habrá generado el código tanto para la plataforma Android, como para la plataforma browser. Durante la ejecución del mismo, nos indicará la ruta y el fichero .apk para instalarlo en un dispositivo Android. Todo el código generado tras la compilación se encuentra en: `examlist/src/main/webapp/examlist_mobile/platforms`

Cada vez que se modifique el código, y queramos probarlo deberemos repetir este último comando.

Tras estos pasos, podemos empezar a generar el código de nuestra aplicación. El código se encuentra en las rutas que se enumeran a continuación.

Interfaces de usuario, escritos en html:

`examlist/src/main/webapp/examlist_mobile/www/index.html`

Hojas de estilo CSS:

`examlist/src/main/webapp/examlist_mobile/www/css`

Programación en JavaScript:

`/home/usuario/git/examlist/src/main/webapp/examlist_mobile/www/js`

Durante el desarrollo, podemos ir probando la aplicación de distintas maneras. Se puede usar un navegador, ya que hemos añadido la plataforma browser, se puede compilarla para Android mediante el comando `cordova build`, y copiar el fichero .apk a un dispositivo móvil, o también, usar un emulador de Android. Para utilizar el emulador de Android, es necesario el Android SDK que se instaló previamente, si ejecutamos en un terminal de Linux el comando:

```
Android
```

Aparece el Android SDK Manager. En la opción del menú `Tools / Manage AVDs...` podemos configurar un dispositivo virtual Android.

Para ejecutar la aplicación en el dispositivo virtual, con un terminal Linux, nos posicionamos en la carpeta de nuestra aplicación y ejecutamos el comando:

```
cordova run android
```

Se arranca el dispositivo virtual, con la aplicación instalada en el mismo, según la figura 5.9

Fig. 5.9 Aplicación ejecutándose en dispositivo virtual Android

5.4 Ejecución de pruebas

Pruebas unitarias

La realización de la pruebas unitarias, se realiza mediante el uso de **junit**[40] integrado con el framework Spring. Por cada método de acceso a base de datos creamos otro de prueba que ejecutamos directamente con el entorno de desarrollo.

Las clases java encargadas de realizar las pruebas unitarias pueden encontrarse en:

`examlist/src/test/java`

Par ejecutar cada clase de pruebas, en el entorno de pruebas, hay que hacer click con el botón derecho y luego hacer click en `Run As/Junit Test`

A continuación se muestran pantallas de ejemplo con los resultados obtenidos en la máquina de desarrollo:

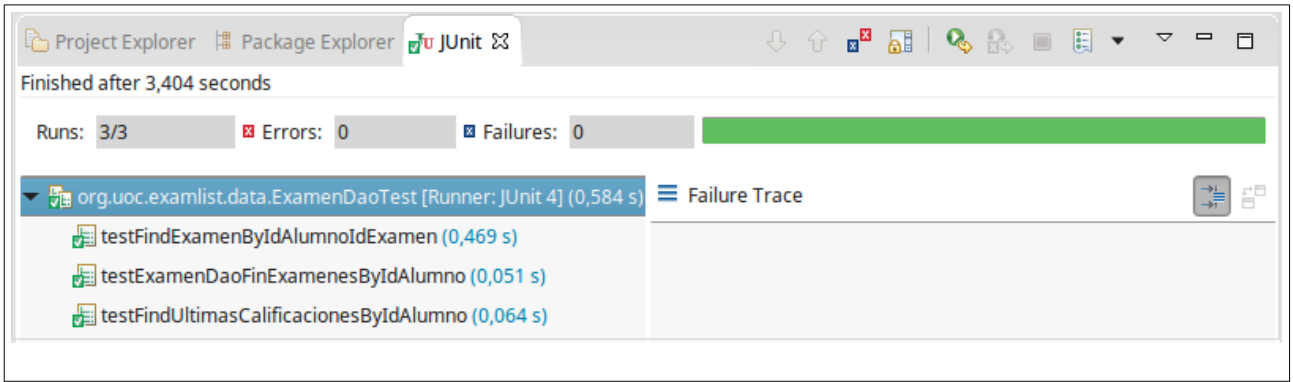


Fig. 5.10 Resultados de pruebas unitarias a ExamenDao

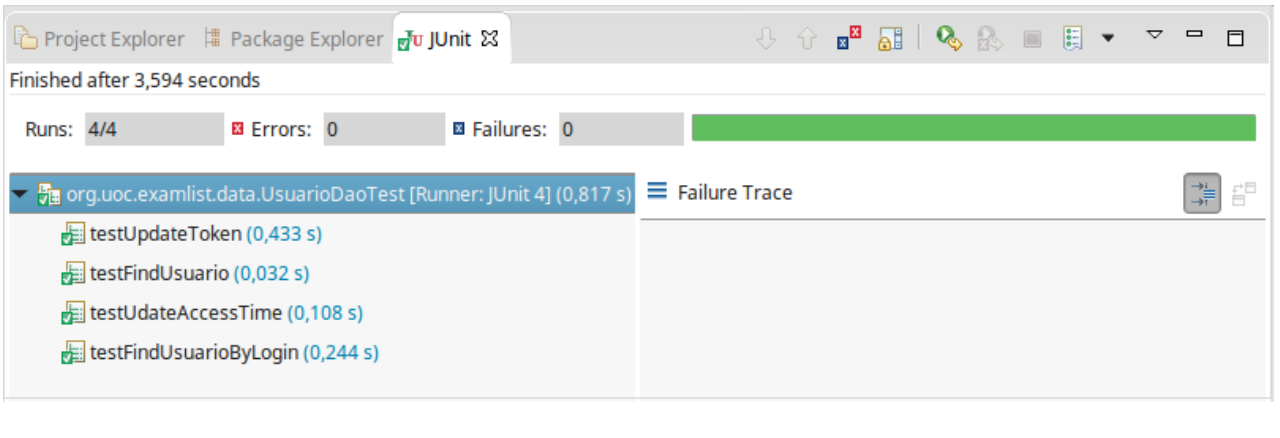


Fig. 5.11 Resultados de pruebas unitarias a UsuarioDao

Pruebas de integración

Las pruebas de integración se realizan siguiendo el punto 3.6 Primeramente se prepara un guión de pruebas y tras la realización de las mismas cumplimentamos el resultado obtenido. Las pruebas se pueden realizar de forma manual o automatizadas con Selenium IDE[61] A continuación se indica un ejemplo de este guión.

Test diseñado por: Roberto Calvo Mendoza

Área de actuación: Dispositivo móvil Android

Objetivos: Verificar la aplicación para dispositivos móviles

Requisitos: Dispositivo móvil Android, con la aplicación 'Examlist' instalada y con acceso a internet.

Prueba	Descripción	Pasos	Resultado esperado	Resultado obtenido	Realizado por/fecha
PU101-1	Validar login de alumno en aplicación móvil	1-Acceder a la aplicación móvil 2-Introducir usuario/password del alumno 3-Pulsar botón acceder	Se valida el login y se muestra la pantalla de selección de información	OK	RCM / 31/08/2015
PU101-2	Validar login de alumno en aplicación móvil	1-Acceder a la aplicación móvil 2-Introducir usuario/password del profesor 3-Pulsar botón acceder	Se valida el login y se muestra la pantalla con el listado de sus exámenes.	OK	RCM / 31/08/2015
(...)					

Los resultados completos de las pruebas se encuentran en el documento de Anexos.

Pruebas de sistema

La pruebas de carga sobre el sistema se realizan con Jmeter. En primer hay que descargarse **Jmeter**[5] de la página oficial: http://jmeter.apache.org/

Basta con descomprimir la carpeta, dar permisos de ejecución al script: `<Jmeter home>/bin/jmeter` y ejecutarlo. Una vez arrancado, asignamos un nombre al plan de pruebas y añadimos un 'Grupo de Hilos' haciendo click con el botón derecho sobre el plan de pruebas y seleccionando: `Añadir/Hilos(usuarios)/Grupo de hilos`.

En la pantalla de configuración del grupo de hilos podemos introducir el número de usuarios concurrentes y el tiempo máximo para ejecutar esos hilos.

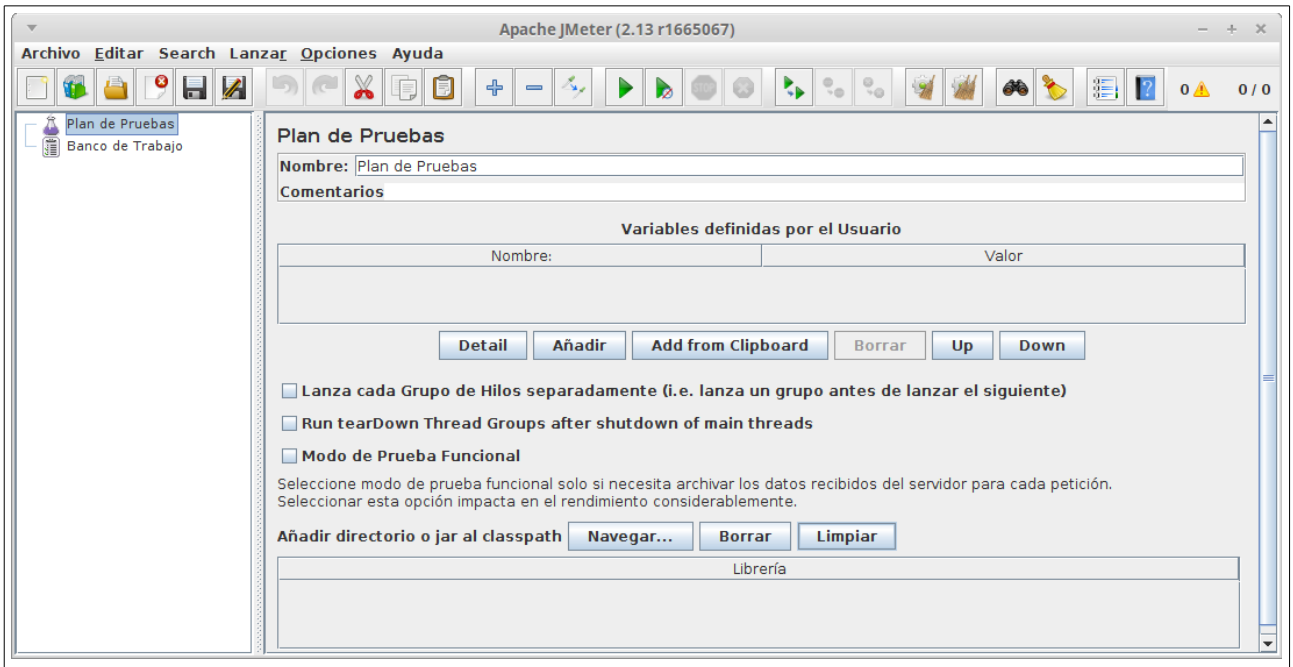


Fig. 5.12 Jmeter, pantalla inicial

Al grupo de hilos, tenemos que añadirle las peticiones http que deseamos realizar, para ello, con el botón derecho sobre el grupo de hilos `Añadir/Muestreador/Petición HTTP` . El proceso se realiza 5 veces para añadir las siguientes acciones:

- login
- listado de próximos exámenes
- detalle de un examen
- últimas calificaciones
- logout

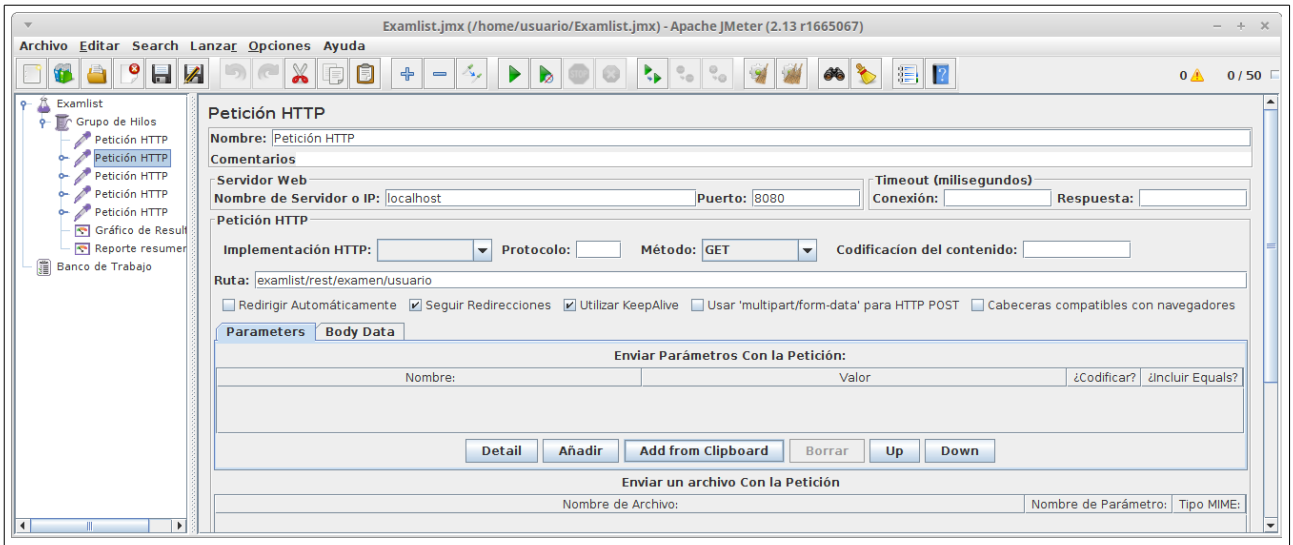


Fig. 5.13 Jmeter, añadir peticiones http

Hay que señalar, que en nuestro caso, necesitamos añadir parámetros a la cabecera de la petición http. Sobre la petición HTTP hacemos : `Añadir/Elementos de configuración/Gestor de cabecera HTTP` . Y añadimos los parámetros de cabecera necesarios

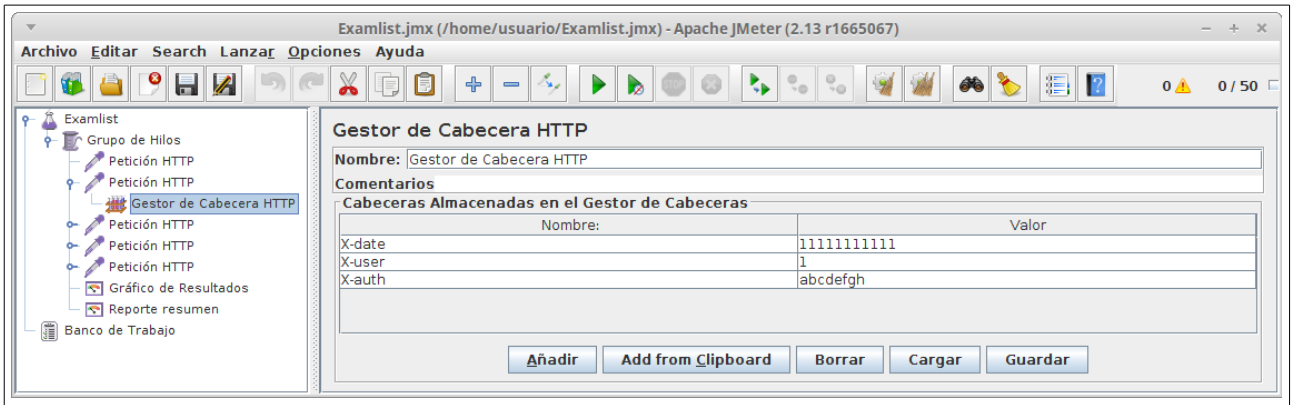


Fig. 5.14 Jmeter, añadir cabeceras http

Finalmente vamos a añadir, al grupo de hilos, los receptores para poder visualizar los resultados. Seleccionamos: `Añadir/Receptor/Reporte Resumen` y también: `Añadir/Receptor/Gráfico de resultados` para ver los resultados de forma gráfica .

Ya solo queda hacer click en el botón de Play y al finalizar el test podremos visualizar el reporte resumen y el gráfico de resultados que hemos añadido

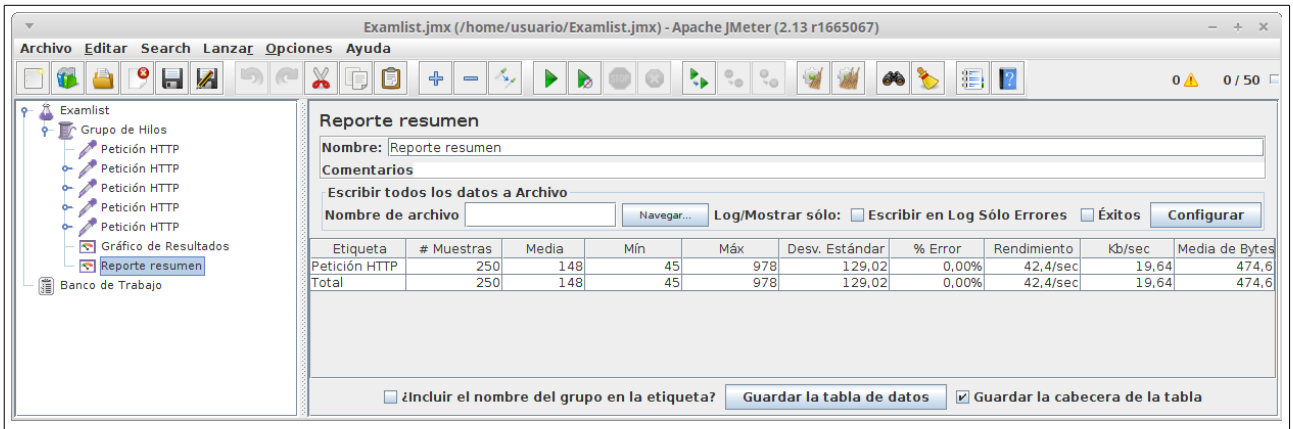


Fig. 5.15 Jmeter, tabla de resultados

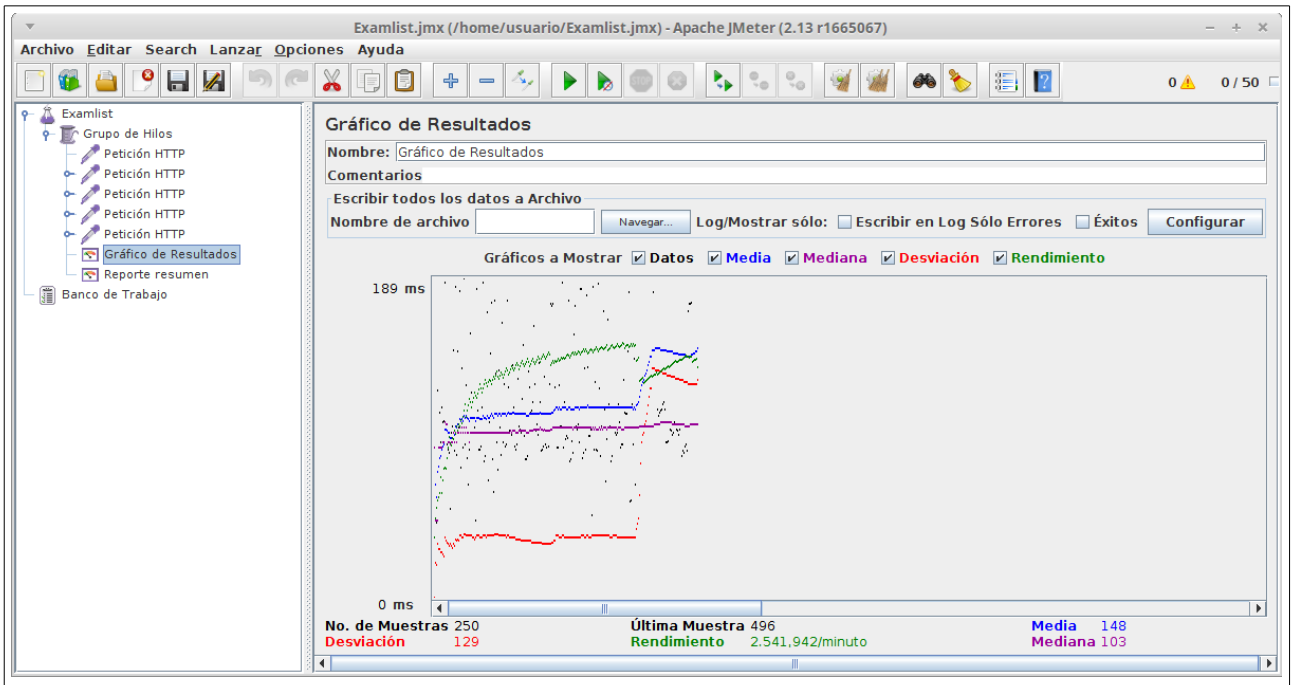


Fig. 5.16 Jmeter, gráfico de resultados

Se puede observar en los resultados que el tiempo medio por petición es de 148 milisegundos y el máximo obtenido ha sido 978 milisegundos.

A continuación podemos realizar la misma prueba elevando el número de hilos para ver la carga que podría soportar nuestro sistema.

Si subimos a 300 hilos, obtenemos tiempos medios de 740 milisegundos, que aunque están dentro de los requisitos, observamos que ya hay tiempos máximos de 2.701 milisegundos.

Fig. 5.17 Jmeter, tabla de resultados, 300 usuarios

Hay que señalar que estas pruebas se han realizado desde la misma máquina de desarrollo, en mi caso un portátil con procesador i3 y 4GB de Ram. Es previsible que en un entorno real de producción, con máquinas dedicadas, el rendimiento sea mejor. No obstante, estas mismas pruebas deberían volver a realizarse contra un servidor de producción.

5.5 Documentación del código

Es importante mantener el código documentado, de esta forma se facilita la corrección de posibles errores o la modificación para futuras versiones. Una de las formas más básicas de documentar es la incorporación de comentarios explicativos en todo aquello que el desarrollador considere oportuno aclarar. Esta técnica es válida para cualquier lenguaje de programación, no obstante, hoy en día existen herramientas específicas de documentación de código que a partir de comentarios introducidos en un formato específico, permiten generar la documentación en formato HTML. En este proyecto en concreto, para documentar el código java utilizamos **Javadoc**[29] y para documentar el código javascript utilizaremos **JSDoc3**[38].

Código javascript

La herramienta **JSDoc 3** se puede instalar mediante el uso de npm, mediante el comando:

```
npm install jsdoc
```

Mediante la inclusión de comentarios al inicio de cada componente Javascript (clase, función, evento..) introduciremos el texto de la documentación. A continuación se muestra un ejemplo de este tipo de comentarios:

```
(...)
/** Evento lanzado al pulsar el botón de acceder de la
 * pantalla de login.
 * Lanza una petición Ajax, al web-services que valida
 * el usuario y contraseña.
 * Si la petición es correcta muestra la pantalla de menu.
 * Si se produce un error, muestra un mensaje indicativo
 * @event Click en botón 'Acceder'
 */
$("#btn-acceder").click( function( event, ui) {
        event.preventDefault();
        clickBotonAcceder();
} );
(...)
```

En el código fuente del fichero:

`examlist/src/main/webapp/examlist_mobile/www/js/index.js`

puede leerse todos los comentarios para la documentación.

Para generar la documentación, desde un terminal, lanzamos el siguiente comando.

```
<ruta de node_modules>/.bin/jsdoc  -d <ruta de destino> <ruta index.js>/index.js
```

Se genera la documentación en formato HTML, La siguiente pantalla muestra un ejemplo:

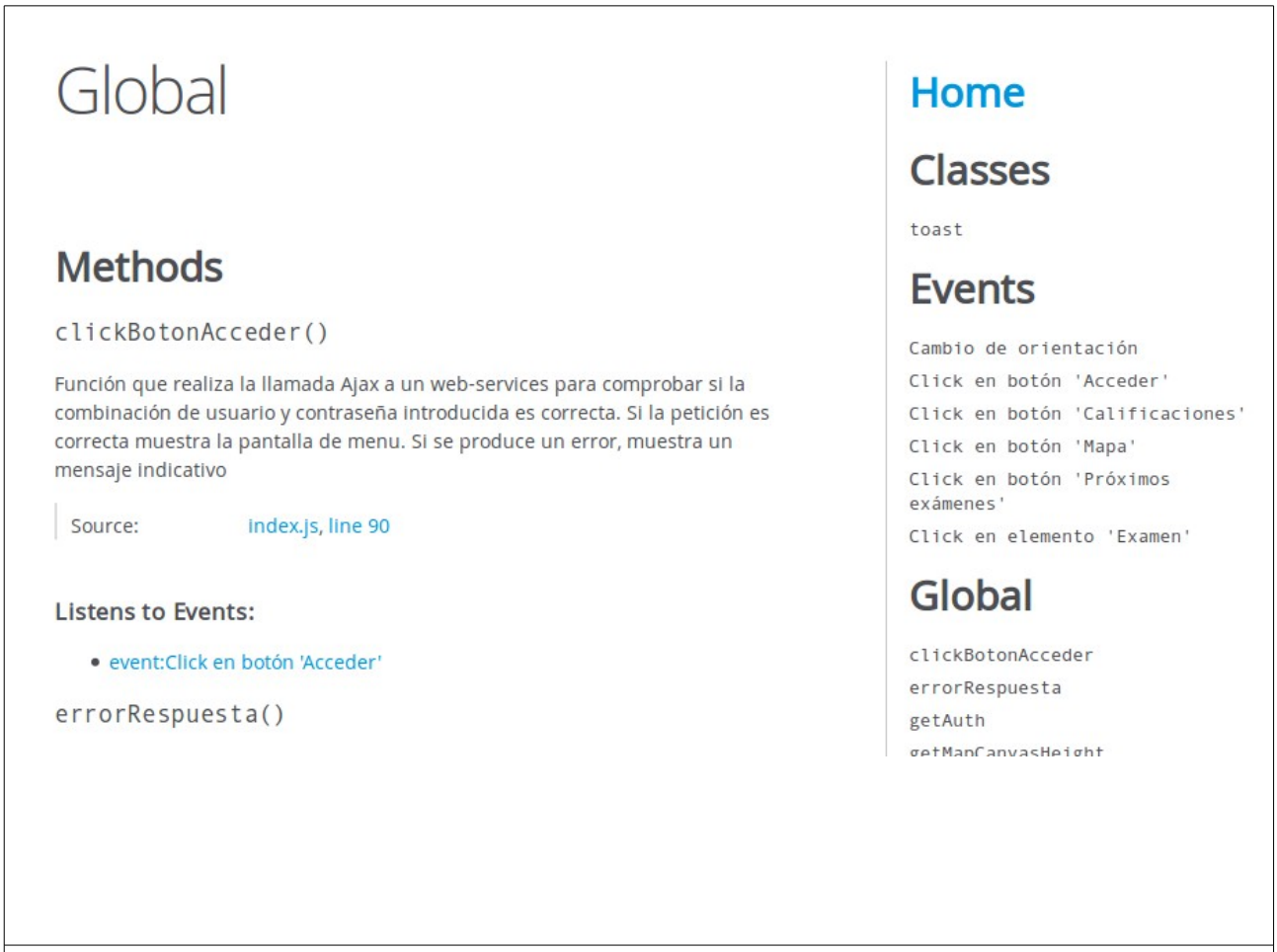


Fig. 5.18 Documentación JSDoc de index.js

La documentación completa puede encontrarse en la carpeta:

`examlist/doc/JSDoc`

Código java

La herramienta **javadoc**[29] para la documentación de código java, se encuentra incluida en el JDK y viene integrada en cualquier entorno de desarrollo java.

El desarrollador debe incluir comentarios con unas anotaciones especificas al inicio de cada clase o método que la herramienta javadoc es capaz de interpretar.

Un ejemplo de los comentarios a introducir sería:

```
(...)
/**
* Método encargado de mostrar la página con el
* listado de los alumnos y sus calificaciones.
*
* @param idExamen  código del exámen
* @param model     objeto modelo del framework Spring
* @return          nombre de la página jsp de respuesta
* @since           1.0
*/
@RequestMapping(value = "/{idExamen}", method = RequestMethod.GET)
public String buscadorExamen(@PathVariable("idExamen")int idExamen, Model model) {
(...)
```

Luego basta con ejecutar javadoc en nuestro entorno en el menú `project/Generate javadoc`, y se obtiene la documentación incluida en la carpeta:

`examlist/doc/JavaDoc`

A continuación se muestran pantallas de ejemplo con el resultado:

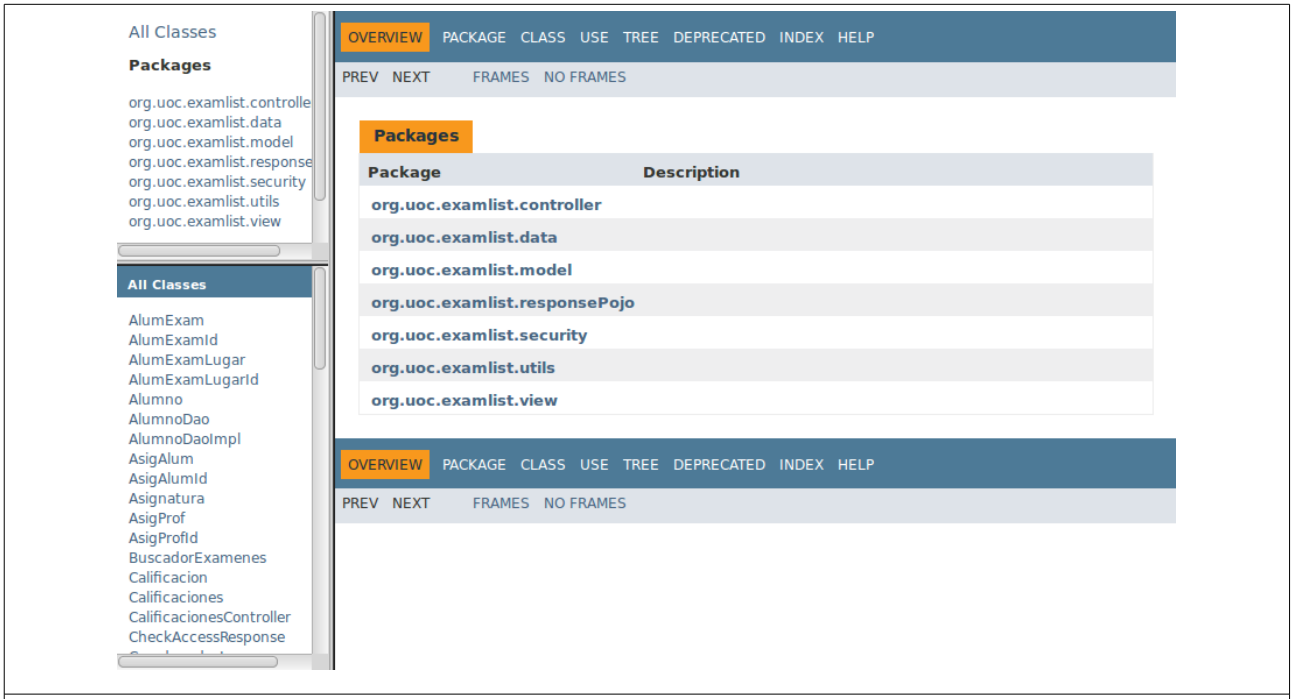


Fig. 5.19 Página de inicio de la documentación javadoc del proyecto

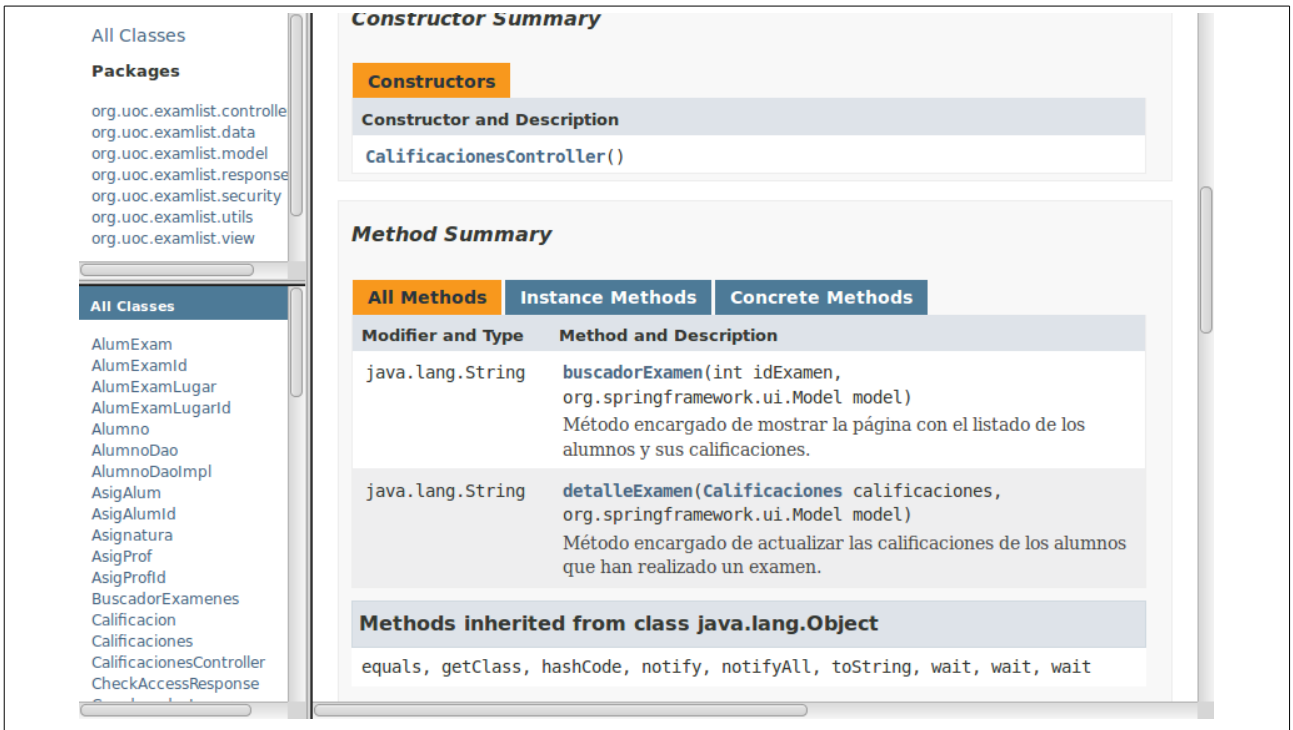


Fig. 5.20 Ejemplo documentación javadoc

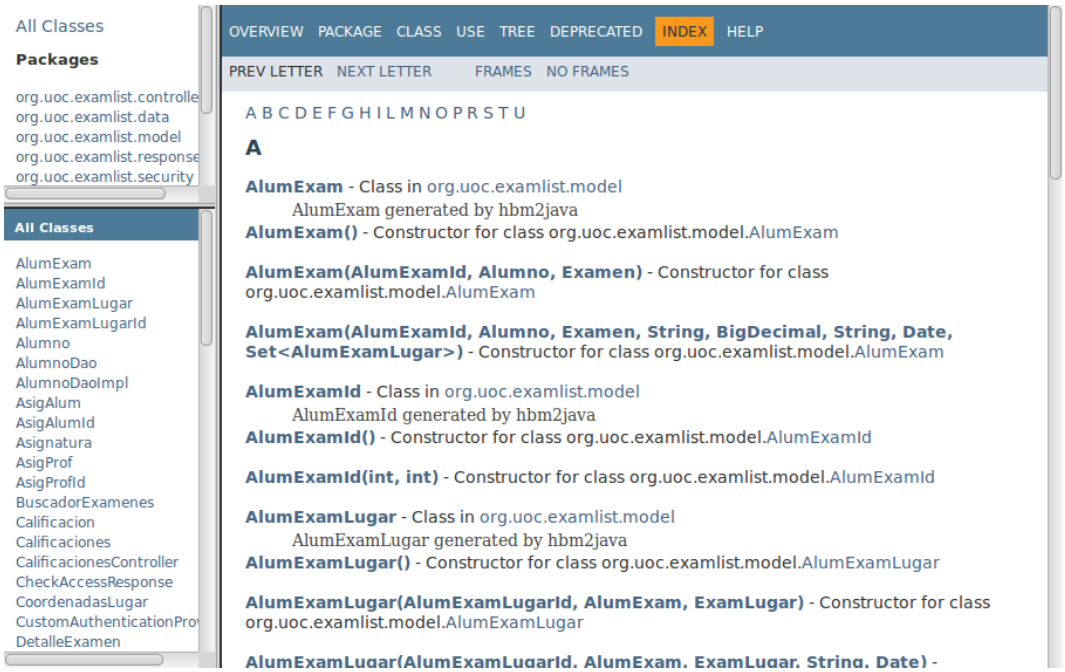


Fig. 5.21 Índice de documentación javadoc

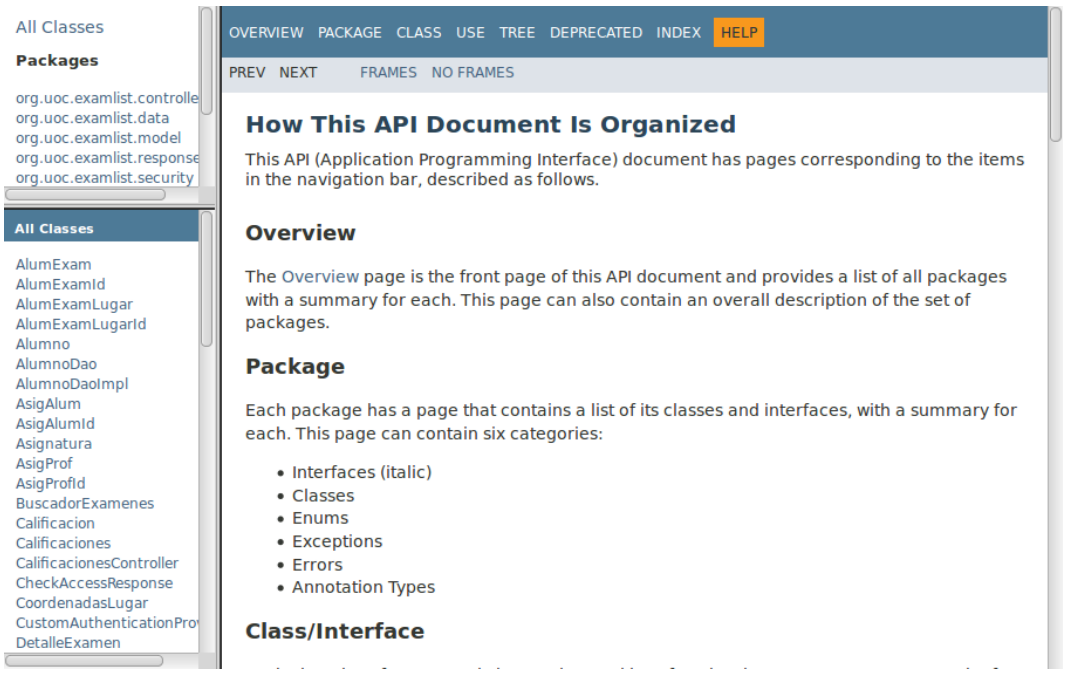


Fig. 5.22 Ayuda de la documentación javadoc

6 IMPLANTACIÓN

En el alcance del proyecto se incluye el soporte durante la fase de implantación en las instalaciones del cliente. El cliente establecerá el calendario de planificación que considere más oportuno. A continuación se incluye una guía de implantación con los pasos a seguir en esta fase. También se incluye unas recomendaciones sobre seguridad y copias de respaldo o Backups para asegurarnos un nivel de servicios óptimo.

6.1 Guía de implantación

En este punto se indican los pasos a seguir para instalar la aplicación en un entorno de producción. La configuración final del hardware será decisión del cliente. No obstante, en el siguiente punto se dan algunas recomendaciones de configuración de la infraestructura con el objeto de asegurarnos un nivel de servicio óptimo.

Sistema operativo del servidor

La decisión del servidor sobre el que se instalará la aplicación corre a cargo del cliente. Se podrá utilizar los servidores existentes, instalar nuevos equipos o la creación de máquinas virtuales, bien en los sistemas ya existentes del cliente o mediante la contratación externa de este servicio. Como sistema operativo del servidor, podemos recomendar la utilización de alguna distribución de Linux orientada a servidores, por ejemplo: **Ubuntu Server**[69], **Debian**[14], **Red Hat**[60] o **CentOS**[11].

Instalación de base de datos

La instalación de la BBDD se realizará de forma análoga a la indicada en el punto 5.2.4. Se realizarán las siguientes operaciones:

- Instalación del sistema gestor de base de datos MySQL.
- Creación de la base de datos 'examlist'.
- Ejecución del script para la creación de tablas y la introducción de los primeros datos de prueba.
- Finalmente deberá introducirse la información real que contendrá la aplicación mediante scripts de inserción de datos.

Instalación de la aplicación java

Para la instalación de la aplicación java en los servidores de producción, se deberán seguir los siguientes pasos:

- Instalación del JRE de java en el servidor. Los detalles exactos para su instalación dependerán de la distribución de Linux utilizada. En todo caso, se podrá encontrar disponible en el gestor de paquetes de la distribución escogida o también se podrá instalar manualmente.
- Instalación del servidor de aplicaciones Tomcat. Al igual que ocurre con el JRE de java, el servidor Tomcat se encuentra disponible en muchos de los gestores de paquetes de las distribuciones de Linux. También se puede descargar e instalarlo manualmente desde[7]: https://tomcat.apache.org/tomcat-7.0-doc/appdev/installation.html
- Tras instalar el servidor Tomcat, se deberá establecer la conexión a la BBDD instalada en el punto anterior. Habrá que actualizar el fichero `context.xml` de igual manera que se indicó en el punto 5.2.6
- Desplegar la aplicación en el servidor. Para ello, primero se creará un fichero .WAR de la aplicación, a continuación se copiará ese fichero en la carpeta `$CATALINA_BASE/webapps` de nuestro servidor tomcat y se arrancará el servidor.

Instalación de la aplicación móvil

Para instalar la aplicación móvil, primero deberá compilar para las plataformas de destino seleccionadas por el cliente según se indicó en el punto 5.3.3. Antes de compilar la aplicación, habrá que actualizar la URL para llamar a los web-services del servidor de aplicaciones a la URL final de producción. El fichero con la ruta se encuentra en:
`examlist/src/main/webapp/examlist_mobile/www/js/index.js`
Inicialmente se instalará de forma manual. Cuando el cliente considere oportuna su distribución, se incluirá en las distintas plataformas de distribución.

Pruebas de implantación

El la fase de implantación se repetirán las pruebas que se realizaron durante el desarrollo:

- Se ejecutarán las pruebas unitarias sobre para los accesos a base de datos con el objetivo de ver los tiempos de respuesta de la conexión entre el servidor de aplicaciones y la base de datos.
- Se ejecutarán las pruebas de sistema para ver la capacidad de respuesta del sistema de producción.
- Se repetirán las pruebas de integración, de esta forma se comprobará que el sistema incluye todas las funcionalidades y cumple con los requisitos iniciales.

Formación de usuarios

Al tratarse de una aplicación sencilla en su utilización, no se se prevé un formación específica a los usuarios finales. Sí que se incluye en la entrega final un manual de usuario que puede encontrarse en el documento de anexos.

6.2 Nivel de servicios

Para asegurarnos un nivel de servicios con disponibilidad 24x7, a continuación se enumeran unas recomendaciones de infraestructura y seguridad.

Infraestructura

Con objeto de conseguir una alta disponibilidad del sistema, se recomienda disponer de una infraestructura con los sistema duplicados, para que ante un eventual fallo de algún componente, se pueda seguir ofreciendo el servicio. Una posible configuración se indica en el siguiente diagrama:

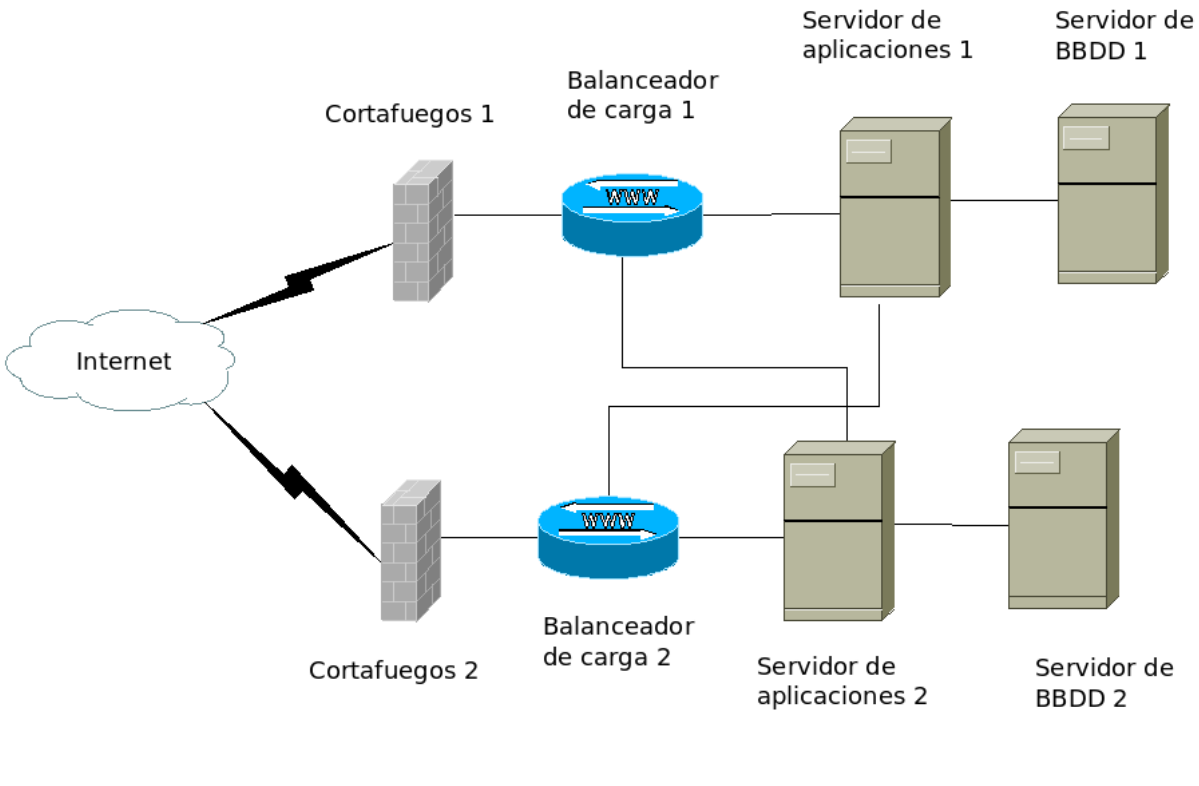


Fig. 6.1 Ejemplo de infraestructura

Recomendaciones de seguridad

- Las conexiones con los servidores de producción deben estar cifradas, se utilizarán protocolos https, ssh, ftps...
- Se recomienda la instalación de todas las actualizaciones de seguridad de cualquier componente del software utilizado.
- Se debe utilizar una política de contraseñas fuertes con fecha de caducidad.
- A nivel de sistema operativo, se eliminará cualquier servicio o programa que no sea necesario.
- Deberán revisarse periódicamente los logs del sistema operativo, en particular aquellos relativos a los accesos y autorizaciones.

Política de copias de respaldo

- Periódicamente se realizarán copias de seguridad, se deberá utilizar un sistema abuelo-padre-hijo con copias diarias, semanales y mensuales.
- También periódicamente, deberán realizarse simulaciones de restauración de los datos a partir de las copias de seguridad, con el objetivo de comprobar que las copias de seguridad son correctas.

Monitorización del sistema

Es importante monitorizar periódicamente el sistema para comprobar el correcto funcionamiento o

detectar posibles fallos. Las medidas de actuación deben incluir:

- Monitorización del sistema operativo: se puede utilizar las utilidades propias del sistema operativo linux: comandos como `top`, `free`, `sar`, `vmstat`, `netstat`, `pmap`, `mpstat`... nos permiten monitorizar en tiempo real el sistema. También dispone de un sistema de logs donde se recoge información de cualquier incidencia del sistema.
- Monitorización del servidor de aplicaciones Tomcat. Tomcat dispone de herramientas propias para la monitorización[8] y de otras open-source desarrolladas por terceros. Se puede encontrar una descripción en: https://wiki.apache.org/tomcat/FAQ/Monitoring
- Monitorización de la base de datos MySQL. Existen herramientas como **MyTop**[48], que funciona de forma similar al comando top de Linux y nos muestra información del rendimiento de MySQL: http://jeremy.zawodny.com/mysql/mytop/

También se puede utilizar herramientas de alerta y monitorización que cubran toda la infraestructura. Dentro del open-source disponemos de **Nagios**[49] que dispone de plugins con los que podemos monitorizar la infraestructura completa:, redes, nodos, sistemas operativo, servidor de aplicaciones, sistema gestor de base de datos. Además, nos permite configurar alertas en caso de fallo de algún componente del sistema.
https://www.nagios.org/

6.3 Aceptación del sistema

Para la aceptación del sistema por parte del cliente, deberá hacerse entrega tanto del código fuente como de la documentación del mismo. Además, el cliente deberá realizar pruebas similares a las pruebas de integración definidas en el punto 3.6. Tras las mismas, podrá realizarse la firma del acta de recepción.

7 MANTENIMIENTO

La resolución de cualquier error detectado en la aplicación correrá a cargo de la empresa desarrolladora, estando incluida su resolución en la garantía del sistema.

La aplicación pretende ser un desarrollo inicial sobre el que añadir nuevas funcionalidades. Si se desea la incorporación de nuevas funcionalidades o la modificación de las existentes, se ofrece dos posibilidades:

- Realizar un un contrato de mantenimiento evolutivo, donde la misma empresa desarrolladora se encargue de los nuevos trabajos. También se puede incluir las tareas de administración y monitorización del sistema.
- Contratar una formación en la que la empresa desarrolladora forme sobre las técnicas de desarrollo y administración de la aplicación, capacitándoles para hacerse cargo de las tareas de mantenimiento y evolución del software.

8 CONCLUSIONES

Este proyecto se ha realizado como parte del Máster Universitaro en Software Libre de la UOC en colaboración con la empresa Implemental Systems S.L. La idea surgió porque la empresa había desarrollado un proyecto similar para uno de sus clientes utilizando la tecnología de desarrollo nativa para la plataforma Android. Se deseaba disponer de la misma aplicación en otras plataformas.

Con el uso de la tecnología Apache Cordova, hemos conseguido migrar con éxito la aplicación de forma sencilla, utilizando lenguajes estándares en el desarrollo web y sin requerir un esfuerzo de aprendizaje de nuevas tecnologías. El hecho de utilizar software libre, ha permitido mantener los costes bajos. Además, al existir una gran comunidad online, ha sido sencillo encontrar en internet la solución a cualquier dificultad surgida en el desarrollo, sin necesidad de recurrir a soporte de pago de terceras empresas.

En relación con los contenidos del máster, he puesto en práctica conocimientos adquiridos tanto en el área de especialización como en el resto de asignaturas.

En concreto, respecto al desarrollo web he utilizado HTML, JavaScript, CSS, java, servidores de internet, servicios web, Spring como framework de desarrollo y APIs como las de Google Maps.

Dentro del ámbito de las bases de datos, he instalado, diseñado e implementado una base de datos siguiendo el modelo entidad-relación y utilizando como sistema gestor de base de datos a MySQL

En temas de ingeniería del software he realizado actividades como la planificación, el análisis o la toma de requisitos y he hecho uso del lenguaje de modelado UML, de herramientas para documentar el código y métodos de distinto tipo de pruebas.

También se han incluido tareas relacionadas con la administración de sistemas Linux. Todo el desarrollo se ha realizado utilizando únicamente el sistema operativo GNU-Linux, y las operaciones de instalación y configuración del equipo de desarrollo han sido similares a las que se realizan en un servidor de producción .

Incluso se han aplicado algunos conocimientos relativos a la implantación de sistemas ya que se han dado unas recomendaciones de infraestructura, instalación y políticas de seguridad y copias de respaldo.

Como profesional del sector, el proyecto me ha permitido mejorar mi desempeño como desarrollador de software vinculado principalmente a proyectos de internet y me ha adentrado en el desarrollo de aplicaciones para dispositivos móviles, de lo cual hay gran demanda en el mundo empresarial.

El desarrollo de la aplicación móvil pretende ser un desarrollo inicial que sirva de ejemplo para crear un aplicación con Apache Cordova y sobre el que poder añadir nuevas funcionalidades. Con el fin de facilitar la adaptación de la aplicación a distintas empresas clientes, se ha mantenido un aspecto lo más limpio posible, y la comunicación entre la aplicación móvil y el servidor se ha realizado mediante servicios web usando el API REST.

Comparando el resultado con la aplicación nativa original, da la sensación de ser algo más lenta, no obstante, en dispositivos con hardware más moderno, pasa inadvertido. También el hecho de usar la librería JQuery Mobile para el desarrollo de los interfaces, hace a la aplicación algo más pesada y ocupa más tamaño que una aplicación nativa.

La experiencia con el framework java Spring ha sido desigual. Por una parte el framework simplifica el desarrollo con respecto a otras especificaciones, sin embargo requiere de un esfuerzo importante de aprendizaje y configuración, y aunque ofrece componentes para cualquier

problemática de proyectos web, resulta tentador utilizar formas de resolver el problema más clásicas, sin aprovechar las ventajas del framework. Sí que me ha agradado el echo de poder ejecutarse en un servidor sencillo y ligero como Tomcat y que no consume grandes recursos en comparación con otras tecnologías puramente java EE con las que tengo experiencia.

La utilización de Openshift me ha permitido disponer de toda una plataforma de desarrollo, incluyendo un sistema gestor de versiones y un servidor online de forma gratuita y compatible con las herramientas de software libre. Su configuración e integración con el entorno local de desarrollo ha sido sencilla, ahorrándome tiempo de configuración. Solo he echado de menos poder disponer de una herramienta online de desarrollo, aunque según sus últimas noticias están trabajando junto con la empresa Codenvy, en un entorno de programación de estas características.

Como trabajo futuro, la aplicación móvil se puede ampliar con múltiples funcionalidades, por ejemplo, algunos centros académicos realizan prácticas presenciales, se podría incluir los lugares de realización de estas actividades. Igualmente se podrían añadir otro tipo de funcionalidad como las fechas de próximas entregas de trabajos a realizar y sus calificaciones, acceso a foros de consulta, avisos de noticias importantes etc.

Igualmente, se pueden añadir muchas funcionalidades a la aplicación web. En la versión actual, solo se ha incluido los casos de uso más básicos para dar soporte a la aplicación móvil, se podría haber añadido funcionalidades para introducir toda la información necesaria en la base de datos como: asignaturas, alumnos matriculados, profesores asignados, reparto de alumnos por examen, etc. La parte web solo pretendía ser una demostración, ya que esta parte es menos adaptable a otros clientes porque depende de los sistemas de información de los que dispongan.

Durante el desarrollo del proyecto también he visto que existen otros frameworks para el desarrollo de aplicaciones móviles híbridas. Por ejemplo el framework Ionic basado en Cordova y AngularJS que está ganando gran popularidad, sería interesante hacer un desarrollo y compararlo con los resultados obtenidos. En los próximos años seguramente surjan nuevos frameworks y formas de desarrollo para móviles y asistiremos a la decadencia y consolidación de algunos de ellos.

También considero el cloud computing y el uso de software online como una de las tecnologías con más desarrollo potencial, ya que simplifica todo el proceso de configuración de equipos y permite acelerar los desarrollos.

REFERENCIAS

[1] **Android SDK:** Herramientas para el desarrollo de apliacaciones para Android
http://developer.android.com/sdk/index.html

[2] **Appcelerator Titanium**: Framework open-source de desarrollo de aplicaciones móviles usando JavaScript
http://www.appcelerator.com/

[3] **Apache Cordova**: Herramienta de desarrollo de aplicacions móviles a partir de código HTML5, CSS3 y JavaScript
https://cordova.apache.org/

[4] **Apache HTTP Server**: Servidor web de la fundación de software Apache
https://httpd.apache.org/

[5] **Apache JMeter**: Software open-source para la realización de pruebas de cargas
http://jmeter.apache.org/

[6] **Apache Maven**: herramienta para automatizar la construcción de proyectos de software, especialmente los creados con java
https://maven.apache.org/

[7] **Apache Tomcat:** servidor de aplicaciones web que cumple con varias de las especificaciones incluidas en Java EE.
http://tomcat.apache.org/

[8] **Apache Tomcat Monitorización:** Wiki para la monitorización del servidor Apache Tomcat
https://wiki.apache.org/tomcat/FAQ/Monitoring

[9] **Apache Tomcat 7, Api de referencia**
https://tomcat.apache.org/tomcat-7.0-doc/api/

[10] **Bootstrap**: Framework el diseño de páginas web adaptables a distintos tamaños de pantalla
http://getbootstrap.com/

[11] **CentOS**: Distribución de Linux
https://www.centos.org/

[12] **Cross-site scripting**. Técnica de inyección de código para atacar vulnerabilidades en sitios web.
https://en.wikipedia.org/wiki/Cross-site_scripting

[13] **CSS**: Es un lenguaje de hojas de estilo para describir la presentación de un documento HTML.
http://www.w3.org/Style/CSS/

[14] **Debian**: Distribución de linux.
https://www.debian.org/

[15] **Doxygen**: Herramienta para genaración de documentación de código
http://www.stack.nl/~dimitri/doxygen/

[16] **Eclipse:** entorno de desarrollo integrado open-source para distintos lenguajes de programación.
https://eclipse.org/

[17] **Git**: Sistema de control de versiones distribuido.

https://git-scm.com/

[18] **GNU Free Documentation License (GFDL):** Licencia de tipo copyleft diseñada por la Free Software Foundation
http://www.gnu.org/licenses/fdl-1.3.html

[19] **GNU/Linux**: Sistema operativo de software libre
http://www.linux.org/

[20] **Google Maps API**: API de la empresa Google para la utilización de su servicio de mapas.
https://developers.google.com/maps/

[21] **Google Trends**: Herramienta de la empresa Google para comparar el interés de términos de búsqueda.
https://www.google.com/trends/

[22] **Hibernate**: Framework ORM (object-relational mapping) escrito en java.
http://hibernate.org/

[23] **Hibernate JPA, Api de Referencia**
http://docs.jboss.org/hibernate/jpa/2.1/api/

[24] **Hibernate ORM 4.3, documentación**
http://hibernate.org/orm/documentation/4.3/

[25] **HTML**: Lenguaje de marcado para la creación de páginas web
http://www.w3.org/TR/html5/

[26] **IOS**: Sistema operativo de la empresa Apple Inc. para dispositivos móvile.
http://www.apple.com/ios/

[27] **Java**: Lenguaje de programación orientado a objetos diseñado para ejecutarse en cualquier dispositivo.
https://www.java.com/es/

[28] **Java 7, API de referencia**:
http://docs.oracle.com/javase/7/docs/api/

[29] **Javadoc**: Herramienta para la generación de documentación del código java, está incluida en el JDK
http://www.oracle.com/technetwork/articles/java/index-137868.html

[30] **Java EE (Java Enterprise Edition)**: Plataforma java para el desarrollo de aplicaciones empresariales
http://www.oracle.com/technetwork/java/javaee/overview/index.html

[31] **Java EE 7 API de referencia**
https://docs.oracle.com/javaee/7/api/

[32] **Java OpenJDK**: Versión libre de las herramientas para desarrollar en java.
http://openjdk.java.net/

[33] **JavaScript**: Lenguaje de programación interpretado por los navegadores web.
https://developer.mozilla.org/en-US/docs/Web/JavaScript

[34] **JBoss Developer Studio**: entorno de desarrollo integrado basado en Eclipse:
http://www.jboss.org/products/devstudio/overview/

[35] **JPA Java Persistence API**: API de persistencia para la plataforma Java EE:
http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html

[36] **JQuery**: Librería con el objeto de simplificar el desarrollo en JavaScript.
https://jquery.com/

[37] **Jquery Mobile:** Framework de desarrollo para aplicaciones de dispositivos móviles basado en JQuery.
https://jquerymobile.com/

[38] **JSDoc 3**: Herramienta para la documentación de código JavaScript.
https://github.com/jsdoc3/jsdoc

[39] **Json:** Formato estándar y abierto para intercambiar datos.
http://www.json.org/

[40] **JUnit**. Framework para el desarrollo de pruebas unitarias para el lenguaje de programación Java.
http://www.linux.org/

[41] **Linux Mint**: Distribución del sistema operativo GNU-Linux orientada a equipos de escritorio
http://www.linuxmint.com/

[42] **MariaDB**: Fork de MySQL surgido tras de adquisición de MySQL por Oracle
https://mariadb.org/

[43] **Mit License:** Licencia de software libre creada por el Massachusetts Institute of Technology (MIT)
https://opensource.org/licenses/MIT

[44] **MongoDB**: Sistema gestor de base de datos NoSQL
https://www.mongodb.org/http://www.appcelerator.com/

[45] **MySQL**: Sistema gestor de bases de datos open-source.
https://www.mysql.com/

[46] **MySQL, API de referencia**
http://dev.mysql.com/doc/refman/5.5/en/

[47] **MySQL Workbench**: Entorno visual de desarrollo y administración para MySQL
https://www.mysql.com/products/workbench/

[48] **Mytop**: Herramienta de monitorización para MySQL
http://jeremy.zawodny.com/mysql/mytop/

[49] **Nagios**: Herramienta de monitorización de sistemas
https://www.nagios.org/

[50] **.NET:** Framework de desarrollo de la empresa Microsoft
https://www.microsoft.com/net

[51] **NPM**: sistema gestor de paquetes javascript:
https://www.npmjs.com/

[52] **OpenShift**: Plataforma en la nube de la empresa Red Hat para el desarrollo de software usando tecnologías open-source.
https://www.openshift.com

[53] **Oracle Database**: Sistema gestor de base de datos de la empresa Oracle
https://www.oracle.com/database/

[54] **PhoneGap**: Herramienta de desarrollo de aplicacions móviles de Adeobe y basada en Apache Cordova
http://phonegap.com

[55] **PhoneGap Explained Visually**
http://phonegap.com/2012/05/02/phonegap-explained-visually/

[56] **PhoneGap, Cordova, and what's in a name?**
http://phonegap.com/2012/03/19/phonegap-cordova-and-what%E2%80%99s-in-a-name/

[57] **PHP**: Lenguaje de programación de uso general del lado del servidor.
http://www.php.net/

[58] **PostgreSQL**: Sistema gestor de base de datos open-source
http://www.postgresql.org/

[59] **Red Hat Inc**: Empresa desarrolladora de software open-source
http://www.redhat.com/

[60] **Red Hat Entreprise Linux**: Distribución de Linux
https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux

[61] **REST (Transferencia de Estado Representacional)**: estilo de arquitectura para componentes distribuidos
http://www.w3.org/TR/ws-arch/#relwwwrest

[62] **Selenium**: Framework para la realización de pruebas de aplicaciones Web
http://www.seleniumhq.org/

[63] **Spring Framework**: Framework open source para aplicaciones java.
https://spring.io/

[64] **Spring Framework, referencia**:
http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/

[65] **SQL Injection**: Técnica de inyección de código para atacar vulnerabilidades en sistemas que utilizan SQL
https://en.wikipedia.org/wiki/SQL_injection

[66] **Stanford Javascript Crypto Library (SJCL)**: Librería para el uso de criptografía desde JavaScript
https://crypto.stanford.edu/sjcl/

[67] **Swift**: Lenguaje de programación para la plataforma IOS
https://developer.apple.com/swift/

[68] **UML (Unified Modelling Language).** Lenguaje de modelado de propósito general para la ingeniería del software
http://www.uml.org/

[69] Ubuntu Server**.** Distribución de Ubuntu Linux orientada a servidores
http://www.ubuntu.com/download/server

[70] **Windows Phone**. Sistema operativo de la empresa Microsoft para teléfonos móviles
https://www.microsoft.com/

[71] **World Wide Web Consortitun**: Organización para la generación de estándares de la WWW
http://www.w3.org/