Sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje móvil

Proyecto de Fin de Carrera

INGENIERÍA TÉCNICA INFORMÁTICA DE SISTEMAS

Febrero de 2015

Autor

Diana García Díaz

Tutor

Dr. D. Francisco José García Peñalvo

Cotutor

Dr. D. Miguel Ángel Conde González

Dr. D. Francisco José García Peñalvo, profesor del Departamento de Informática y Automática de la Universidad de Salamanca y Dr. D. Miguel Ángel Conde González, profesor del Departamento de Ingenierías Mecánica, Informática y Aeroespacial de la Universidad de León.

CERTIFICAN:

Que el trabajo titulado "Sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje móvil" ha sido realizado por Dña. Diana García Díaz, con DNI 70937842T y constituye la memoria del trabajo realizado para la superación de la asignatura Proyecto Fin de Carrera de la Titulación Ingeniería Técnica Informática de Sistemas de esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca, a 11 de Febrero de 2015

Dr. D. Francisco José García Peñalvo
Dpto. Informática y Automática
Universidad de Salamanca

Dr. D. Miguel Ángel Conde González
Dpto. Ingenierías Mecánica, Informática y Aeroespacial
Universidad de León

Tabla de contenidos

1. INTRODUCCIÓN 1

1.1. ORGANIZACIÓN DE LA MEMORIA 3

1.2. DOCUMENTACIÓN TÉCNICA 3

2. OBJETIVOS DEL PROYECTO 5

2.1. OBJETIVOS SOFTWARE 5

2.2. OBJETIVOS TÉCNICOS 6

2.3. OBJETIVOS PERSONALES 6

3. CONCEPTOS TEÓRICOS 7

3.1. E-LEARNING 7

3.2. M-LEARNING 8

3.3. LMS 9

3.3.1. Moodle 12

3.4. PLE 13

3.5. INTEROPERABILIDAD 14

3.6. SERVICIOS WEB 15

3.6.1. REST 15

4. TÉCNICAS Y HERRAMIENTAS 17

4.1. PROCESO UNIFICADO 17

4.2. MDB - MÉTODO DE DURÁN Y BERNÁRDEZ 19

4.3. UML - UNIFIED MODELING LANGUAGE 21

4.4. LENGUAJES DE PROGRAMACIÓN 24

4.4.1. Java 24

4.4.2. XML 25

4.4.3. PHP 25

4.4.4. SQL 26

4.4.5. JSON 27

4.4.6. HTML 28

4.5. HERRAMIENTAS 28

4.5.1. REM 28

4.5.2. Android 30

4.5.3. SDK - SOFTWARE DEVELOPMENT KIT 32
4.5.4. XAMPP 32
4.5.5. Visual Paradigm 35
4.5.6. StarUML *36*
4.5.7. Eclipse *36*
4.5.8. Google API Chart *37*
4.5.9. TCPDF *38*
4.5.10. Microsoft Office Word *38*
5. DESCRIPCIÓN DEL PROYECTO 39
5.1. APLICACIÓN ANDROID: PLE 39
5.2. FICHERO DE LOG 41
5.3. SERVICIO WEB DE INTERACCIÓN CON MOODLE 41
5.4. HERRAMIENTA PARA EL CONSUMO DE LA INFORMACIÓN DESDE MOODLE 41
5.5. HERRAMIENTA PARA LA EXPLOTACIÓN Y TOMA DE DECISIONES 46
6. ASPECTOS RELEVANTES 47
6.1. ANÁLISIS 47
6.2. DISEÑO 51
6.3. IMPLEMENTACIÓN 53
6.3.1. Aplicación Android *53*
6.3.2. Servicio web *55*
6.3.3. Bloque de Moodle *56*
6.4. PRUEBAS 58
7. TRABAJOS RELACIONADOS 59
7.1. IMPLEMENTACIÓN DE UN FRAMEWORK DE SERVICIOS PARA FACILITAR LA INTEROPERABILIDAD ENTRE MOODLE Y UN PLE BASADO EN WIDGETS 59
7.2. PERSONALIZACIÓN DEL APRENDIZAJE: FRAMEWORK DE SERVICIOS PARA LA INTEGRACIÓN DE APLICACIONES ONLINE EN LOS LMS 59
7.3. CLIENTE ANDROID PARA MOODLE 60
7.4. APLICACIÓN PARA REALIZAR CUESTIONARIOS DESDE DISPOSITIVOS ANDROID 60
8. CONCLUSIONES 61
9. LÍNEAS FUTURAS DE TRABAJO 63
10. LISTA DE ACRÓNIMOS 65
11. BIBLIOGRAFÍA 67

Lista de Figuras

FIGURA 1. DESPLIEGUE DEL PROYECTO 2
FIGURA 2. DIAGRAMA DE CASOS DE USO 20
FIGURA 3. DIAGRAMA DE COMPONENTES DE MOODLE 22
FIGURA 4. INTERFAZ REM....29
FIGURA 5. VERSIONES DE ANDROID 31
FIGURA 6. DATOS DE USO DE LAS VERSIONES ANDROID....32
FIGURA 7. INTERFAZ XAMPP....33
FIGURA 8. INTERFAZ PHPMYADMIN....34
FIGURA 9. INTERFAZ VISUAL PARADIGM 35
FIGURA 10. INTERFAZ STAR UML 36
FIGURA 11. INTERFAZ ECLIPSE CON PLUGIN ADT 37
FIGURA 12. INTERFAZ PLE MÓVIL....39
FIGURA 13. PANTALLA PARA INTRODUCIR LOS DATOS DEL USUARIO 40
FIGURA 14. LISTADO DE APLICACIONES INSTALADAS EN EL DISPOSITIVO 40
FIGURA 15. CAPTURA DEL BLOQUE EN MOODLE 41
FIGURA 16. “TODAS LAS ENTRADAS” 42
FIGURA 17. “REGISTROS DE HOY” 43
FIGURA 18. FORMULARIO PARA GESTIONAR UN INFORME 43
FIGURA 19. FORMULARIO RELLENADO....45
FIGURA 20. INFORME GESTIONADO....45
FIGURA 21. PDF INFORME GESTIONADO 46
FIGURA 22. PAQUETES DE CASOS DE USO 48
FIGURA 23. DIAGRAMA DE CLASES DEL DOMINIO DE LA APLICACIÓN ANDROID 48
FIGURA 24. DIAGRAMA DE SECUENCIA 49
FIGURA 25. DIAGRAMA DE ACTIVIDAD....50
FIGURA 26. DIAGRAMA DE NAVEGACIÓN....50
FIGURA 27. PATRÓN MVC EN UN PROYECTO DE ANDROID....51
FIGURA 28. ESCENARIO AUTORIZAR EL USO DE SUS DATOS....52
FIGURA 29. EJEMPLO FICHERO DE LOG 54
FIGURA 30. FUNCIÓN WSPLECONNECTION 55

FIGURA 31. INTERFAZ PLE EN MODO LANDSCAPE.....58
FIGURA 32. INTERFAZ DE LA APLICACIÓN DE CUESTIONARIOS: DE DIOS, 2012.....59
FIGURA 33. INTERFAZ INICIAL DE LA APLICACIÓN DEL PROYECTO: DE LUIS, 2012.....60
FIGURA 34. INTERFAZ CUESTIONARIO DE APLICACIÓN DEL PROYECTO: BOUAYAD, 2013.....60

Lista de Tablas

TABLA 1. PLANTILLA PARA ACTORES – ACTOR PROFESOR 21

TABLA 2. TABLA PLE 52

1. Introducción

A lo largo del tiempo los procesos de aprendizaje, al igual que otros ámbitos, se ven influenciados por el contexto que los rodea, es decir, por las tendencias pedagógicas, sociológicas, tecnológicas, políticas, etc. En este sentido cabe destacar la aparición de las Tecnologías de la Información y la Comunicación (TIC) (*Peñalvo, 2005*).

El empleo de las TIC, y especialmente de Internet, en el ámbito de la educación supone una revolución en la forma en que se enseña y se aprende. Han aparecido un gran número de herramientas *software* que facilitan la gestión y el desarrollo de actividades de aprendizaje haciendo que los alumnos pasen a tener un mayor protagonismo dentro de su formación (*Conde, 2012*). Pero estos avances pueden traer consigo una serie de problemas, como puede ser una sobrecarga en el número de herramientas que un alumno tenga que utilizar en su formación, provocando que ese gran número de posibles contextos de aprendizaje se convierta en un problema. Además, gran parte del aprendizaje que lleva a cabo un alumno se realiza de manera invisible al docente, sin que incluso el propio alumno se dé cuenta de que muchas de las acciones que realiza también forman parte del mismo, como puede ser la consulta de manuales o foros donde puede acudir para resolver ciertas dudas *(de Dios, 2012)*.

Estos problemas quedan en parte solucionados gracias a las plataformas de aprendizaje (*Learning Management System,* o LMS) que centralizan gran parte de los recursos necesarios para un alumno, dándoles un lugar común donde compartir sus dudas y poder seguir el curso, y además proporcionan una serie de herramientas a los profesores con los que gestionar y calificar a los alumnos. Pero estos LMS se centran en el curso, con lo que encajan con un modelo educativo basado en la clase en la que el profesor imparte unos contenidos, establecidos por un plan de estudios, a un conjunto de estudiantes, con un nivel y un ritmo de aprendizaje artificialmente uniformes (*de Dios, 2012*).

Para los estudiantes los LMS suponen espacios concretos en los que poder llevar a cabo sus actividades lectivas o con los que se complementan sus clases. Sin embargo, estos entornos no resuelven problemas como los mencionados anteriormente, ya que no están centrados en el usuario sino en la institución y en el curso, no ofrecen soporte al aprendizaje a lo largo de la vida y no dan soporte a la incorporación de nuevas tendencias tecnológicas.

Por estas razones, se hacen necesarios entornos más adaptados a sus necesidades, abiertos a la inclusión de todo tipo de tecnologías innovadoras y de las herramientas que ellos utilizan para aprender. Además, en estos contextos, se debe considerar un cambio de enfoque, ya que, las herramientas no van a estar tan focalizadas en el curso o la institución sino que se centran en los gustos y necesidades específicas del estudiante. (Conde, 2012) Estas nuevas plataformas son los Entornos Personalizados de Aprendizaje (*Personal Learning Environment*, PLE) *(Wilson et al., 2007)*. Espacios que buscan facilitar el aprendizaje al usuario al permitir que este utilice aquellas herramientas que considere oportunas para aprender (normalmente con las que están familiarizados), sin estar vinculados a un entorno institucional concreto o a un período de tiempo específico *(Adell & Castañeda, 2010).*Con los PLE el discente pasa a ser el responsable de su propio aprendizaje, al poder determinar qué herramientas va a usar para aprender, al pasar de ser consumidor a proveedor de aprendizaje, y al aprender al relacionarse con los demás, pero según sus necesidades específicas, etc. *(Adell &*

*Castañeda, 2010; Schaffert & Hilzensauer, 2008).*Pero los PLE no son un sustituto de los LMS, sino que pueden actuar como un complemento que dota de una nueva concepción del aprendizaje.

Este proyecto va a perseguir la realización de un sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje (PLE) móvil, que permitirá recuperar las peculiaridades de las herramientas incluidas en el PLE y de la actividad del usuario en ellas. Esta información se enviará a un LMS, en este caso *Moodle* (http://moodle.org/), desde el que se hará una representación flexible de la misma que permitirá al profesor estudiar la influencia de la actividad en el móvil en las notas de los estudiantes y que facilite la toma de decisiones.

Las herramientas a implementar para llevar a cabo el proyecto serán:

- **Aplicación *Android*** en la que se definirá un PLE móvil. Se creará una aplicación *Android* en la que se recuperará un listado de las aplicaciones que estén instadas en el dispositivo móvil y permitirá a los usuarios incluir una serie de aplicaciones que podrán organizar a su gusto en su dispositivo móvil. Además, en esta aplicación se deberá implementar un sistema de log en el que se almacene la actividad llevada a cabo en el PLE móvil y una comunicación con el LMS basada en servicios web.
- **Servicio web** para la interacción con el entorno institucional, que en este caso es *Moodle*. Este servicio va a ser el encargado de insertar la información que reciba de la aplicación *Android* en una tabla de la base de datos de *Moodle*.
- **Bloque en *Moodle***. Se creará un bloque en el LMS que recuperará la información sobre la actividad llevada a cabo en el PLE móvil a través de consultas a la base de datos de *Moodle*. Este bloque permitirá al profesor la visualización de informes en los que se representará gráficamente y en una tabla dicha información. Además, el profesor podrá exportar dichos informes a formato PDF.

En la Figura 1 se muestra un ejemplo del despliegue del proyecto:

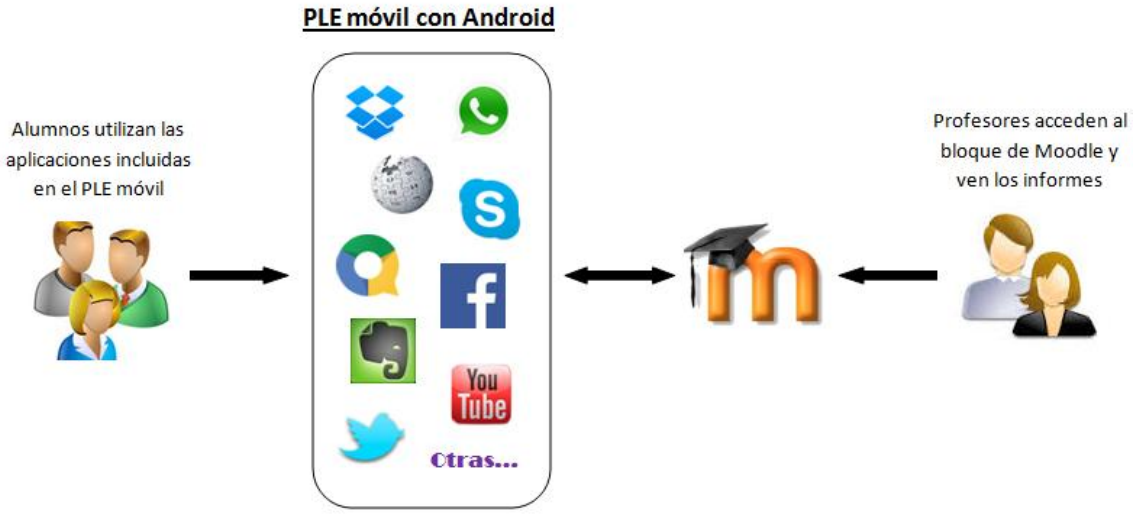


Figura 1. Despliegue del proyecto

1.1. Organización de la memoria

Para realizar la memoria del proyecto, se han seguido las pautas propuestas en "Proyecto de Final de Carrera en la Ingeniería Técnica en Informática: Guía de Realización y Documentación" *(Peñalvo, Raedo, Velthuis, Giner, &García, 2000)*. La estructura de la memoria es la siguiente:

Introducción, donde se hace una breve presentación del tema que se aborda en el proyecto, a la vez que se hace una breve descripción del contenido y estructura de la memoria.

Objetivos del proyecto, se explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto. Además, se hace una distinción entre objetivos software, técnicos y personales.

Conceptos teóricos, en el que se definen una serie de conceptos para que se pueda comprender los posteriores requisitos marcados para las aplicaciones.

Técnicas y herramientas, se presentan las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto.

Descripción del proyecto, se hace una descripción detallada del proyecto que se ha realizado.

Aspectos relevantes del proyecto, recoge los aspectos más interesantes del desarrollo del proyecto. Incluye una exposición del ciclo de vida utilizada y detalles de mayor relevancia de las fases de análisis, diseño e implementación.

Conclusiones y líneas de trabajo futuras, conclusiones técnicas y posibles líneas futuras por las que se podrían mejorar el proyecto.

Lista de acrónimos, donde se listan todos los acrónimos que aparecen en la memoria del proyecto.

Bibliografía

1.2. Documentación técnica

En este apartado de la memoria se indica el conjunto de anexos que constituyen la documentación técnica del proyecto.

Anexo 1. *Planificación del Proyecto*.

La documentación de este apartado está dirigido hacia dos puntos concretos: la planificación temporal del proyecto y el estudio de viabilidad económica del mismo.

Anexo 2. *Especificación de requisitos del software*

En este anexo se recogen todos los requisitos del sistema software a construir. Comprende la definición y la especificación de los requisitos del software, siguiendo la Metodología para la Elicitación de Requisitos de Sistemas Software.

Anexo 3. *Especificación de diseño*

Este documento se corresponde con la fase de diseño y supone la entrada en el dominio de la solución del problema, es decir, comienza la fase tradicional del

desarrollo del sistema software. En él se incluye el diseño arquitectónico, el diseño de las interfaces y el modelo de datos.

Anexo 4. *Documentación técnica de programación*

Este anexo se corresponde con la fase de implementación. Recoge la documentación de las bibliotecas, la estructura de ficheros del código fuente y el manual del programador para futuras ampliaciones y mantenimiento de los componentes.

Anexo 5. *Manuales de usuario*

En este anexo se recogen todos los tutoriales, manuales y guías de usuario necesarios para el correcto manejo de la aplicación *Android* y del bloque de *Moodle*. Además, se especifica la manera para poder instalar todas las partes del proyecto en el ordenador de explotación.

2. Objetivos del proyecto

El objetivo general del proyecto es la **implementación de un sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje (PLE) móvil** que será integrado en un entorno institucional como el LMS.

2.1. Objetivos software

Los objetivos software en los que se divide la realización de este proyecto son los siguientes:

1) **Definir un contenedor de aplicaciones educativas en el móvil:** Este objetivo consistirá en la creación de una aplicación para dispositivos móviles con sistema operativo *Android* en la que los usuarios podrán utilizar distintos tipos de aplicaciones en actividades de aprendizaje. Esta aplicación permitirá reunir un total de 8 aplicaciones en una única interfaz en la que los usuarios podrán incluir las aplicaciones instaladas en el dispositivo que deseen en un entorno personalizado de aprendizaje (PLE) móvil y utilizarlas de forma anónima o autenticada.
2) **Establecer un modelo de gestión y almacenamiento de la actividad:** En este objetivo, el presente proyecto pretende facilitar un fichero de log de la actividad llevada a cabo dentro del contexto de PLE móvil. De esta manera, se almacenará información de los distintos tipos de aplicaciones incluidas en el PLE móvil y el uso de los usuarios en ellas en dicho fichero. En concreto, se almacenará información sobre el usuario (nombre y DNI de usuario) que hace uso del PLE móvil, las aplicaciones que utiliza (nombre de estas) y cuándo (fecha y hora de uso).
3) **Construir un sistema de interacción con un entorno institucional basado en servicios web:** En este objetivo, se pretende crear un sistema de interacción con el entorno institucional. Para llevar a cabo esta tarea, se creará un servicio web que será el encargado de recibir los datos que le envíe el cliente, la aplicación *Android*, y los insertará en una tabla en la base de datos de *Moodle* creada para tal fin.
4) **Definir una herramienta que permita la consumición de la información desde el entorno institucional:** Esta parte del proyecto consistirá en la creación de un bloque en *Moodle* desde el que el profesor podrá consumir la información sobre la actividad llevada a cabo en el PLE móvil a través de distintos informes y gráficos. Además, dicho bloque va a permitir al profesor filtrar la información, dependiendo de lo que desee visualizar en cada momento.
5) **Definir una herramienta que facilite la explotación de esa información y la toma de decisiones:** El último objetivo del proyecto consistirá en añadir una funcionalidad al bloque de *Moodle* mediante la que el profesor pueda exportar los distintos informes que genere sobre la actividad llevada a cabo en el PLE móvil a formato PDF.

2.2. Objetivos técnicos

Los objetivos de carácter técnico planteados a la hora de desarrollar el proyecto son:

- **Aprendizaje del desarrollo de aplicaciones Android:** Necesario para el desarrollo de la aplicación *Android* en la que se definirá el entorno personalizado de aprendizaje (PLE) móvil y se creará el fichero de log. Este objetivo también implica el aprendizaje del lenguaje de programación **Java** necesario para realizar la lógica de negocio de las aplicaciones *Android* y **XML** para definir las interfaces.
- **Aprendizaje de PHP:** Necesario para la creación del servicio web, así como para la creación del bloque para *Moodle.*
- **Aprendizaje de HTML:** Necesario para mostrar la información desde el bloque de *Moodle*.
- **Aprendizaje de SQL:** Necesario para hacer las inserciones y consultas en la base de datos de *Moodle.*
- **Aprendizaje de Servicios web:** Necesarios para la creación del servicio web para la interacción con *Moodle.* Además se deberá trabajar desde la parte del cliente con el lenguaje de programación **Java**.
- **Aprendizaje de REST y JSON:** Necesarios para realizar la comunicación con el servicio web.
- **Aprendizaje de Google API Charts:** Se hará uso de Google API Charts para la creación de los gráficos que representarán la información llevada a cabo en el PLE móvil en el bloque de *Moodle*.
- **Aprendizaje de TCPDF:** Biblioteca utilizada para la exportación de los informes que se generen desde el bloque de Moodle a formato PDF.

2.3. Objetivos personales

La realización del proyecto de final de carrera es una de las primeras oportunidades con las que cuenta un estudiante de una Ingeniería Técnica en Informática para poner en práctica los conocimientos adquiridos durante la carrera y, al mismo tiempo, enfrentarse a innumerables tecnologías y herramientas nuevas. Dicho esto, los objetivos personales del proyecto son:

- Aplicar de forma integrada los conceptos teóricos y prácticos obtenidos en el conjunto de asignaturas cursadas durante la carrera.
- Aprendizaje de nuevas tecnologías necesarias para el desarrollo del proyecto. Esto será de gran ayuda de cara al futuro laboral, ya que las tecnologías de las que se ha hecho uso para realizar este proyecto son muy utilizadas actualmente en el mercado profesional.
- Complementar alguna de las líneas trabajo futuras descritas en trabajos anteriores, como el Proyecto Fin de Carrera de Alberto de Dios: "Implementación de un framework de interoperabilidad entre PLE y Moodle" (*de Dios, 2012*).

3. Conceptos teóricos

Este apartado contiene los conceptos teóricos necesarios para la comprensión y el desarrollo del presente proyecto.

3.1. E-Learning

El concepto *e-learning* se puede definir de muchas formas diferentes debido, fundamentalmente, a que los actores que hacen uso de él son muy diversos. Si se toma como referencia la raíz de la palabra, *e-learning* se traduce como "aprendizaje electrónico", y como tal, en su concepto más amplio puede comprender cualquier actividad educativa que utilice medios electrónicos para realizar todo o parte del proceso formativo *(Peñalvo, 2005)*.

La formación virtual o *e-learning* suele definirse de manera simple como un proceso de aprendizaje con TIC (especialmente mediante el uso de redes como Internet). Sin embargo, la formación virtual de calidad supone mucho más que tecnología. Circunscribirse exclusivamente al factor tecnológico del *e-learning* supone centrarse sólo en la "e" inicial del término y, con frecuencia, abocarse al fracaso de la formación precisamente por olvidarnos del elemento crucial: el aprendizaje efectivo por parte de los usuarios. Así pues, más que una cierta tecnología, el *e-learning* es una metodología o modo de aprendizaje que presupone el uso de determinada tecnología como medio de formación. La formación virtual es evidentemente tecnológica, pero no es sólo (sino también) tecnología *(Pardo & Peñalvo, 2015)*.

Las principales ventajas que ofrece el *e-Learning* son las siguientes (*e-Learning, 2015)*:

- **Elimina las distancias físicas.** Se utilizan herramientas como correo electrónico, foro o chat para establecer la comunicación entre los participantes.
- **Se alternan diversos métodos de enseñanza.** Los participantes pueden trabajar individualmente o de manera grupal.
- **Permite flexibilidad horaria.** El alumno accede en el momento que dispone de tiempo.
- **Aumenta el número de destinatarios.** Esta modalidad de formación se puede dirigir a una audiencia mucha más amplia.
- **Favorece la interacción.** Los alumnos pueden comunicarse unos con otros, con el tutor y con los recursos on-line disponibles en Internet.
- **Disposición de recursos *on-line* y multimedia.** Internet proporciona acceso instantáneo e ilimitado a una gran cantidad de recursos, como textos, gráficos, audio, vídeo, animaciones, etc.
- **Facilita el mantenimiento, actualización y distribución de contenidos.** Dado el carácter digital que asumen los contenidos es más fácil mantener estos actualizados, cambiarlos, incluir nuevos elementos para soportar el aprendizaje y que los contenidos puedan ser consumidos y consultados con mayor facilidad.

3.2. M-Learning

Una de las tecnologías que con mayor fuerza ha irrumpido en la sociedad actual son los dispositivos móviles, tales como: teléfonos móviles, *smartphones*, *tablets*, videoconsolas portátiles, reproductores mp3, *ebooks*, y todo dispositivo de mano que tenga alguna forma de conectividad inalámbrica. Como es evidente esta tecnología también influye en los procesos de enseñanza/aprendizaje lo que origina lo que se conoce como *mLearning* y que consiste en usar los dispositivos móviles para aprender.

De esta manera, podemos resumir el *mLearning* como *eLearning* a través de dispositivos móviles, con las ventajas y libertades que éstos proporcionan. Las principales características del *mLearning* son las siguientes (*Conde, 2012*):

- **Mayor tiempo útil.** El tiempo que le puede dedicar una persona para llevar a cabo actividades de enseñanza y aprendizaje se incrementa dada la facilidad de acceso que provee el dispositivo móvil a los contenidos, lo que evita desplazamientos y la necesidad de un PC con conexión a Internet.
- **Disponibilidad geográfica.** Gracias a la portabilidad de los dispositivos móviles y facilidad de integración en entornos físicamente dispersos, se garantiza que un usuario en condiciones de ubicación y conectividad muy limitadas pueda acceder al aprendizaje.
- **Autonomía y personalización.** Los dispositivos móviles facilitan los procesos autónomos y orientados de aprendizaje hacia necesidades concretas del estudiante. Este puede tener un mayor control y conocimiento del dispositivo que de un entorno que le sea ajeno, además es posible personalizarlo según sus necesidades y acceder a la formación en cualquier momento y lugar.
- **Adaptación a la ubicación física.** Algunos dispositivos móviles incorporan elementos GPS (Global Positioning System), que posibilitan determinar la posición del estudiante, y, en función de esta, proporcionar contenidos formativos adecuados para enriquecer al usuario en el contexto en que se encuentran.
- **Necesidad de conexión.** Los dispositivos móviles permiten acceder a datos disponibles en la Red y, por lo tanto, a contenidos formativos dispuestos en otros entornos. Además, la tecnología facilita que el usuario pueda trabajar sin conexión con lo que en caso de no disponer de ese acceso a datos, igualmente se podrían realizar ciertas actividades formativas.
- **Incremento en los servicios para móviles.** El aprendizaje mediante los dispositivos móviles se aprovecha del incremento exponencial en cuanto a los servicios y aplicaciones que existen para ese tipo de dispositivos de cara a enriquecer el proceso de aprendizaje.
- **Pequeña curva de aprendizaje.** Muchas personas ya están habituadas a usar y a consumir servicios a través de terminales móviles, lo que ayuda a reducir los periodos de formación en el uso de la tecnología que da soporte al proceso formativo.

3.3. LMS

Un LMS es un software basado en un servidor web que provee módulos para los procesos administrativos y de seguimiento que se requieren para un sistema de enseñanza, simplificando el control de estas tareas. Los módulos administrativos permiten, por ejemplo, configurar cursos, matricular alumnos, registrar profesores, asignar cursos a un alumno, llevar informes de progreso y calificaciones. También facilitan el aprendizaje distribuido y colaborativo a partir de actividades y contenidos preelaborados, de forma síncrona o asíncrona, utilizando los servicios de comunicación de Internet como el correo, los foros, las videoconferencias o el *chat* (*Peñalvo, 2005*).

Actualmente existe un gran número de LMS, tanto comerciales como de código abierto. Para el caso que nos ocupa en este proyecto, se ha utilizado *Moodle* que es una plataforma de licencia libre que se está implantando con mucha fuerza y es usado ampliamente en la Universidad de Salamanca.

Un LMS debe ofrecer todo tipo de servicios para gestionar de forma eficiente la actividad de aprendizaje, es por ello que cualquier LMS se caracteriza por un conjunto de propiedades básicas y unas funcionalidades.

Se considera que un LMS debe tener las siguientes **propiedades** (*Conde, 2012*):

- **Personalización.** Conseguir que la persona que usa la plataforma tenga conciencia de que es el protagonista de su formación.
- **Interactividad y usabilidad.** La plataforma debe plantear herramientas que faciliten la interacción de los usuarios con otros usuarios y con la propia plataforma. Esta interactividad debe ser amigable, de forma que sea sencillo acceder a los recursos necesarios para la formación.
- **Flexibilidad.** Conjunto de funcionalidades que permiten que la plataforma tenga una adaptación fácil en la organización en la que se quiere implantar. Esta adaptación se puede dividir en los siguientes puntos:
    - Capacidad de adaptación a la estructura de la institución.
    - Capacidad de adaptación a los planes de estudio de la institución donde se quiere implantar el sistema.
    - Capacidad de adaptación a los contenidos y estilos pedagógicos de la organización.
- **Escalabilidad.** Capacidad de la plataforma de aprendizaje de funcionar igualmente con un número pequeño o grande de usuarios.
- **Estandarización.** Hablar de plataformas estándares es hablar de la capacidad de utilizar cursos realizados por terceros; de esta forma, los cursos están disponibles para la organización que los ha creado y para otras que cumplen con el estándar. También se garantiza la durabilidad de los cursos, al evitar que estos queden obsoletos y, por último, se puede realizar el seguimiento del comportamiento de los estudiantes dentro del curso.
- **Portabilidad.** No solamente los contenidos deberían seguir estándares, sino también la forma en que se almacena la información de los usuarios y/o los

cursos, de manera que sea posible portar esta información a otras plataformas de aprendizaje.

- **Interoperabilidad.** Debe facilitarse la comunicación de las plataformas de aprendizaje con otras herramientas. Esta propiedad es opcional, pero es algo que se está demandando desde la irrupción del 2.0.
- **Seguridad.** Las contraseñas, y así como cualquier información sensible, debe almacenarse en la plataforma encriptada para evitar el acceso a la misma por personal no autorizado.
- **Internalización o arquitectura multi-idioma.** La interfaz de la plataforma debe estar traducida, o se debe poder traducir fácilmente, para que los usuarios se familiaricen con ella de una forma sencilla.
- **Amplia comunidad de usuarios y documentación.** La plataforma debe contar con el apoyo de comunidades dinámicas de usuarios, con foros orientados a cada uno de los roles que se relacionan con ella (usuarios, desarrolladores, expertos, etc.).
- **Soporte a diversas modalidades de *eLearning*.** Las plataformas de aprendizaje deben proporcionar soporte a modalidades formativas ya sean estas totalmente *online*, presenciales o mixtas.

Por otro lado, también debe presentar al menos las siguientes **funcionalidades** (*Conde, 2012*):

- **Visualización, acceso e interacción con los contenidos.** El usuario tiene que ser capaz de visualizar la estructura de los cursos, sus contenidos, sus actividades, su actividad personal, etc.
- **Gestión de usuarios.** Es necesario que el sistema permita el registro de usuario, actualización, matriculación en los itinerarios formativos, baja, etc.
- **Gestión de itinerarios de aprendizaje.** Cada estudiante puede tener asignado uno o varios itinerarios formativos en función de sus características específicas.
- **Gestión de cursos.** Posibilidad de dar alta curso, incluir recursos y actividades en ellos, asignar profesores, asignar formatos de cursos, etc.
- **Sistemas de evaluación.** Debe posibilitarse al profesor y al propio estudiante llevar a cabo un seguimiento de las actividades realizadas en la plataforma. Es decir, que las actividades puedan ser evaluadas por los profesores o el sistema y proporcionar esa información a los estudiantes.
- **Sistemas de seguimiento y monitorización.** Tener la posibilidad de no solamente determinar qué ha hecho el estudiante a través de actividades de evaluación, sino también los recursos más visitados, número de *clics*, tiempo empleado en la realización del curso, etc.
- **Elaboración de informes.** Generación de informes a nivel de actividad del usuario, plataforma o curso, así como otras posibles temáticas.
- **Búsqueda inteligente de recursos.** Los usuarios deben disponer de motores de búsqueda que les permitan buscar entre los elementos incluidos en el LMS. Se

puede diferenciar diferentes niveles como recursos, actividades, *post*, cursos, módulos, etc.

- **Funcionalidades para la comunicación síncrona y asíncrona.** En una plataforma de aprendizaje es fundamental la comunicación entre los estudiantes y de estos con el profesor, es por ello que se debe permitir esta actividad tanto de forma síncrona como asíncrona.
- **Gestión de recursos humanos.** Algunos autores (Greenberg, 2002; Totkov, 2003; Watson & Watson, 2007) consideran que las plataformas deben soportar actividades relativas a los recursos humanos, como gestión de turnos, asignación de tutorías, reserva de aulas, gestión presupuestaria, gestión de certificados, etc.
- **Gestión y evaluación de competencias.** Gestión de competencias que permitan identificar las necesidades de los usuarios y asignarles formación en consecuencia.

Además de las propiedades y funcionalidades anteriormente citadas, son muchas las herramientas que se incluyen en cualquier LMS. A continuación se citan las categorías básicas de **herramientas** (*Conde, 2012*):

- **Herramientas de administración.** Se corresponden a diferentes tipos de gestiones relativos a la administración de la plataforma. Se incluyen: gestión de usuarios, página personal, gestión de cursos, gestión de la plataforma, etc.
- **Herramientas de comunicación.** Se subdividen en síncronas o asíncronas. Como ejemplo: foros, *chat*, correo electrónico, comentarios, tablón de anuncios, etc.
- **Herramientas para dar soporte a otras modalidades de aprendizaje.** Herramientas que permitan la videoconferencia y audioconferencia, la compartición de documentos y el modo presentación.
- **Herramientas de participación.** Herramientas que implican la colaboración de diferentes estudiantes, como pueden ser herramientas de trabajo en grupo, *blogs*, *wikis*, comunidades virtuales, etc.
- **Herramientas de gestión de actividades.** Herramientas que facilitan la realización de actividades como agenda, tareas, ejercicios, etc.
- **Herramientas de contenido.** Herramientas para la creación de contenidos, publicación en los cursos, compartición, etc.
- **Herramientas de evaluación y seguimiento.** Herramientas que permiten observar qué ha hecho un usuario y cuándo, su grado de actividad e implicación. También en este apartado se incluyen herramientas para gestionar la evaluación de los usuarios dentro de un curso, gestionar sus notas, etc. De estas herramientas se pueden generar informes.
- **Herramientas orientadas a la productividad.** Herramientas que pueden ayudar a mejorar el desempeño del estudiante en la plataforma. Por ejemplo: anotaciones (*bookmarks*), calendario, buscador de cursos, mecanismos para facilitar el trabajo *offline*, control de publicaciones, avisos, soporte a sindicación de contenidos, etc.

- **Herramientas de soporte.** Necesarias para la realización de actividades de formación que facilitan la labor de los actores del proceso de aprendizaje. Por ejemplo: sistemas para la autenticación, registro, ayuda, buscadores de recursos, sistemas de auditoría, asignación de permisos, definición de roles, etc.
- **Sistemas de gestión del conocimiento en el ámbito educativo.** Herramientas que hacen énfasis en diferentes aspectos para gestionar el conocimiento. Por ejemplo: sistemas de gestión de competencias, de conocimientos, de repositorio, herramientas de consulta y recomendación, etc.
- **Herramientas para la integración.** Herramientas para facilitar la integración de elementos externos como sistemas de *portfolio*, repositorios de objetos de aprendizaje, comunicación con otras herramientas, etc.
- **Herramientas para la personalización.** Permiten entregar la información, evaluaciones, actividades, etc. que ha sido personalizada para los distintos individuos según sus necesidades particulares.

3.3.1. Moodle

Como ejemplo de LMS se puede poner Moodle, una plataforma de aprendizaje diseñada para proporcionar a los educadores, administradores y estudiantes un sistema integrado único, robusto y seguro para crear ambientes de aprendizaje personalizados (*Moodle, 2015a*).

La palabra Moodle era al principio un acrónimo de *Module Object-Oriented Dynamic Learning Environment* (Entorno Modular de Aprendizaje Dinámico Orientado a Objetivos).

Las razones por las cuales se ha escogido Moodle como LMS de referencia en este proyecto son las siguientes:

- Los números de Moodle a nivel mundial se sitúan en más de 68 millones de usuarios, tanto a nivel de uso académico como a nivel empresarial, lo que la convierte en la plataforma de aprendizaje más ampliamente utilizada del mundo (*Moodle, 2015b*).
- Está diseñado para soportar tanto la enseñanza como el aprendizaje. Con más de 10 años de desarrollo, Moodle proporciona un conjunto poderoso de herramientas centradas en el estudiante y ambientes de aprendizaje colaborativo, que le dan poder, tanto a la enseñanza como al aprendizaje.
- Es fácil de aprender y usar. Posee una interfaz simple, características de arrastrar y soltar, y recursos bien documentados, junto con mejoras en usabilidad.
- Moodle es proporcionado gratuitamente como programa de código abierto, bajo la Licencia Pública General GNU (*GNU General Public License*). Cualquier persona puede adaptar, extender o modificar Moodle, tanto para proyectos comerciales como no comerciales, sin pago de cuotas por

licenciamiento, y beneficiarse del costo/beneficio, flexibilidad y otras ventajas de usar Moodle.

- Moodle está siempre actualizado, la implementación de Moodle en código abierto significa que es continuamente revisado y mejorado para adecuarse a las necesidades actuales y cambiantes de sus usuarios.
- Las capacidades multilingües de Moodle aseguran que no haya limitaciones lingüísticas para aprender en línea. Moodle se ha traducido a más de 120 idiomas (y sigue aumentando), para que los usuarios puedan adaptar su sitio Moodle al idioma local o nacional, junto con muchos recursos, soporte y discusiones comunitarias disponibles en varios idiomas.
- Debido a que es código abierto, Moodle puede ser personalizado en cualquier forma deseada, para adecuarlo a necesidades individuales. Su configuración modular y diseño inter-operable permite a los desarrolladores crear plugins e integrar aplicaciones externas para lograr funcionalidades específicas.

3.4. PLE

Los PLE son sistemas que ayudan a los estudiantes a tomar el control y gestión de su propio aprendizaje. Esto incluye el apoyo a los estudiantes a:

- Fijar sus propios objetivos de aprendizaje
- Gestionar su aprendizaje, la gestión de los contenidos y procesos
- Comunicarse con otros en el proceso de aprendizaje

Las principales características de un PLE son (*Conde, 2012*):

- **Agregación.** Uso de herramientas que permitan la recopilación de materiales de diferentes tipos para su utilización como fuentes en el proceso de aprendizaje.
- **Organización.** Herramientas que permiten realizar acciones sobre los contenidos como la clasificación, agrupación, diferenciación, extracción de metadatos, etc.
- **Edición/Creación de contenidos.** Herramientas para poder crear y modificar contenidos para ponerlos disponibles para otros usuarios.
- **Propiedad y protección de datos.** Determinar cómo el PLE contempla la temática de los derechos de autor de los contenidos.
- **Difusión/Socialización.** Cómo el PLE facilita al usuario la difusión de los datos y la conexión con herramientas sociales que permitan su enriquecimiento y planteamientos constructivistas.
- **Personalización.** Referida a la capacidad que el usuario tiene de modificar el entorno según sus necesidades o preferencias, de forma que se centre en el usuario y no en un entorno global impuesto por la institución.
- **Escalabilidad.** Capacidad del entorno personalizado para ampliarse y evolucionar ante las nuevas realidades que aparezcan.

- **Control.** Responsabilidad pedagógica en el proceso de aprendizaje, es decir, quién determina cómo se aprende, con qué herramientas, etc.
- **Conectividad.** Capacidad para conectarse con otros entornos, contextos e instituciones. Por ejemplo, si un PLE puede conectarse con un LMS, con un repositorio de contenidos o puede accederse a través de un dispositivo móvil. También se contempla aquí el modo en que se accede al PLE en sí, si es a través del escritorio, de la Web, etc.
- **Interoperabilidad.** Capacidad del PLE de interactuar con otros sistemas sin necesidad de complejas adaptaciones.
- **Relaciones.** Tipo de relación de los usuarios del PLE con otros usuarios.

Los tipos de elementos que compongan el PLE determinan en gran medida la potencialidad de este y la forma cómo los usuarios interactúan con el mismo.

Desde una perspectiva tecnológica se consideran tres grupos de herramientas que debe tener un PLE básico (*Conde, 2012*):

- **De acceso a la información:** sitios de publicación, repositorios y bases de datos de audio, vídeo, multimedia, objetos de aprendizaje estandarizados, lectores de RSS, sitios de noticias, portales de información específica, etc.
- **De creación y edición de información:** *Wikis*, *suites* ofimáticas de escritorio y en red, herramientas de mapas mentales, herramientas de edición de audio, de vídeo, creación de presentaciones, mapas conceptuales, cronogramas y en general cualquier tipo de artefacto informacional.
- **De relación con otros:** herramientas de red social o de las que emerge una red. Dentro de estas se dividen en función de su propósito:
    - La relación se da a través de objetos de información (textos, vídeos, fotografías, etc.) que publican los usuarios y en donde el interés radica en aprender de dichos objetos.
    - La relación se basa en la comunicación de lo que se hace y se aprende fuera del entorno, esto es, se pone el énfasis en compartir sitios, experiencias y recursos para aprender.
    - Relaciones con otras personas, de forma que el aprendizaje es producto de la interacción entre personas.

3.5. Interoperabilidad

Según el IEEE (**I***nstitute of* ***E****lectrical and* ***E****lectronic* ***E****ngineers*), se entiende como **interoperabilidad** la habilidad de dos o más sistemas o componentes para intercambiar información y usar esta información, es decir, la habilidad de trabajar juntos.

En los últimos años, en la sociedad de la información, ha crecido la necesidad de utilizar la interoperabilidad de cara a ofrecer un mejor servicio.

Las ventajas que ofrecen los sistemas interoperables son:

- Reusabilidad: Capacidad de un programa para que partes de él puedan volver a ser utilizadas con otros propósitos.
- Independencia tecnológica
- Posibilidad de crear sistemas centralizados que comparten información.

3.6. Servicios web

La W3C (http://www.w3c.org/) define "Servicios Web" (en inglés, *Web Service* o *Web services*) como un conjunto de aplicaciones o tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios, de manera que, los proveedores ofrecen sus servicios como procedimiento remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web (*W3C, 2015*).

Como norma general, el transporte de los datos se realiza a través del protocolo HTTP y la representación de los datos mediante XML (*Calle, 2009*). Sin embargo, no hay reglas fijas en los servicios web y en la práctica no tiene por qué ser así. De hecho, para el presente proyecto se ha utilizado JSON (http://www.json.org/) en lugar de XML.

El uso de servicios web aporta un gran número de ventajas a los sistemas que los usan, algunas de ellas son:

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los Servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Se pueden utilizar con HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante firewalls que filtran y bloquean gran parte del tráfico de Internet, cierran casi todos los puertos TCP salvo el 80, que es, precisamente el que usan los navegadores. Los servicios Web utilizan este puerto por la simple razón de que no resultan bloqueados. Es importante señalar que los servicios web se pueden utilizar sobre cualquier protocolo, sin embargo, TCP es el más común.
- Pueden aportar gran independencia entre la aplicación que usa el servicio web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más utilizada.

3.6.1. REST

REST (REpresentational State Transfer) es un estilo de arquitectura de software para sistemas distribuidos en la red. REST fue introducido en la tesis doctoral de Roy

Fielding en el año 2000, uno de los principales autores de las especificaciones del protocolo HTTP (*Fielding, 2000*).

REST se refiere, en el más estricto sentido, a una colección de principios de arquitecturas de sistemas en red acerca de cómo se definen los recursos y cómo se accede a ellos. Aunque REST es un estilo arquitectónico en el desarrollo de software, se usa muchísimo en el contexto de los WebServices.

En un servicio web REST, las peticiones habitualmente codifican los parámetros en la URL. El resultado de una petición a un servicio REST es la información en formato XML o en formato JSON (http://www.json.org/) sin ningún tipo de información adicional (como ocurre en otros servicios web).

El servicio web REST ha sido utilizado en este proyecto para realizar las peticiones al servicio en Java desde la aplicación Android mediante las cuales se envían los parámetros que correspondan. En cuanto al formato de datos utilizado para el resultado de estas peticiones al servicio REST ha sido JSON.

4. Técnicas y herramientas

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. El objetivo no es analizar detenidamente cada una de ellas, sino hacer una pequeña introducción de las mismas y comentar sus aspectos más destacados. Dicho esto, las técnicas y herramientas que se han utilizado en el desarrollo del proyecto son las siguientes:

4.1. Proceso Unificado

Para la realización de las distintas partes de este proyecto se han seguido las pautas del Proceso Unificado, debido a su estudio y utilización en la asignatura de Ingeniería del Software durante la carrera.

El Proceso Unificado es más que un simple proceso *(Jacobson et al., 1999)*, es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

Características generales:

- Está **basado en componentes**: El sistema software en construcción está formado por componentes software interconectados a través de interfaces.
- **Utiliza UML** para preparar todos los esquemas de un sistema software. De hecho, UML, es una parte esencial del Proceso Unificado (sus desarrollos fueron paralelos).

Características principales:

- Es un proceso **conducido por casos de uso**: Los casos de uso guían el desarrollo del sistema. Como los casos de uso contienen las descripciones de las funciones, afectan a todas las fases y vistas.
- Está **centrado en la arquitectura**: La arquitectura se representa mediante vistas del modelo.
- Es **iterativo e incremental**: En cada iteración se identifican y especifican los casos de uso relevantes, se crea un diseño basado en la arquitectura seleccionada, se implementa el diseño mediante componentes y se verifica que los componentes satisfacen los casos de uso. Si una iteración cumple con los objetivos establecidos en función de la evaluación de las iteraciones precedentes, se pasa a la siguiente. En cada iteración se va desarrollando el sistema de forma incremental.

Los elementos del Proceso Unificado son:

- **Fases**: Es preciso diferenciar temporalmente las fases del ciclo de vida. La división temporal necesita puntos de control.
- **Puntos de control o hitos**: Separan las etapas, las fases (hitos principales), las iteraciones (hitos secundarios). Los participantes en el proyecto revisan el progreso del proyecto.

- **Disciplinas o flujos de trabajo**: Organizan las actividades fundamentales de gestión y desarrollo. Se pueden solapar en el tiempo. El resultado de las actividades de los flujos de trabajo son los artefactos.
- **Artefactos**: Cualquier tipo de información producida por los desarrolladores de un sistema (diagramas UML, código, ejecutables, casos de prueba,…). Se construyen de forma incremental.

El Proceso Unificado se repite a lo largo de una serie de ciclos de desarrollo que constituyen la vida del sistema. Estos ciclos se dividen en dos etapas bien diferenciadas, la etapa de Ingeniería y la etapa de Producción, cada una de las cuales está formada por dos fases.

En total, el Proceso Unificado consta de cuatro fases que dividen temporalmente el proceso: inicio, elaboración, construcción y transición. Las dos primeras comprenden la etapa de Ingeniería y las otras dos la etapa de Producción.

Dentro de cada fase se puede, a su vez, descomponer el trabajo en iteraciones con sus incrementos resultantes. Cada fase termina con un hito. Las iteraciones discurren a lo largo de las disciplinas que, al contrario de lo que ocurre con las fases, se pueden solapar en el tiempo. Estas disciplinas son cinco: especificación de requisitos, análisis del sistema, diseño del sistema, implementación y pruebas.

A continuación, se presenta el listado de las tareas realizadas durante el desarrollo del proyecto siguiendo las pautas del Proceso Unificado:

1. **Especificación de requisitos**

   - Reunión con el cotutor.
   - Obtención/elicitación de requisitos.
   - Elaboración del documento de requisitos.
   - Validación de los requisitos por parte del cotutor.
   - Modificación del documento de requisitos.
   - Entrega del documento de requisitos.

2. **Análisis del sistema**

   - Enfoque de síntesis propio del Proceso Unificado:
     - Inicio por la funcionalidad: Casos de uso del sistema.
     - Refinamiento por la información: Diagramas de clases.
     - Consolidación por la funcionalidad: Diagramas de secuencia.
   - Entrega del documento de análisis del sistema.

3. **Diseño del sistema**

   - Diseño arquitectónico
   - Diseño de datos
   - Diseño de componentes
   - Diseño de la interfaz

- Entrega del documento de diseño del sistema.

4. **Implementación**
   - Implementación de prototipos: aplicación Android, servicio web y bloque de Moodle.
   - Implementación del sistema final: aplicación Android, servicio web y bloque de Moodle.
   - Entrega de implementación.
5. **Pruebas**
   - Pruebas de la aplicación Android en distintos dispositivos móviles con diferentes tamaños de pantalla y distintas versiones del sistema operativo.
   - Pruebas del servicio web desde un cliente PHP creado para comprobar el correcto funcionamiento del servicio web y finalmente desde el cliente Java (aplicación Android).
   - Pruebas del bloque de Moodle.

4.2. MDB - Método de Durán y Bernárdez

El método de Durán y Bernárdez *(Toro, 2000; Toro & Jiménez, 2000)* es una metodología para la obtención y documentación de los requisitos de sistemas de información desarrollada en el Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla. Es una metodología soportada por herramientas CASE.

El objetivo de esta metodología es la definición de las tareas a realizar, los productos a obtener y las técnicas a emplear durante la actividad de elicitación de requisitos de la fase de ingeniería de requisitos del desarrollo de software.

Las tareas recomendadas de esta metodología para la obtención de los productos descritos en esta metodología son las siguientes:

- Obtener información sobre el dominio del problema y el sistema actual.
- Preparar y realizar las reuniones de obtención/negociación.
- Identificar/revisar los objetivos del sistema.
- Identificar/revisar los requisitos de información.
- Identificar/revisar los requisitos funcionales.
- Identificar/revisar los requisitos no funcionales.
- Priorizar objetivos y requisitos.

Las técnicas recomendadas más importantes para llevar a cabo las tareas anteriores son los casos de uso, las plantillas y los patrones lingüísticos.

Por lo tanto, esta metodología ha sido utilizada en el presente proyecto para obtener y documentar los requisitos que debe cumplir el sistema.

Casos de uso

Los casos de uso son una técnica para la especificación de requisitos funcionales propuesta inicialmente por Ivar Jacobson e incorporada actualmente en UML *(Rumbaugh, Jacobson, & Booch, 1999).*

Los casos de uso no pertenecen estrictamente al enfoque orientado a objetos, son una técnica de captura de requisitos que cubren la carencia existente en métodos previos en cuanto a la determinación de requisitos. Están basados en el lenguaje natural, por lo que son accesibles por los usuarios y presentan ventajas sobre la descripción meramente textual de los requisitos funcionales.

Un caso de uso describe una interacción entre el sistema y uno o más actores (idealizaciones de personas, procesos o entidades externas que interaccionan con un sistema, subsistema o clase) en la que se considera al sistema como una caja negra y los actores obtienen resultados observables.

Los casos de uso se representan gráficamente en los denominados diagramas de casos de uso, que proporcionan una visión global del conjunto de casos de uso de un sistema, así como de los actores y casos de uso en los que éstos intervienen. En la Figura 2 se muestra un ejemplo de diagrama de casos de uso para este proyecto.

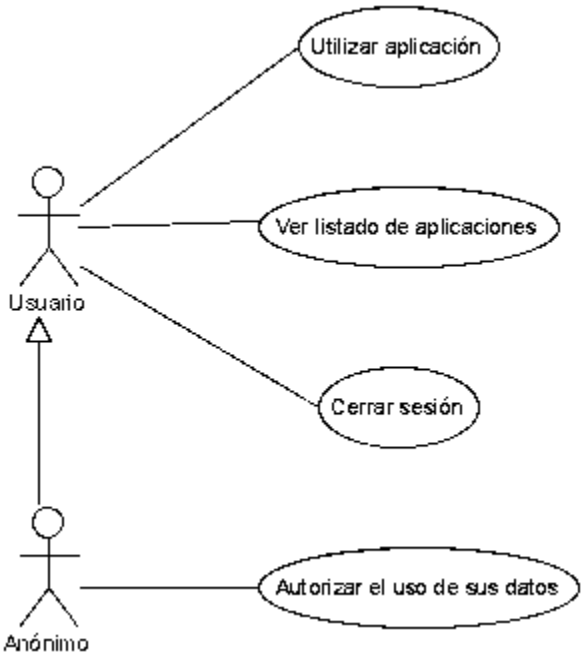


Figura 2. Diagrama de casos de uso para la exportación de la funcionalidad de la aplicación Android

Plantillas y patrones lingüísticos

Las plantillas son tablas utilizadas para la correcta descripción de los requisitos del sistema. Cada plantilla presenta los campos de información necesarios para especificar el concepto que se está representando. Hay distintos tipos de plantillas:

- Plantilla para los objetivos del sistema (OBJ).

- Plantillas para requisitos de información (IRQ y CRQ).
- Plantilla para actores (ACT).
- Plantilla para requisitos funcionales (CU).
- Plantilla para requisitos no funcionales (NFR).

En la Tabla 1 se muestra un ejemplo de la plantilla utilizada para la captura de los actores del sistema, concretamente del actor Profesor que va a ser el usuario del bloque de Moodle:

ACT-0004	**Profesor**
Versión	1.0 (01/03/2015)
Autores	• Diana García Díaz
Fuentes	• Francisco José García Peñalvo • Miguel Ángel Conde González
Descripción	Este actor representa *a un usuario con el rol de Profesor en el bloque de Moodle*
Comentarios	Ninguno

Tabla 1. Plantilla para actores – Actor Profesor

4.3. UML – Unified Modeling Language

UML es un lenguaje de modelado de objetos independiente del método que se implemente. Es un lenguaje para especificar, construir, visualizar y documentar ingenios software, resultando un lenguaje simple, común y ampliamente utilizable por los usuarios de otros métodos. UML es un lenguaje de modelado estándar, pero no es una metodología, método o proceso. Su objetivo es la unificación de los métodos de modelado de objetos. Define varios modelos para la representación de los sistemas que pueden verse y manipularse mediante un conjunto de diagramas diferentes.

UML define varios modelos de representación de los sistemas:

- Modelo de clases
- Modelo de estados
- Modelo de casos de uso
- Modelo de interacción
- Modelo de realización
- Modelos de despliegue

Los modelos son manipulados por medio de vistas que se clasifican en tres áreas *(Rumbaugh, Jacobson, & Booch, 1999)*:

- **Estructural:** Describe las cosas que hay en el sistema y sus relaciones con otras cosas.

- **Dinámica:** Refleja el comportamiento del sistema en el tiempo.
- **De gestión de modelos:** Describe la organización de los modelos en unidades jerárquicas.

Los diagramas que se pueden crear con UML son los siguientes (pueden verse algunos ejemplos de diagramas creados en los anexos 2 y 3 del proyecto para el modelado de la información en las figuras que se indican de esta memoria):

- **Diagrama de Clases:** Muestra una colección de elementos de modelado declarativo (estáticos), tales como clases, tipos y sus contenidos y relaciones. Véase apartado 6.1 - Figura 17: diagrama de clases de la aplicación Android.
- **Diagrama de Actividades:** Representa los procesos de negocio de alto nivel, incluido el flujo de datos. También puede utilizarse para modelar lógica compleja y/o paralela en un sistema. Véase apartado 6.1 - Figura 19: diagrama de actividades para el servicio web.
- **Diagrama de Secuencia:** Representa una interacción, poniendo el foco en la secuencia de los mensajes que se intercambian, junto con sus correspondientes ocurrencias de eventos en las líneas de vida. Véase apartado 6.1 - Figura 18: diagrama de secuencia.
- **Diagrama de Componentes**: Representa los componentes que componen una aplicación, sistema o empresa. Los componentes, sus relaciones, interacciones y sus interfaces públicas. En la Figura 3, se puede ver el diagrama de componentes de Moodle.

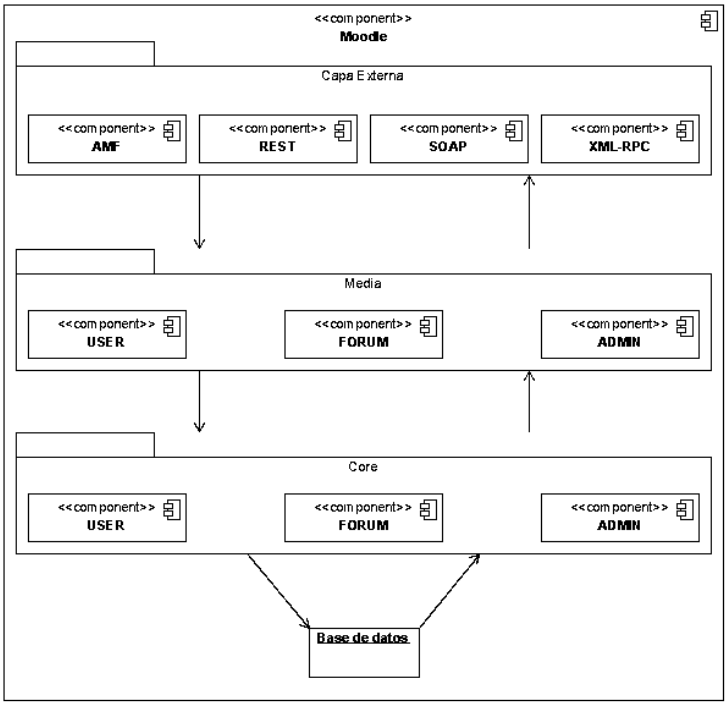


Figura 3. Diagrama de componentes de Moodle

- **Diagrama de Despliegue Físico:** Muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos y la construcción interna puede ser representado por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso las especificaciones de despliegue.
- **Diagrama de Máquinas de Estado:** Ilustra cómo un elemento, muchas veces una clase, se puede mover entre estados que clasifican su comportamiento, de acuerdo con disparadores de transiciones, guardias de restricciones y otros aspectos de los diagramas de Máquinas de Estados, que representan y explican el movimiento y comportamiento.
- **Diagrama de Casos de Uso:** Un diagrama que muestra las relaciones entre los actores y el sujeto (sistema), y los casos de uso. Véase apartado 4.2 - Figura 1: diagrama de casos de uso para la exportación de la funcionalidad de la aplicación Android.
- **Diagrama de Objetos:** Presenta los objetos y sus relaciones en un punto del tiempo. Se puede considerar como un caso especial de un diagrama de clases o de comunicaciones.
- **Diagrama de Paquetes:** Presenta cómo se organizan los elementos de modelado en paquetes y las dependencias entre ellos, incluyendo importaciones y extensiones de paquetes.
- **Diagrama de Revisión de la Interacción:** Enfocan la revisión del flujo de control, donde los nodos son interacciones u ocurrencias de interacciones. Las líneas de vida de los mensajes no aparecen en este nivel de revisión.
- **Diagrama de Comunicación:** Es un diagrama que enfoca la interacción entre líneas de vida, donde es central la arquitectura de la estructura interna y cómo ella se corresponde con el pasaje de mensajes. La secuencia de los mensajes se da a través de un esquema de numerado de la secuencia.
- **Diagrama de Estructura de Composición:** Representa la estructura interna de un clasificador (tal como una clase, un componente o un caso de uso), incluyendo los puntos de interacción de clasificador con otras partes del sistema.
- **Diagrama de Tiempos:** El propósito primario es mostrar los cambios en el estado o la condición de una línea de vida (representando una instancia de un clasificador o un rol de un clasificador) a lo largo del tiempo lineal. El uso más común es mostrar el cambio de estado de un objeto a lo largo del tiempo, en respuesta a los eventos o estímulos aceptados. Los eventos que se reciben se anotan, a medida que muestran cuándo se desea mostrar el evento que causa el cambio en la condición o en el estado.

Los elementos son los bloques básicos de UML y pueden ser:

- **Elementos de modelado**: Representan las abstracciones del sistema en curso de modelado. Son los conceptos utilizados en los diagramas, que representan los conceptos del paradigma objetual (clases, objetos, relaciones,…).

- **Elementos de visualización**: Son proyecciones textuales o gráficas que permiten la manipulación de los elementos de modelado.

Los bloques de construcción de UML tienen que combinarse siguiendo un conjunto de reglas que especifican un modelo bien formado, es decir, un modelo semánticamente autoconsistente y en armonía con todos sus modelos relacionados. Esto indica que el modelo o una parte suya se atiene a todas las reglas semánticas y sintácticas que le son de aplicación. UML tiene reglas para:

- **Nombres**: Cómo llamar a los elementos, relaciones y diagramas.
- **Alcance**: El contexto que da significado específico a un nombre.
- **Visibilidad**: Cómo se pueden ver y utilizar los nombres por otros.
- **Integridad**: Cómo se relacionan apropiada y consistentemente unos elementos con otros.
- **Ejecución**: Qué significa ejecutar o simular un modelo dinámico.

Además, UML tiene una serie de mecanismos que aseguran la integridad conceptual de la notación. Ofrecen comentarios extra, información o semántica sobre un elemento de modelado y los mecanismos de extensión a UML. Los mecanismos generales son:

- **Estereotipos**: Especializan las clases del metamodelo. Ej. `<<actor>>`
- **Etiquetas**: Extienden los atributos de las clases del metamodelo. Una etiqueta es un par (nombre, valor). Ej. {versión=2.0 autor=diana}
- **Restricciones**: Extienden la semántica del metamodelo. Ej. {ordenado}
- **Notas**: Comentarios vinculados a uno o más elementos de modelado.
- **Adornos:** La especificación de los elementos pueden incluir detalles gráficos o textuales adicionales. Ej. + ó #.

4.4. Lenguajes de programación

Se han utilizado gran variedad de lenguajes de programación para realizar la implementación de las diferentes partes del proyecto. A continuación, se explica cada uno de los lenguajes de programación utilizados.

4.4.1. Java

Java es un lenguaje de programación orientado a objetos desarrollado por *Sun Microsystems* que surgió a principios de los años 90 *(Oracle, 2012)*. Las aplicaciones de Java son generalmente compiladas a *bytecode* (clase Java) que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

Es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los

desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo, lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso.

La sintaxis de Java se deriva en gran medida de C++, pero a diferencia de éste, que combina la sintaxis para programación genérica, estructurada y orientada a objetos, Java fue construido desde el principio para ser completamente orientado a objetos. Todo en Java es un objeto (salvo algunas excepciones), y todo en Java reside en alguna clase.

Actualmente, el desarrollo de aplicaciones en *Android* solo está soportado en lenguaje Java, concretamente J2SE. Gracias a este lenguaje y mediante las APIs provistas por Google se pueden acceder a todos los recursos de los dispositivos móviles para crear las aplicaciones. Por este motivo, se ha utilizado Java en este proyecto para el desarrollo de la aplicación para *Android*.

4.4.2. XML

XML (*eXtensible Markup Language* ó Lenguaje de Marcado eXtensible), es un estándar del W3C (http://www.w3c.org) basado en etiquetas y similar a HTML.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

En este proyecto, se ha utilizado XML para definir las interfaces del PLE móvil, ya que Android provee un vocabulario XML que se corresponde con las clases y subclases utilizadas para definir los *layouts* de las aplicaciones.

4.4.3. PHP

PHP, acrónimo recursivo de *PHP: Hipertext Pre-processor*, es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y puede ser incrustado en HTML (*PHP, 2015*). Fue creado por Rasmus Lerdorf en 1994, sin embargo la implementación principal de PHP es producida ahora por *The PHP Group* y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la *PHP License*, la *Free Software Foundation* considera esta licencia como software libre.

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

- Soporte para una gran cantidad de bases de datos que se utilizan en la actualidad, entre las que destaca MySQL.
- Es un lenguaje multiplataforma.
- El código se pone al día continuamente con mejores y extensiones de lenguaje para ampliar las capacidades de PHP.

- El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Posee una amplia documentación en su página oficial (http://www.php.net/).
- Es libre, por lo que se presenta como una alternativa de fácil acceso.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que están manejando en tiempo de ejecución.

En el desarrollo de este proyecto se ha utilizado PHP 5 para la implementación del servicio web y del bloque de Moodle. Algunas de las características o cambios principales de esta versión respecto de la anterior son:

- Mejor soporte para la Programación Orientada a Objetos.
- Mejoras de rendimiento.
- Mejor soporte para MySQL.
- Mejor soporte a XML.
- Soporte nativo para SQLite.
- Iteradores de datos.
- Manejos de excepciones.
- Mejoras con la implementación con Oracle.

4.4.4. SQL

SQL (*Structured Query Language* o Lenguaje de Consultas Estructurado) (*W3Schools, 2015*), es un lenguaje estándar para acceder y manipular bases de datos. SQL está estandarizado por el ANSI (Instituto Americano de Normas) y por la ISO (Organización de Estándares Internacional), por lo tanto puede ser interpretado por cualquier motor de base de datos.

SQL permite:

- Ejecutar consultas contra una base de datos
- Recuperar datos de una base de datos
- Insertar registros en una base de datos
- Actualizar registros en una base de datos
- Eliminar registros en una base de datos
- Crear nuevas bases de datos
- Crear nuevas tablas en una base de datos
- Crear procedimientos almacenados en una base de datos
- Crear vistas en una base de datos

- Establecer permisos en tablas, procedimientos y puntos de vista

Las sentencias SQL pueden dividirse en tres tipos:

- **Sublenguaje de definición de datos** (**DDL**): proporciona órdenes para definir esquemas de relación, eliminar relaciones, crear índices y modificar esquemas de relación.
- **Sublenguaje de manipulación de datos** (**DML**):
  - Interactivo: lenguaje de consulta basado en el álgebra relacional y el cálculo relacional de tuplas. También incluye órdenes para insertar, suprimir y modificar tuplas de la base de datos.
  - Inmerso: lenguaje diseñado para utilizar dentro de los lenguajes de programación de propósito general (lenguajes anfitriones).
- **Sublenguaje de control de datos** (**DCL**): incluye órdenes que permiten especificar controles de seguridad a los datos almacenados como especificación de privilegios de acceso y control de concurrencia.

Se ha utilizado el lenguaje SQL en este proyecto para insertar los registros (parámetros enviados por la aplicación Android) en la base de datos de Moodle desde el servicio web y para ejecutar las consultas contra la base de datos de Moodle desde el bloque creado en dicha plataforma.

4.4.5. JSON

JSON (JavaScript Object Notation) (*JSON, 2015*) es un formato ligero de intercambio de datos. Es fácil para los seres humanos a leer y escribir y es fácil para las máquinas para analizar y generar. Se basa en un subconjunto del lenguaje de programación JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son familiares para los programadores de la C-familia de lenguajes, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen de JSON un lenguaje ideal para el intercambio de datos.

JSON es una forma de codificar objetos, arrays o cualquier otra serie de datos en un string y posteriormente poder descodificarlo sin muchos problemas. Esto es especialmente útil para enviar información entre cliente y servidor de una forma muy sencilla, puesto que cada componente descodifica la información según le convenga, indistintamente del resto del sistema. Originalmente estaba destinado ser usado para Javascript, pero cada vez más lenguajes lo soportan de forma nativa.

JSON está pensando para el tráfico de información por internet, minimizando el uso de caracteres para conseguir una codificación más pequeña. Se emplea en habitualmente en entornos donde el tamaño del flujo de datos entre cliente y servidor es de vital importancia y cuando la fuente de datos es de fiar. Permite que el procesado de los datos sea muy rápido, a diferencia de XML.

Por estas razones, se ha elegido este formato en el resultado de las peticiones al servicio web REST en Java desde la aplicación Android.

4.4.6. HTML

HTML (*Hyper Text Markup Language*) (*Raggett, 2005*) es el lenguaje utilizado por los navegadores web para presentar texto y gráficos. Es un lenguaje de marcas para describir documentos web (páginas web), es decir, documentos de texto presentado de forma estructurada, con enlaces (links) que conducen a otros documentos o a otras fuentes de información (por ejemplo bases de datos) que pueden estar en la propia máquina o en máquinas remotas de la red. Todo ello se puede presentar acompañado de gráficos estáticos o animados y sonidos.

La estética de los documento escritos en HTML no se limita a texto plano; consigue todos los efectos que habitualmente se pueden producir con un procesador de textos: negrita, cursiva, distintos tamaños y fuentes, tablas, párrafos tabulados, sangrías, incluso texto y fondo de página de colores, y muchos más.

La pieza clave es el "browser", "navegador", "visualizador" o "cliente" HTML. Todas las codificaciones de efectos en el texto que forman el lenguaje HTML son instrucciones para el visualizador. Esto explica el por qué no se ve lo mismo con todos los visualizadores, depende de cómo estén diseñados y para qué versión de lenguaje estén diseñados.

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser *Gedit* en *Linux*, el *Bloc de notas* de *Windows* o cualquier editor de textos que admita texto sin formato.

Los documentos HTML son descritos por etiquetas, cuya notación consiste en los símbolos < y > que encierran dentro una instrucción.

Este lenguaje se ha utilizado en el proyecto para presentar los gráficos y tablas con los datos en el bloque de Moodle.

4.5. Herramientas

En este apartado se definen todas las herramientas que se han utilizado para el desarrollo de las distintas partes del proyecto.

4.5.1. REM

Para la elicitación de los requisitas se ha utilizado la herramienta REM, que utiliza la metodología de Durán y Bernárdez, explicada en apartados anteriores. Esta metodología está enfocada a recoger los requisitos de un sistema y construir un documento, el "Documento de análisis del sistema".

REM (*REquirements Management*) es una herramienta experimental gratuita de gestión de requisitos diseñada para soportar la fase de ingeniería de requisitos de un proyecto de desarrollo software de acuerdo con la metodología definida en la Tesis Doctoral "Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información", presentada por Amador Durán en septiembre de 2000. La aplicación REM la podemos obtener desde la página de los profesores Amador Durán y Beatriz Bernárdez en la universidad de Sevilla, España *(Durán, Ruiz-Cortés, Corchuelo, &Toro, 2002; Sevilla, 2015).*

En esta herramienta se pueden distinguir los siguientes tipos de requisitos:

- **Objetivo:** No es un requisito en sí mismo, pero es muy útil a la hora de fijar los requisitos.
- **Actor:** Cada uno de los usuarios que interactúan con el sistema.
- **Requisito de almacenamiento de información**
- **Requisito de restricción**
- **Requisito de casos de uso**
- **Requisito funcional**
- **Requisito no funcional**
- **Matriz de rastreabilidad:** En estas matrices se pueden colocar en filas y columnas los diferentes requisitos, casos de uso, tipos (clases del modelo), valores (atributos) y asociaciones, permitiendo por ejemplo relacionar requisitos funcionales con tipos, para ver qué requisitos funcionales afectan a cada tipo concreto.

La elección de esta herramienta para la elicitación de requisitos fue sencilla, ya que REM ha sido la herramienta usada en todos los trabajos realizados en las distintas asignaturas de la carrera, por lo que se está bastante familiarizada con ella, a pesar de que su interfaz tiene ciertas limitaciones que hacen su uso un poco pesado en ciertas ocasiones. La versión utilizada para este proyecto de esta herramienta ha sido la 1.2.2.

En la Figura 4 puede verse una captura de la interfaz de REM.

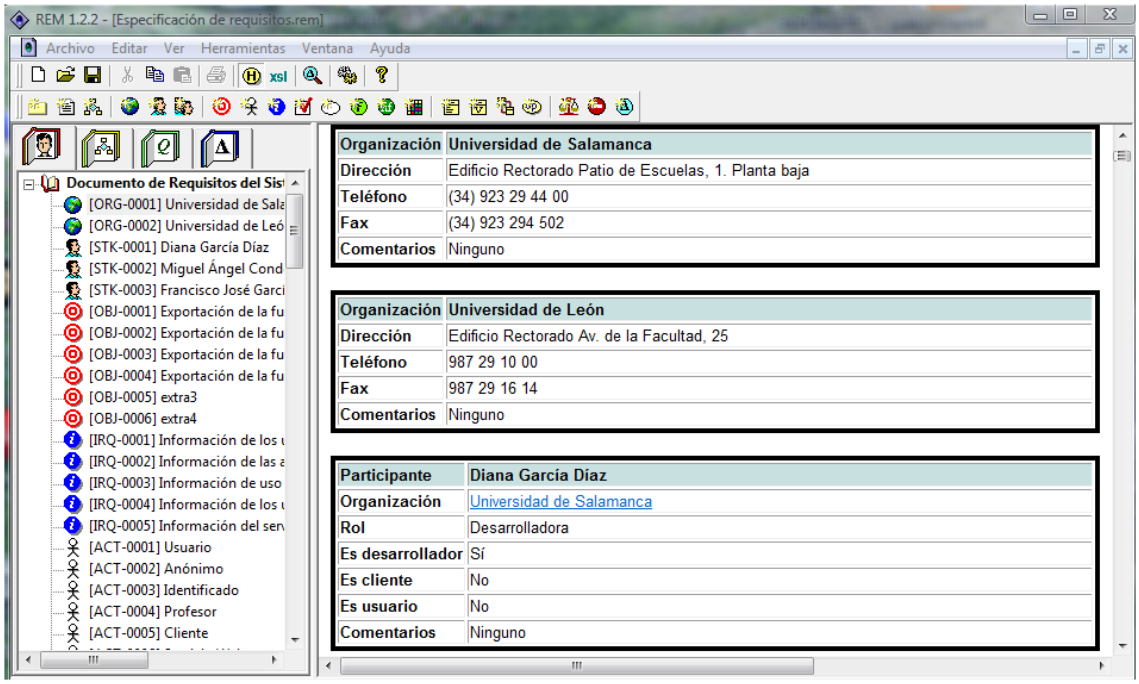


Figura 4: Interfaz REM

4.5.2. Android

Android (*Android, 2015a*) es una plataforma de software y un sistema operativo para dispositivos móviles basado en el núcleo de Linux. Esta plataforma ofrece: sistema operativo, middleware (capa de abstracción software situada entre las aplicaciones y el sistema operativo junto con la red) y herramientas para el desarrollo de aplicaciones móviles.

Android ha sido creada por una serie de empresas unidas en la denominada Open Handset Alliance (*Open Handset Alliance, 2015*). En ella se agrupan 47 empresas de diferentes ámbitos como pueden ser empresas de telefónica, de creación de hardware, de fabricación de hardware y de teléfonos. La empresa más importante y más involucrada en el desarrollo de este sistema operativo es Google (*Google, 2015*).

La plataforma ha sido creada totalmente abierta. El sistema está diseñado para permitir a los desarrolladores crear aplicaciones y explotar todas las ventajas que un teléfono puede ofrecer creando aplicaciones que pueden desde realizar llamadas, mandar mensajes o usar la cámara. Al ser creado como código abierto puede extenderse para incorporar nuevas tecnologías emergentes.

En Android todas las aplicaciones son ejecutadas con los mismos privilegios, no existe diferencia entre las aplicaciones del núcleo y aquellas creadas por terceras personas. Una aplicación puede utilizar todas las características proporcionadas por el teléfono ofreciendo a los usuarios gran variedad de servicios. Se puede desde cambiar la pantalla de inicio, el programa de marcación telefónica o la aplicación por defecto que se utilizará para la visualización de fotografías.

Con Android se intentan romper las barreras de desarrollo e innovación en aplicaciones para móviles. Un programador puede combinar información obtenida desde una página web con la información en su propio teléfono para mejorar la experiencia de usuario.

Se facilita el uso de un gran número de bibliotecas y herramientas que pueden ser utilizadas para desarrollar software. Gracias a las capacidades hardware, con una aplicación se puede obtener la localización del dispositivo y comunicarse con otro mediante aplicaciones *peer-2-peer* (red informática entre iguales, sin clientes-servidores). Además, se incluye un gran número de herramientas que han sido construidas junto con el desarrollo de la plataforma para permitir a los desarrolladores una alta productividad y una profunda visión de sus aplicaciones.

El sistema operativo está construido sobre un núcleo abierto de Linux. Se ha desarrollado una máquina virtual diseñada especialmente para la optimización de memoria y el uso de recursos hardware en entornos móviles. Actualmente Android permite desarrollar con lenguaje Java e incluye bibliotecas desarrolladas o adaptadas por Google (Google APIs).

Permisos en Android

Para proteger ciertos recursos y características especiales del hardware, Android define un esquema de permisos. Toda aplicación que acceda a estos recursos está obligada a declarar su intención de usarlos.

Cuando un usuario instala una aplicación podrá examinar la lista de permisos que solicita la aplicación y decidir si considera oportuno instalar dicha aplicación.

A continuación se muestra una lista con los permisos que se han utilizado en este proyecto para el PLE móvil:

- INTERNET**:** Permite establecer conexiones a través de internet. Este es posiblemente el permiso más importante, en el que más hay que fijarse a quién se le otorga. Muchas aplicaciones lo piden, pero no todas los necesitan.
- ACCESS_NETWORK_STATE**:** Información sobre la red: ver estado de conexión, ver estado de red.
- READ_PHONE_STATE: Leer estado del teléfono e identidad. Muchas aplicaciones piden este permiso para ponerse en pausa mientras hablas por teléfono. En las primeras versiones del SDK este permiso no era necesario.

Versiones

Android es un sistema operativo que fue anunciado el 5 de noviembre de 2007 por la recién creada Open Handset Alliance. De cara al público Android comenzó con la versión 1.5 del sistema operativo en abril de 2009.

Cada versión de Android lleva asociado lo que se denomina un nivel de API, o API level, de forma que se suelen referenciar estas versiones por su API level. Además de esto, tienen un nombre comercial. El nivel de API corresponde a números enteros comenzando desde 1. Para los nombres comerciales se han elegido postres en orden alfabético Cupcake (v1.5), Donut (v1.6), etc.

En la Figura 5 se muestran las distintas versiones de *Android* con sus nombres correspondientes (*Android, 2015b*):

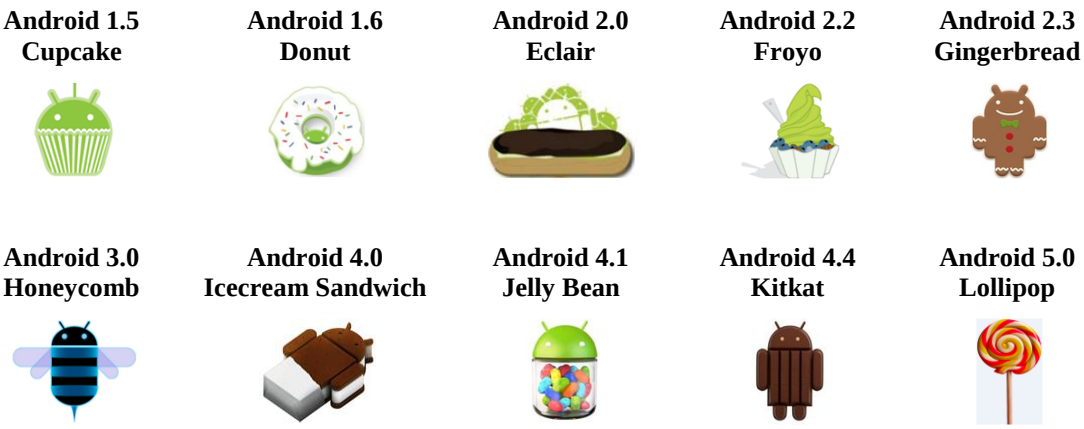


Figura 5. Versiones de Android

Antes de empezar a desarrollar un proyecto en Android hay que elegir la versión del sistema para la que deseamos realizar la aplicación. La elección de la mínima versión de la aplicación para este proyecto se ha hecho tomando como referencia el uso que se hace actualmente (enero de 2015) de las distintas versiones del sistema operativo

por parte de sus usuarios. Esta información, que se refleja en la Figura 6, es publicada por Google (*Android, 2015c*). Por este motivo, para que la aplicación pueda ser utilizada por la totalidad de los usuarios actuales de dispositivos móviles con sistema operativo Android, la versión mínima de esta es Android 2.2, Froyo.

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.8%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.7%
4.1.x	Jelly Bean	16	19.2%
4.2.x		17	20.3%
4.3		18	6.5%
4.4	KitKat	19	39.1%

Data collected during a 7-day period ending on January 5, 2015.
Any versions with less than 0.1% distribution are not shown.

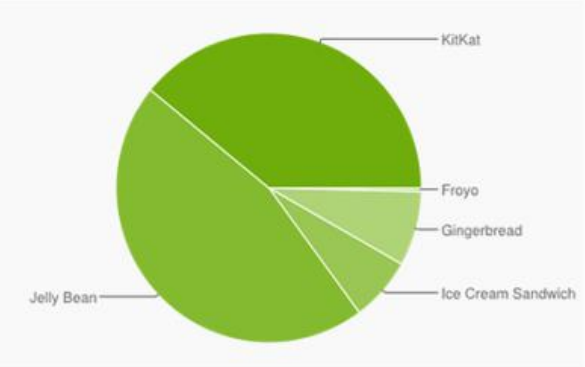


Figura 6. Datos de uso de las versiones Android

4.5.3. SDK – Software Development Kit

Para poder crear aplicaciones para la plataforma Android, se encuentra a disposición de los programadores un conjunto de herramientas que permiten el desarrollo de software: Software Development Kit (SDK) (*Android SDK, 2015*). El SDK se encuentra disponible para Windows, Linux o Mac OS X. Las principales herramientas son el emulador y el plugin para el entorno de desarrollo Eclipse. El desarrollo de software mediante el emulador garantiza el funcionamiento de aplicaciones en los dispositivos debido a que se usa la misma imagen del sistema. Aún así, se recomienda siempre que se quiera desarrollar para dispositivos móviles que se pruebe la aplicación en terminales reales antes de su distribución. Esta herramienta se puede descargar en la siguiente dirección: http://developer.android.com/sdk/installing/installing-adt.html/.

Además, el SDK también proporciona una serie de útiles para depurar, empaquetar e instalar aplicaciones en el emulador o en el dispositivo.

4.5.4. XAMPP

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de **X** (para cualquiera de los diferentes sistemas operativos), **A**pache, **M**ySQL, **P**hp, **P**erl. XAMPP se actualiza regularmente para incorporar las últimas versiones de Apache/MySQL/PHP y Perl. También incluye módulos como OpenSSL y phpMyAdmin.

El programa está liberado bajo la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris y Mac OS X.

Oficialmente, los diseñadores de XAMPP fueron los de Baiker y Anthony Corporation, los cuales solo pretendían su uso como una herramienta de desarrollo, para permitir a los diseñadores de sitios webs y programadores testear su trabajo en sus propios ordenadores sin ningún acceso a Internet. En la práctica, sin embargo, XAMPP es utilizado actualmente como servidor de sitios Web, ya que, con algunas modificaciones, es lo suficientemente seguro para serlo. Con el paquete se incluye una herramienta especial para proteger fácilmente las partes más importantes.

XAMPP es la tecnología que se ha elegido como servidor para las aplicaciones escritas en PHP, como es el caso de Moodle, ya que XAMPP es un paquete fácil de instalar y configurar.

La razón de por qué se ha utilizado esta tecnología en lugar de otras como pueden ser EasyPHP ó Wampserver es porque se produjeron diversos problemas a la hora de instalar *Moodle* y finalmente, después de numerosos intentos fallidos, se consiguió instalar *Moodle* en XAMPP.

La versión utilizada en la realización de este proyecto ha sido la 1.8.3. Esta versión incluye PHP 5.5.9, Apache 2.4.7, MySQL 5.5.32 y phpMyAdmin 4.1.6, entre otros. En la Figura 7 puede verse una captura de su interfaz principal:

Figura 7. Interfaz XAMPP

Apache

Servidor Web cuya misión es la de recibir y responder peticiones de páginas Webs. Este servidor se desarrolla dentro del proyecto *HTTP Server* de la fundación no lucrativa *Apache Software Foundation*.

Se trata de un software libre de código abierto multiplataforma y extensible mediante módulos, con una amplia aceptación en la red, que implementa el protocolo HTTP/1.1. Además, presenta entre otras características altamente configurables, bases

de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

MySQL

Sistema gestor de bases de datos (SGBD) fundamental para el sistema en desarrollo.

Las características principales de MySQL a destacar son las siguientes:

- MySQL es un sistema multiplataforma, manteniéndose así la independencia de la aplicación respecto a un sistema operativo utilizado.
- Se trata de un software gratuito.
- Permite implementar gran parte del estándar SQL sin las particularidades de un SGBD en concreto.

PhpMyAdmin

Sistema mediante el cual, podemos desde un navegador mantener una base de datos MySQL de manera fácil e intuitiva.

PhpMyAdmin es una herramienta escrita en PHP con la intención e manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente, puede crear y eliminar bases de datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos.

Este proyecto se encuentra vigente desde el año 1998 bajo la licencia GPL. Como esta herramienta corre en máquinas con servidores webs y soporte de PHP y MySQL, la tecnología utilizada ha ido variando durante su desarrollo.

En la Figura 8 se puede ver una captura de su interfaz:

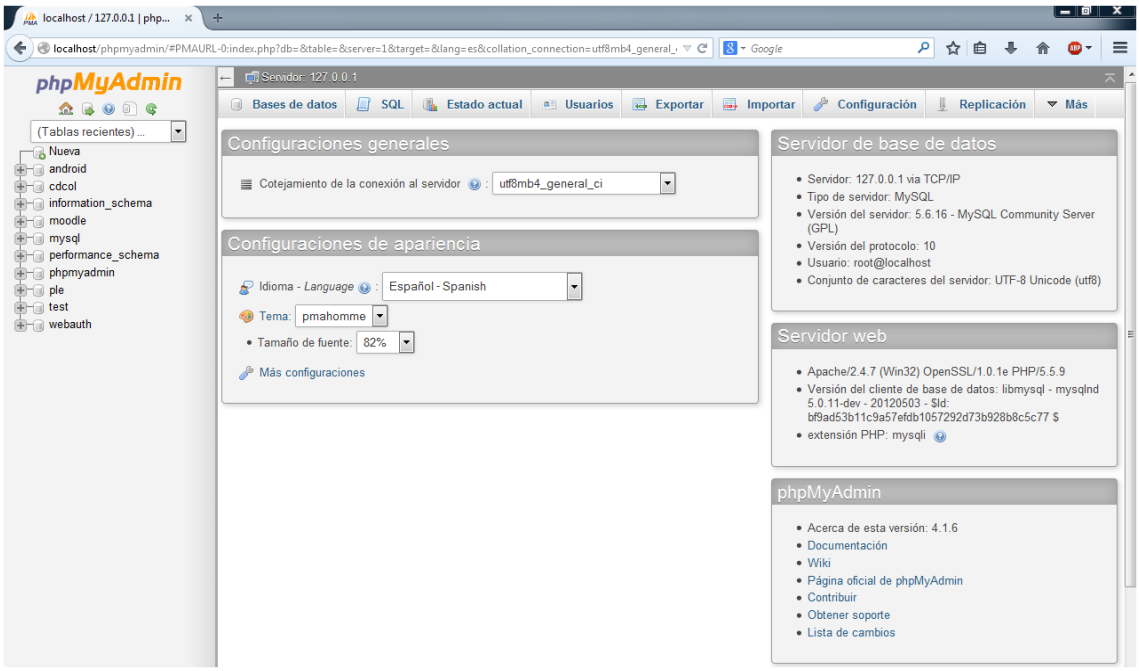


Figura 8. Interfaz PHPMyAdmin

4.5.5. Visual Paradigm

Visual Paradigm para UML es una herramienta para el desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de Software, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos.

Esta herramienta está pensada para el desarrollo orientado a objetos y cuenta con una total integración con UML, dos características que pretenden facilitar al usuario un modelado visual y una construcción basada en componentes de aplicaciones de software. Las características principales de esta herramienta son:

- Notación UML como lenguaje de modelado.
- Soporte para todos los diagramas UML permitiendo mantener una relación entre unos tipos de diagramas y otros.
- Permite generación de código y proporciona una herramienta para realizar el proceso inverso, denominado ingeniería inversa.
- Permite la importación y exportación de los diagramas realizados a diferentes formatos.
- Herramientas para la generación de documentación del proyecto realizado.

En este proyecto, se ha optado por utilizar Visual Paradigm for UML, ya que es una herramienta que, junto con REM, se ha utilizado en todas las prácticas con partes de ingeniería de software que se han tenido que realizar durante la carrera.

Concretamente, se ha utilizado Visual Paradigm for UML Enterprise Edition (VP-UML EE), que es la edición top de la línea productos. Agrega valor en términos de modelado de datos orientado a objetos, hace posible la documentación del proyecto, mapeo relacional de objetos para Java, .NET y PHP, reduciendo costos y aumentando su productividad. En la Figura 9 se muestra un ejemplo de su interfaz.

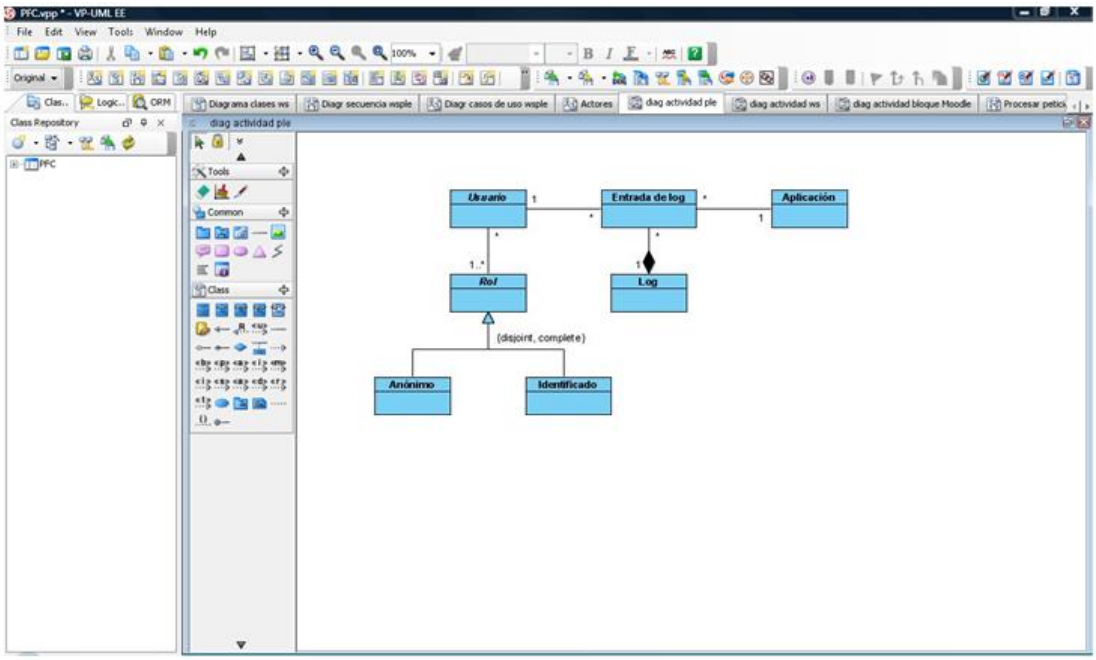


Figura 9. Interfaz Visual Paradigm

4.5.6. StarUML

StarUML es una herramienta que se utiliza para el modelado de *software* que está basada en estándares UML y MDA. En un principio era un producto comercial y pasó a convertirse en uno de licencia abierta GNU/GPL.

Esta herramienta se ha utilizado en el proyecto, en su versión 5.0.2.1570, para la creación de los diagramas de navegación, aunque no soporta la creación de estos diagramas. En la Figura 10 se muestra una captura de su interfaz.

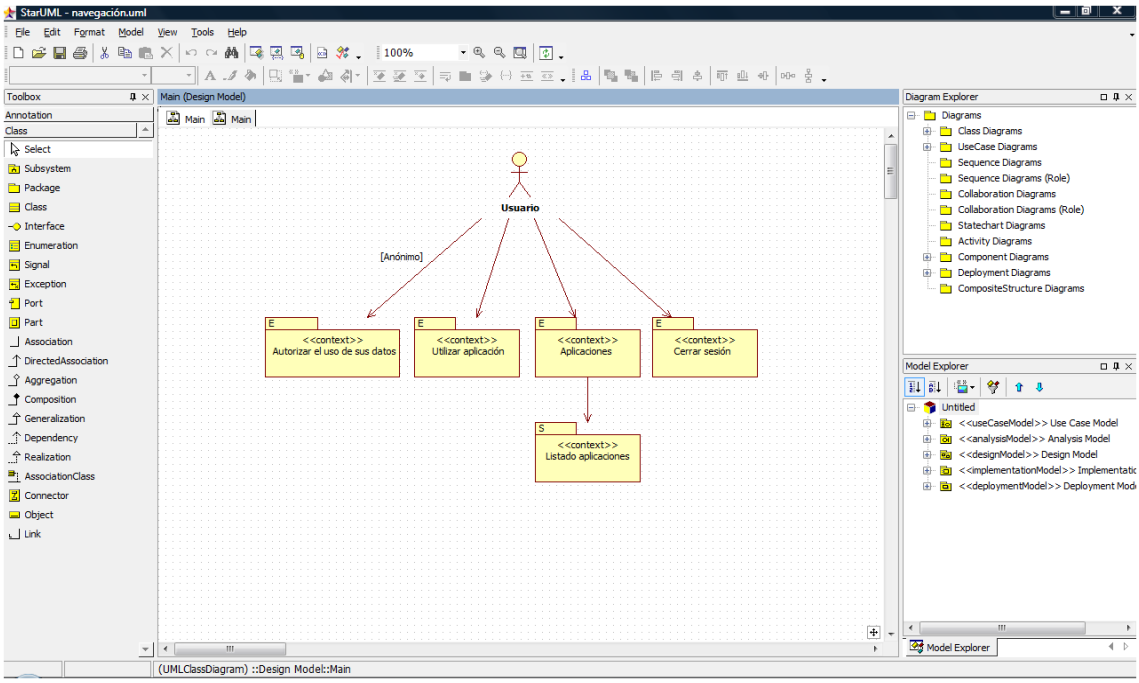


Figura 10: Interfaz StarUML

4.5.7. Eclipse

Eclipse es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado *Java Development Toolkit* (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse).

En este proyecto se ha utilizado el IDE Eclipse para la creación del PLE móvil y el sistema de log. Concretamente, se ha utilizado una versión de Eclipse para Java, para lo que ha sido necesaria la instalación de un *plugin* denominado ADT, que permite integrar Eclipse con el SDK de Android.

Los pasos de la preparación del entorno de desarrollo para la programación de aplicaciones *Android* se pueden consultar en la documentación de *Android (Android, 2015).*

En la Figura 11 se puede ver una captura de Eclipse con el *plugin* ADT instalado.

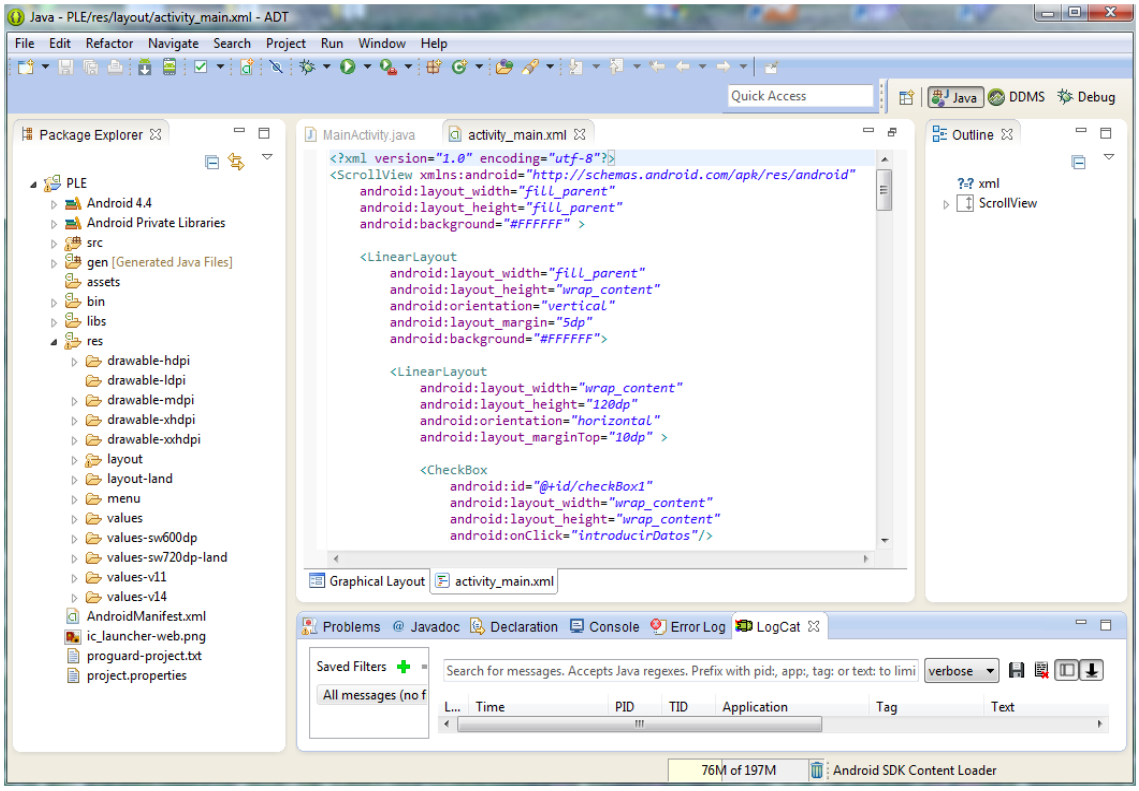


Figura 11. Interfaz Eclipse con *plugin* ADT

4.5.8. Google API Chart

Google API Chart (*Google Charts, 2015*) es una herramienta de Google que permite crear distintos tipos de gráficos estadísticos de una manera rápida y sencilla. Estos gráficos se pueden crear accediendo a una URL con los parámetros adecuados de acceso a la API, el tamaño del gráfico, los datos que se quieran representar en el gráfico, el tipo de gráfico y otros elementos adicionales/opcionales que se quieran añadir al gráfico, como por ejemplo, título, estilos, leyendas, etc.

Esta herramienta ofrece una gran variedad de gráficos, como por ejemplo: gráfico de líneas, palabra gráfica, gráfico de barras, gráfico circular, gráfico de puntos, mapa, etc. En este proyecto se ha optado por utilizar el gráfico de barras para representar los datos en la herramienta que se ha creado para el consumo y explotación de la información en Moodle.

API Chart permite la creación de gráficas a partir de imágenes, con lo que, para mostrar un gráfica, lo único que hay que hacer es utilizar la etiqueta IMG de HTML. Para configurar la forma y datos del gráfico se debe definir simplemente la URL de la imagen de la imagen en el atributo SRC de la etiqueta IMG. Así pues, para crear una gráfica, no hay que instalar ningún componente en el servidor, simplemente colocar una imagen en la página, con un código como este:

```
<img src=”http://chart.apis.google.com/chart?parametro1=valor1&...&parametroN=valorN>”
```

Como puede verse, a la url se le pasan una serie de parámetros, de los cuales algunos son obligatorios y otros opcionales. A continuación, se explican brevemente los parámetros más importantes que se han utilizado para la representación de las gráficas en el bloque de *Moodle*:

- **chs**: Parámetro obligatorio que sirve para indicar el tamaño que tendrá la gráfica.
- **chd:** Parámetro obligatorio que sirve para pasarle los datos que se quieran representar en el gráfico (en este caso, los porcentajes de uso de aplicaciones en el PLE móvil).
- **cht:** Parámetro obligatorio para indicar el tipo de gráfico (en este proyecto: cht=p que se corresponde con la gráfica circular o de tarta).
- **chco:** Parámetro opcional para indicar los colores que se quieren aplicar al gráfico.
- **chtt:** Parámetro opcional que permite indicar el título del gráfico.

4.5.9. TCPDF

Para exportar a formato PDF los distintos informes generados sobre la actividad llevada a cabo en el PLE móvil desde el bloque de Moodle, se ha utilizado la biblioteca TCPDF. La razón de su elección se debe a que esta biblioteca viene incluida en Moodle.

TCPDF *(TCPDF, 2015)* es una clase de software libre PHP para generar documentos PDF iniciada en 2002. Actualmente, es uno de los proyectos de código abierto más activos a nivel mundial. En su página oficial (http://www.tcpdf.com/examples/) se pueden encontrar una gran cantidad de ejemplos de uso de esta biblioteca.

4.5.10. Microsoft Office Word

Para la escritura de la documentación del presente proyecto se ha utilizado *Microsoft Office Word*, en su versión de 2007.

Microsoft Word es un software destinado al procesamiento de textos que fue creado por la empresa Microsoft, y actualmente viene integrado en la *suite* ofimática Microsoft Office.

5. Descripción del proyecto

En este apartado se va a describir brevemente el producto final una vez concluido su desarrollo.

5.1. Aplicación Android: PLE

Se ha creado una aplicación *Android* denominada PLE que permite reunir un total de ocho aplicaciones en una sola interfaz. Dichas aplicaciones serán elegidas por el usuario de entre todas las que tenga instaladas en el dispositivo móvil. En la Figura 12 se muestra un ejemplo de su interfaz principal:

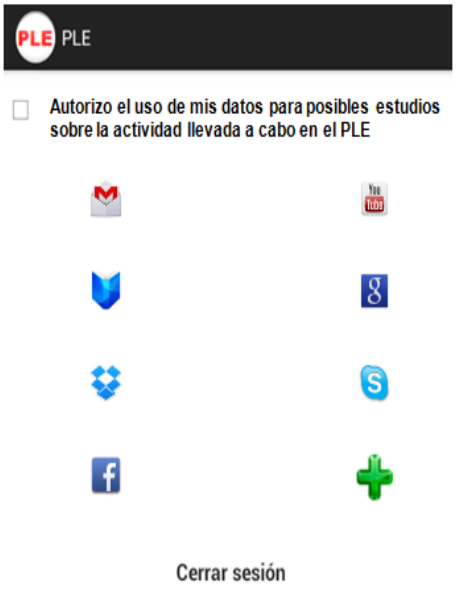


Figura 12: Interfaz PLE móvil

Como se puede observar en la Figura 12, además de las aplicaciones incluidas en el PLE, la interfaz consta de dos funcionalidades u opciones más:

- *"Autorizo el uso de mis datos para posibles estudios sobre la actividad llevada a cabo en el PLE"*: Los usuarios en un dispositivo móvil pueden utilizar las distintas aplicaciones incluidas en el PLE de manera anónima o identificada. Si quieren utilizarlo de forma anónima no pulsarán el CheckBox, mientras que, si por el contrario, quieren hacerlo de forma identificada, deberán pulsarlo y se mostrará una pantalla en la que se les pedirá que introduzcan su nombre y su DNI. Esto puede verse en la Figura 13, en la que se muestra un ejemplo de otra de las interfaces que ofrece la aplicación:

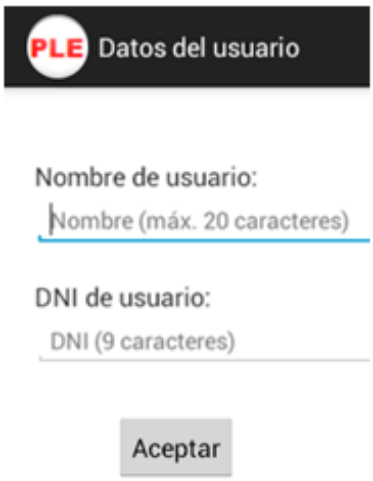


Figura 13: Pantalla para introducir los datos del usuario

- *"Cerrar Sesión"*: Esta opción permite que un mismo PLE pueda ser utilizado por distintos usuarios, o por el mismo pero utilizándolo de manera distinta (anónima o autenticada), en un mismo dispositivo móvil e inicializa el PLE.

Para que el usuario sepa qué aplicaciones están disponibles para poder incluirse en su PLE móvil, se muestra un listado de las mismas en el que se muestra información sobre ellas. En concreto: icono, nombre y estado (si está incluida o no en el PLE). Dependiendo del estado en el que se encuentre cada una de las aplicaciones, el usuario podrá incluirla en el PLE móvil, si la aplicación no está ya incluida, o quitarla, en caso contrario.

En la Figura 14 se puede ver una captura de ejemplo de la interfaz que muestra el listado de aplicaciones instaladas en un dispositivo de prueba y que se pueden incluir en el PLE móvil:

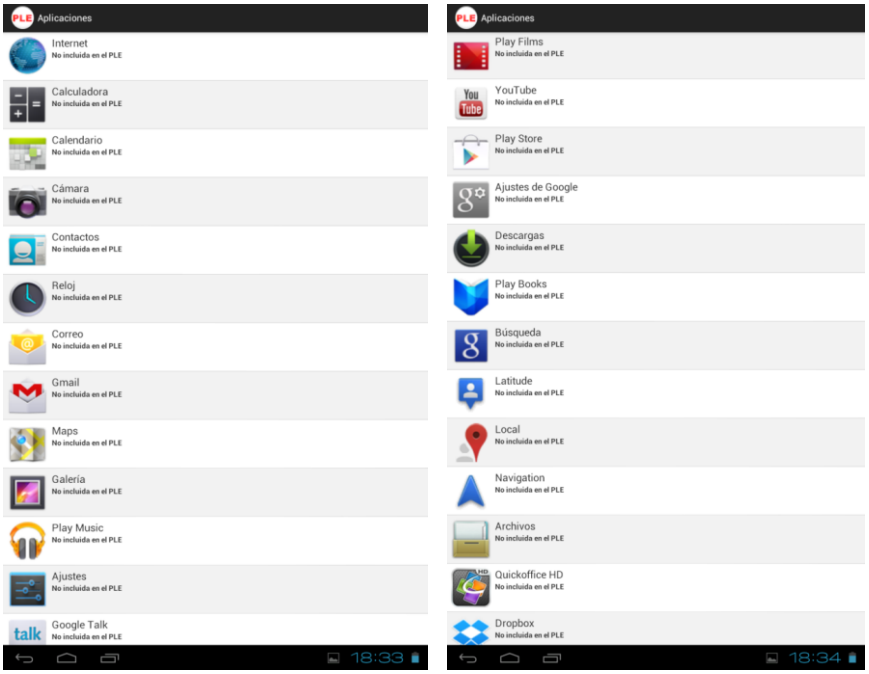


Figura 14: Listado de aplicaciones instaladas en el dispositivo

5.2. Fichero de log

Este fichero de log se ha creado para gestionar y almacenar la actividad llevada a cabo en la aplicación PLE.

En este fichero de log se guardan los datos del usuario (tanto si es anónimo como autenticado), información relativa a las aplicaciones incluidas en el PLE e información de la actividad del usuario en ellas. Concretamente, se guardan los siguientes datos: nombre de usuario, DNI de usuario, nombre de aplicación utilizada, fecha y hora de uso de la aplicación.

Para ello, se ha hecho uso de una de las alternativas que nos ofrece *Android* para el almacenamiento de información en ficheros: el almacenamiento en la memoria interna del dispositivo. Por lo tanto, el almacenamiento de la información en el fichero de log solo es accesible por la aplicación (ni siquiera el usuario del dispositivo tendrá acceso) y será eliminado cuando se desinstale la aplicación.

5.3. Servicio web de interacción con Moodle

La funcionalidad del servicio web creado en el proyecto para la interacción con *Moodle* es insertar la información sobre la actividad llevada a cabo en el PLE móvil en una tabla en la base de datos de *Moodle*. Dicho servicio recibirá la información, la insertará en la tabla de datos y devolverá si la inserción ha tenido éxito o no.

5.4. Herramienta para el consumo de la información desde Moodle

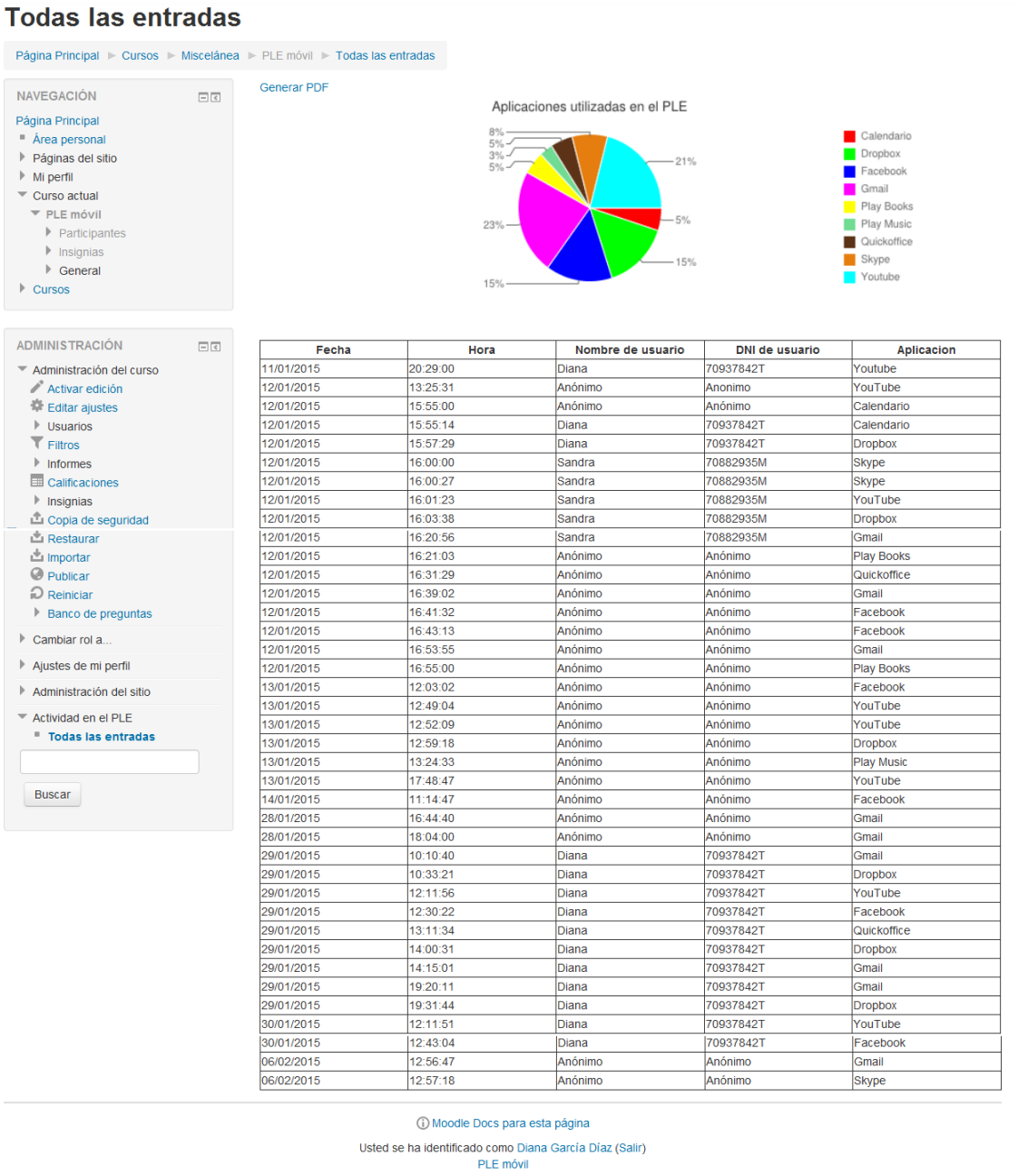
La herramienta para el consumo de la información desde *Moodle* consiste en la creación de un bloque en *Moodle* que permite al profesor el consumo de la información sobre la actividad llevada a cabo en el PLE móvil. Este bloque consta de tres opciones o funcionalidades. En la Figura 15 se puede ver una captura del menú del bloque en la que se pueden ver las distintas opciones que tiene:

Figura 15. Captura del bloque en *Moodle*

Como puede observarse en la Figura 15, las opciones son: “Todas las entradas”, “Registros de hoy” y “Gestionar informe”. El profesor podrá obtener distintos informes dependiendo de la opción que elija. A continuación, se explica el funcionamiento de cada una de ellas.

- **Todas las entradas:** Muestra un gráfico circular en el que se representa el uso que se ha hecho de las aplicaciones en el PLE móvil y una tabla con todos los registros almacenados en la base de datos sobre la actividad llevada a cabo en el PLE móvil. Por cada registro se especifica la fecha, hora, nombre de usuario,

DNI de usuario y nombre de la aplicación utilizada. En la Figura 16 se puede ver una captura de este tipo de informe:



Fecha	Hora	Nombre de usuario	DNI de usuario	Aplicacion
11/01/2015	20:29:00	Diana	70937842T	Youtube
12/01/2015	13:25:31	Anónimo	Anonimo	YouTube
12/01/2015	15:55:00	Anónimo	Anónimo	Calendario
12/01/2015	15:55:14	Diana	70937842T	Calendario
12/01/2015	15:57:29	Diana	70937842T	Dropbox
12/01/2015	16:00:00	Sandra	70882935M	Skype
12/01/2015	16:00:27	Sandra	70882935M	Skype
12/01/2015	16:01:23	Sandra	70882935M	YouTube
12/01/2015	16:03:38	Sandra	70882935M	Dropbox
12/01/2015	16:20:56	Sandra	70882935M	Gmail
12/01/2015	16:21:03	Anónimo	Anónimo	Play Books
12/01/2015	16:31:29	Anónimo	Anónimo	Quickoffice
12/01/2015	16:39:02	Anónimo	Anónimo	Gmail
12/01/2015	16:41:32	Anónimo	Anónimo	Facebook
12/01/2015	16:43:13	Anónimo	Anónimo	Facebook
12/01/2015	16:53:55	Anónimo	Anónimo	Gmail
12/01/2015	16:55:00	Anónimo	Anónimo	Play Books
13/01/2015	12:03:02	Anónimo	Anónimo	Facebook
13/01/2015	12:49:04	Anónimo	Anónimo	YouTube
13/01/2015	12:52:09	Anónimo	Anónimo	YouTube
13/01/2015	12:59:18	Anónimo	Anónimo	Dropbox
13/01/2015	13:24:33	Anónimo	Anónimo	Play Music
13/01/2015	17:48:47	Anónimo	Anónimo	YouTube
14/01/2015	11:14:47	Anónimo	Anónimo	Facebook
28/01/2015	16:44:40	Anónimo	Anónimo	Gmail
28/01/2015	18:04:00	Anónimo	Anónimo	Gmail
29/01/2015	10:10:40	Diana	70937842T	Gmail
29/01/2015	10:33:21	Diana	70937842T	Dropbox
29/01/2015	12:11:56	Diana	70937842T	YouTube
29/01/2015	12:30:22	Diana	70937842T	Facebook
29/01/2015	13:11:34	Diana	70937842T	Quickoffice
29/01/2015	14:00:31	Diana	70937842T	Dropbox
29/01/2015	14:15:01	Diana	70937842T	Gmail
29/01/2015	19:20:11	Diana	70937842T	Gmail
29/01/2015	19:31:44	Diana	70937842T	Dropbox
30/01/2015	12:11:51	Diana	70937842T	YouTube
30/01/2015	12:43:04	Diana	70937842T	Facebook
06/02/2015	12:56:47	Anónimo	Anónimo	Gmail
06/02/2015	12:57:18	Anónimo	Anónimo	Skype

Figura 16. "Todas las entradas"

- **Registros de hoy:** Muestra un gráfico circular en el que se representa el uso que se ha hecho de las aplicaciones en el PLE móvil en el día de hoy y una tabla con los registros almacenados en la base de datos de *Moodle* sobre la actividad llevada a cabo en el PLE móvil en el día de hoy (día en que se utilice esta opción del bloque). En la Figura 17 se puede ver una captura de este informe con fecha 06/02/2015, que es el día en que se ha hecho esta captura:

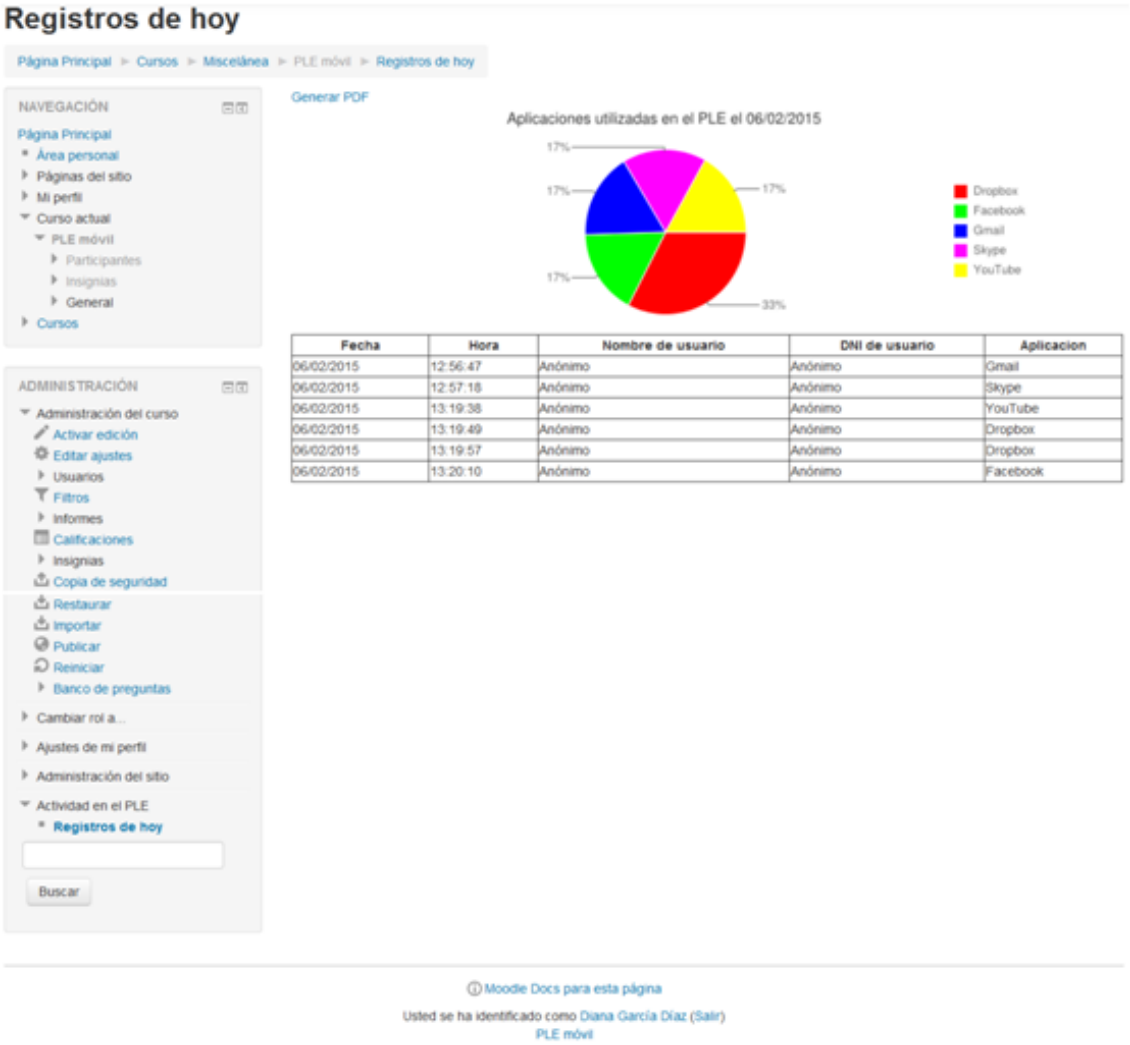


Figura 17. "Registros de hoy"

- **Gestionar informe:** Muestra un formulario en el que el profesor, dependiendo de la información que quiera visualizar en cada momento, podrá poner las condiciones que quiera y elegir el orden en el que quiere que aparezcan los datos en la tabla. En la Figura 18 se puede ver una captura dicho formulario:

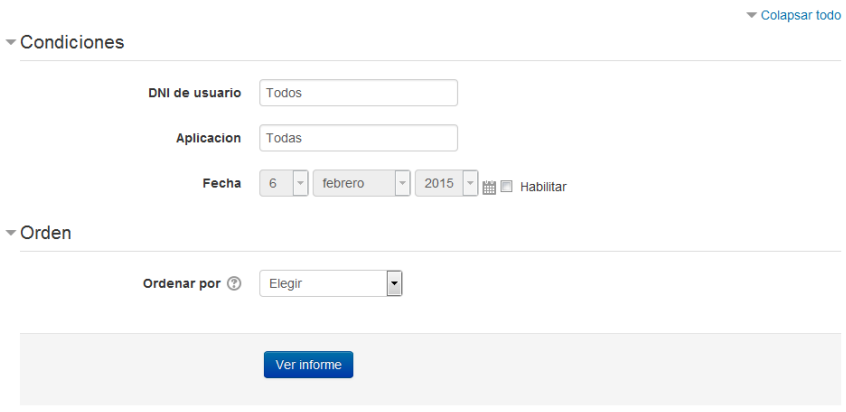


Figura 18. Formulario para gestionar un informe

Las condiciones que puede poner el profesor son opcionales, es decir, no se requiere rellenarlas todas, el profesor podrá poner una, dos, tres o ninguna condición. Estas condiciones son:

- **DNI de usuario**: El profesor introduce el DNI de usuario del que quiere hacer un seguimiento o estudio. Si no quiere hacer el informe sobre un usuario concreto, el valor de este campo por defecto será "Todos".
- **Aplicación**: El profesor podrá seleccionar una de las aplicaciones incluidas en el PLE móvil o todas, dependiendo de lo que le interese en ese momento. Por defecto, el valor de este campo será "Todas".
- **Fecha**: El profesor podrá seleccionar la fecha sobre la que quiere realizar la consulta. Para ello, tendrá que habilitar el selector de fecha que ofrece esta condición, y seleccionar la fecha que desee. Por defecto, el selector de fecha estará deshabilitado y no tiene ningún valor para este campo.

El orden permite al profesor elegir la ordenación de los datos en el informe:

- **Ordenar por**: El profesor podrá seleccionar el orden de aparición de los datos en la tabla. Los campos que puede elegir son ordenar por: Nombre de usuario, DNI de usuario, Aplicación o Fecha. Por defecto, los datos aparecerán según se han ido insertando en la tabla de base de datos.

Una vez que el profesor ha rellenado el formulario para gestionar el informe, podrá visualizarlo pulsando el botón "**Ver informe**" y se mostrará un gráfico circular en el que se representa la información de acuerdo con las condiciones que haya puesto el profesor y una tabla que mostrará los registros que cumplan las condiciones y en el orden elegido por el profesor en el formulario. En la Figura 19 se puede ver una captura del formulario para gestionar un informe con las siguientes condiciones/filtros puestos:

- DNI de usuario: 70882935M
- Aplicación: Todas
- Fecha: 12/01/2015
- Ordenar por: Aplicación

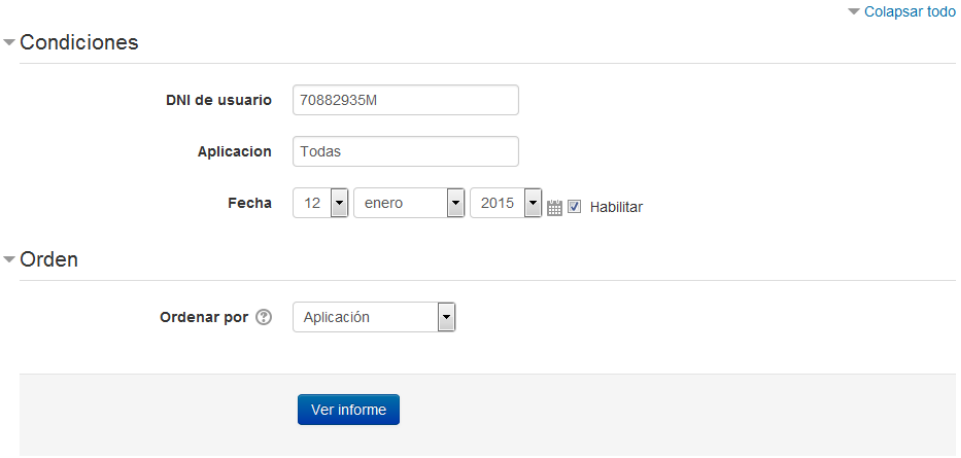


Figura 19. Formulario rellenado

Y en la Figura 20, se puede ver el informe correspondiente que se genera a partir del formulario de la Figura 19:

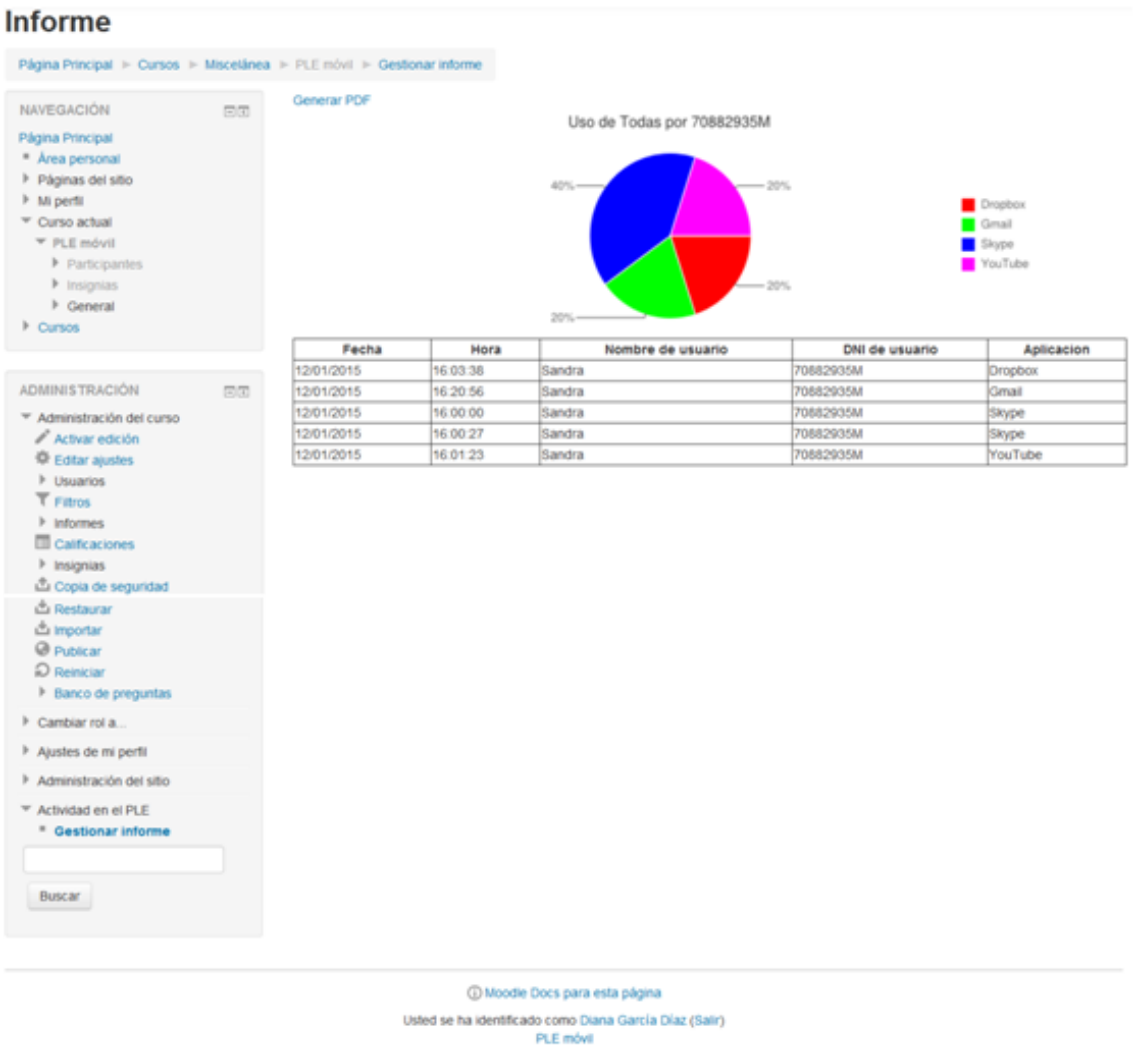


Figura 20. Informe gestionado

5.5. Herramienta para la explotación de la información y toma de decisiones

La herramienta que facilita la explotación de la información desde el entorno institucional y la facilita la toma de decisiones se corresponde con la opción "Generar PDF" que ofrecen los distintos tipos de informes al ser visualizados en la parte superior derecha de la página, como puede verse en las Figuras 16, 17 y 20. Esto permite al profesor exportar los distintos tipos de informes a formato PDF. En la Figura 21, se muestra el ejemplo del PDF generado desde la página de la Figura 20:

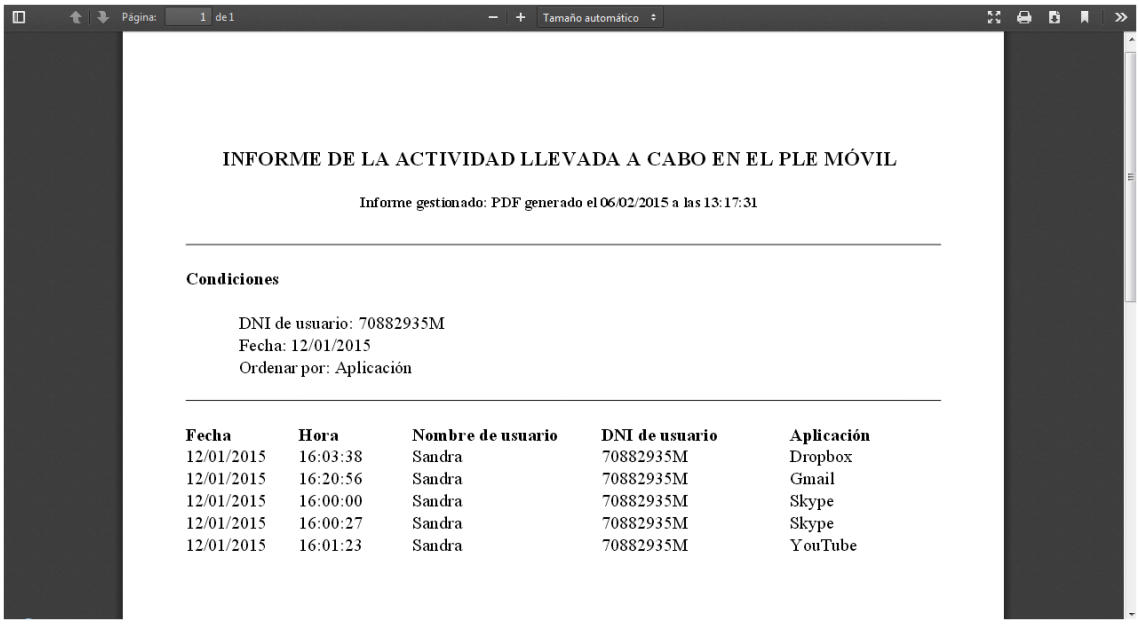


Figura 21. PDF Informe gestionado

6. Aspectos relevantes

En este apartado se recogen los aspectos más interesantes del desarrollo de las diferentes partes del proyecto. De esta manera, se citará la metodología utilizada y las fases de desarrollo por las que se han ido pasando hasta la conclusión del mismo.

Como se ha observado, este proyecto pretende implementar un sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje (PLE) móvil que será integrado en un entorno institucional como el LMS (*Moodle*). Esta tarea ha implicado la creación de: la aplicación *Android*, el servicio web de interacción con el entorno institucional y el bloque de *Moodle* que permite consumir y explotar la información desde el entorno institucional.

La metodología de desarrollo utilizada en todas las partes del proyecto ha sido el Proceso Unificado, explicado anteriormente en el apartado 4.1 de esta memoria. Por lo tanto, todas las partes del proyecto se explicarán de manera conjunta, comenzando por mostrar lo más significativo del análisis y finalizando con los aspectos más interesantes de la implementación y pruebas, junto con una pequeña muestra de su funcionalidad.

Las fases del Proceso Unificado se describen en los siguientes puntos.

6.1.- Análisis

Conocer qué tiene que hacer que hacer el *software* es el punto de partida, y la parte más importante del proceso de desarrollo. Así pues, la correcta obtención de los requisitos es uno de los aspectos más críticos de un proyecto *software*, dado que una mala captura de los mismos es la causa de la mayor parte de los problemas que surgen a lo largo del ciclo de vida.

La metodología empleada para detallar toda la información concretada y la escritura de los casos de uso es la Metodología para la Elicitación de Requisitos de Sistemas Software v2.3 propuesta por Durán y Bernárdez. Para apoyar esta metodología se ha utilizado la herramienta *REM*.

Toda la documentación correspondiente al análisis de las distintas partes del proyecto se recoge en el anexo 2 que se entrega en el CD adjunto con la memoria.

En primer lugar se procedió a la identificación de los principales actores que intervendrían en el proyecto. A continuación, se procedió a la identificación de los requisitos que debe cumplir el sistema: de información, funcionales, no funcionales y de restricción. En la Figura 22 se muestran los paquetes de los casos de uso del proyecto.

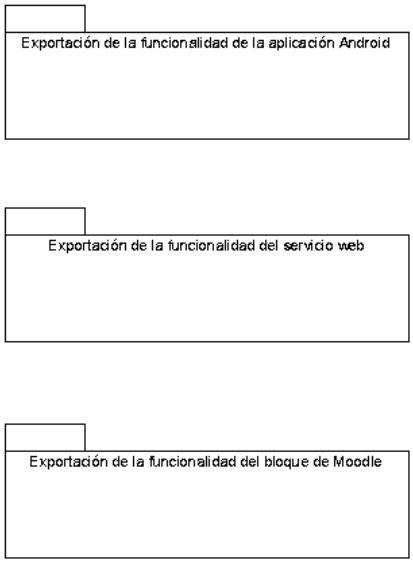


Figura 22. Paquetes de casos de uso

Una vez especificados los casos de uso, se procedió a realizar un refinamiento por la información y una consolidación por la funcionalidad, ambos correspondientes al modelo de dominio.

Para el desarrollo del modelo de dominio, se ha utilizado el lenguaje UML. Esto no ha supuesto un esfuerzo extraordinario de aprendizaje, puesto que ya se ha trabajado con esta herramienta en diferentes prácticas durante la carrera.

Para el refinamiento de la información, se realiza en primer lugar el modelo estático, modelando el sistema con clases del mundo real sin mostrar ningún objeto *software*. Se dice que es estático porque no representa la interacción en el tiempo de los objetos, sino que representa una visión "parada" de las clases y sus interacciones. Todo ello para obtener el diagrama de clases de dominio. En la Figura 23 se muestra uno de los diagramas realizados, el de la aplicación *Android*. El resto de los diagramas de clases se pueden ver en el anexo 2 que se entrega en el CD adjunto con la memoria.

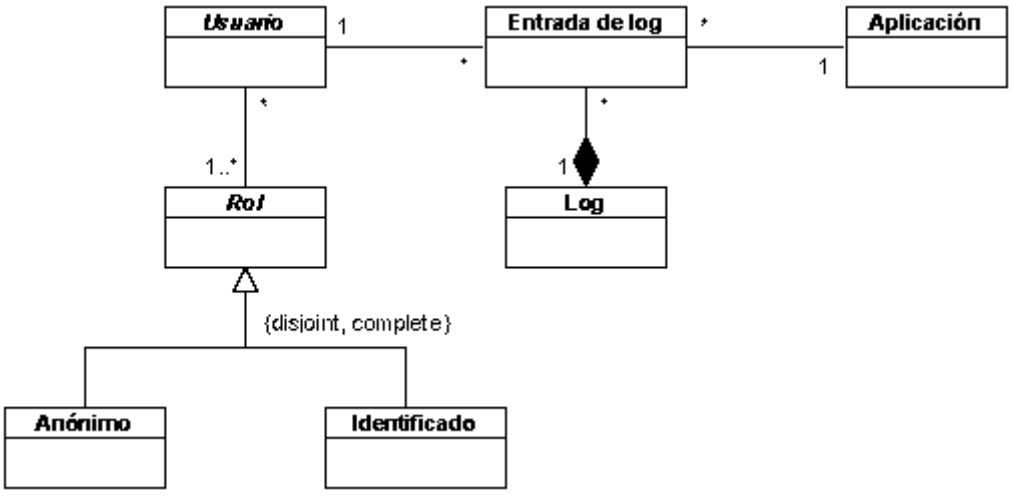


Figura 23. Diagrama de clases del dominio de la aplicación *Android*

Posteriormente, se realiza la vista de interacción, en la que a través de diagramas de secuencia se especifican los escenarios significativos del catálogo de requisitos. Se eligieron los diagramas de secuencia por ser los más intuitivos, ya que muestran como colaboran un conjunto de objetos dentro de un escenario y los mensajes que se intercambian entre ellos a lo largo del tiempo. En la Figura 24 puede verse un ejemplo en el que se representa el diagrama de secuencia del caso de uso Autorizar el uso de sus datos.

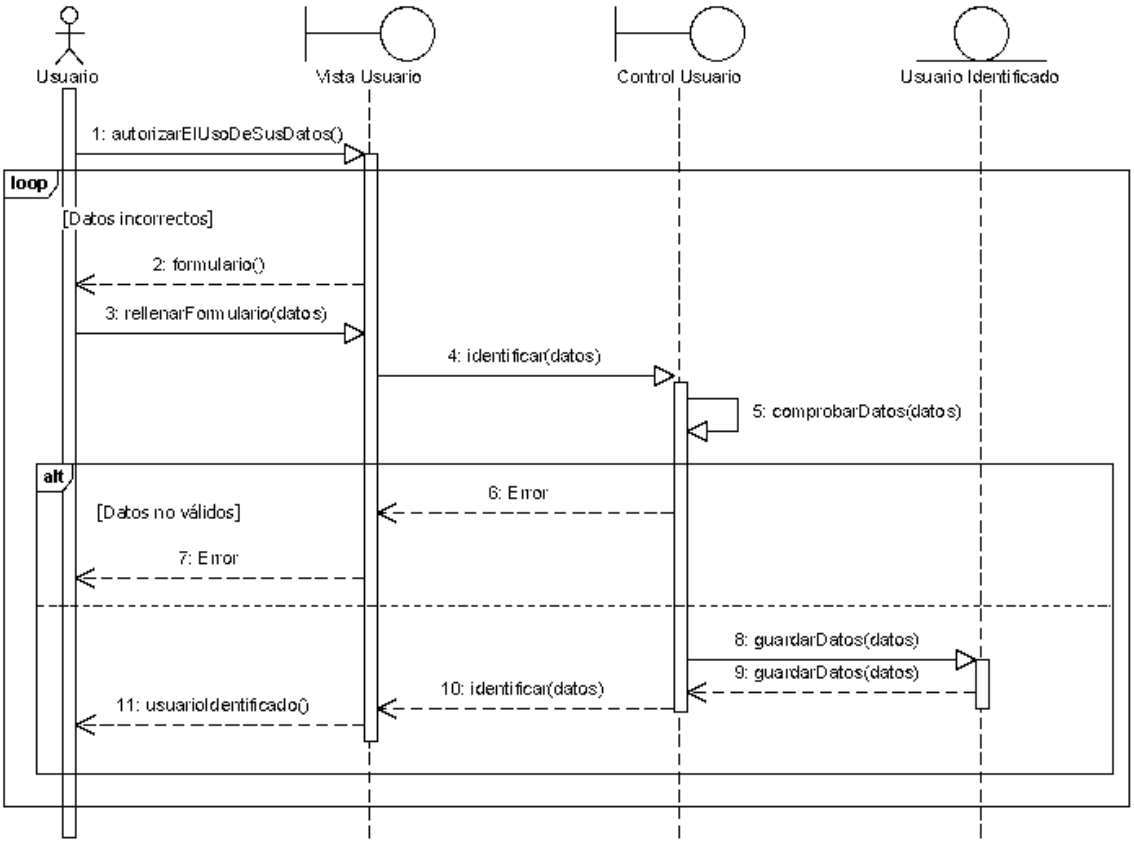


Figura 24. Diagrama de secuencia

También se ha modelado un diagrama de actividad para el servicio web, el cual representa de manera gráfica el flujo de interacción. En la Figura 25 se muestra dicho diagrama de actividad.

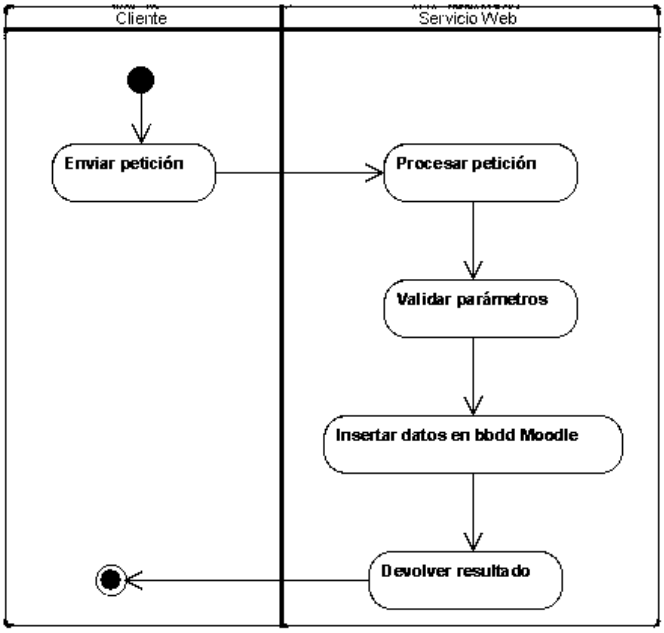


Figura 25. Diagrama de actividad

Y, para completar la fase de análisis, se ha realizado un primer esbozo sobre la interfaz de usuario de la aplicación *Android* y otra sobre la interfaz del bloque de *Moodle*. Para ello se han utilizado los diagramas de navegación. En la Figura 26 se muestra el diagrama de navegación de la aplicación *Android*.

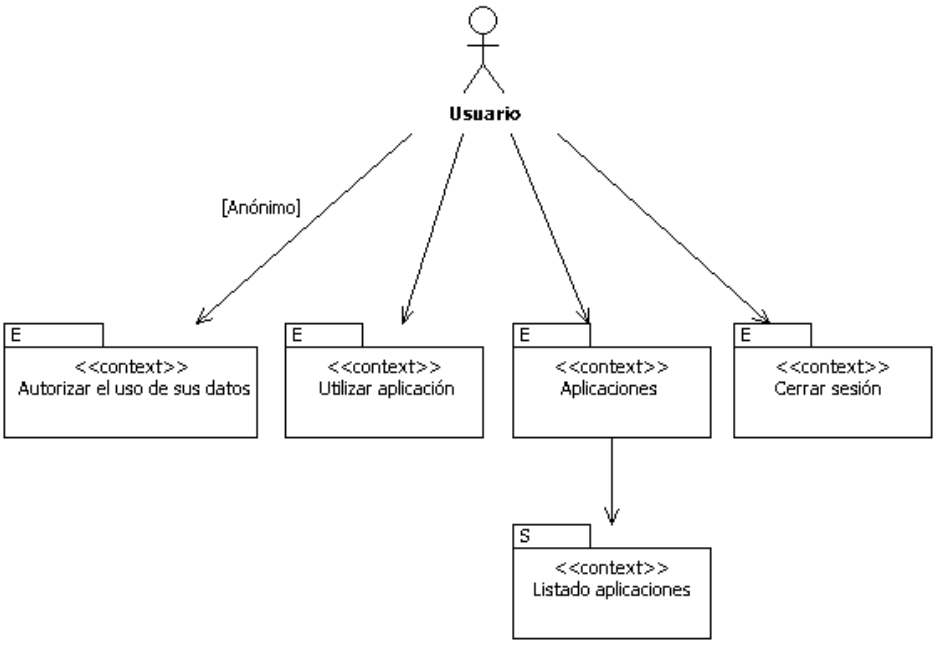


Figura 26. Diagrama de navegación

6.2. Diseño

El proceso de diseño se centra en el dominio de la solución. Para llegar a él se reexaminaron las clases del dominio del problema, refinándolas, extendiéndolas y reorganizándolas para poder mejorar su reutilización.

Se partió del material generado durante la fase de análisis, aumentando el grado de detalle, de tal manera que todos los pasos que hayan de darse durante la implementación queden bien especificados. El hecho de partir de las conclusiones halladas durante el análisis permite eliminar incongruencias entre ambas fases, para poder corregir errores que no hayan sido detectados anteriormente.

Toda la documentación correspondiente al diseño de las distintas partes del proyecto se recoge en el anexo 3 que se entrega en el CD adjunto con la memoria.

Para la creación de la aplicación *Android* se ha utilizado el patrón MVC. El patrón **MVC – Modelo Vista Controlador** divide la aplicación en tres partes: modelo, vista, controlador. El componente de modelo encapsula los datos y la funcionalidad central y es independiente de la entrada y la salida. Los componentes de vista presentan la información al usuario y obtiene datos del modelo, pudiendo haber múltiples vistas del modelo. Los controladores reciben la entrada, normalmente eventos que son trasladados para servir las peticiones del modelo o de la vista, de manera, que el usuario interactúa con el sistema sólo a través de los controladores.

La aplicación *Android* presentará el paradigma MVC, de la siguiente forma:

- **Modelo:** estará formado por las clases que se encargan de la lógica de la aplicación.
- **Vista:** estará formada por los archivos XML.
- **Controlador:** formado por las clases que reflejan las actividades, las cuales tienen asociadas un XML de la vista.

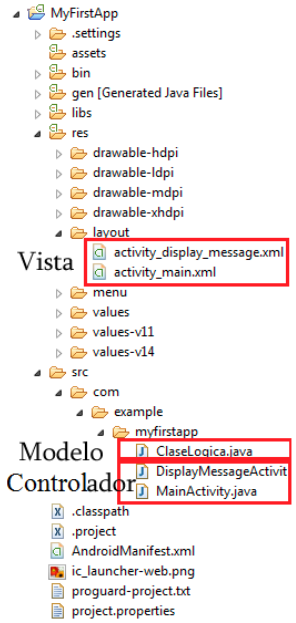


Figura 27. Patrón MVC en un proyecto de Android

A continuación, se adaptaron los escenarios obtenidos durante la fase de análisis. En la Figura 28 se puede ver la representación del escenario Autorizar el uso de sus datos.

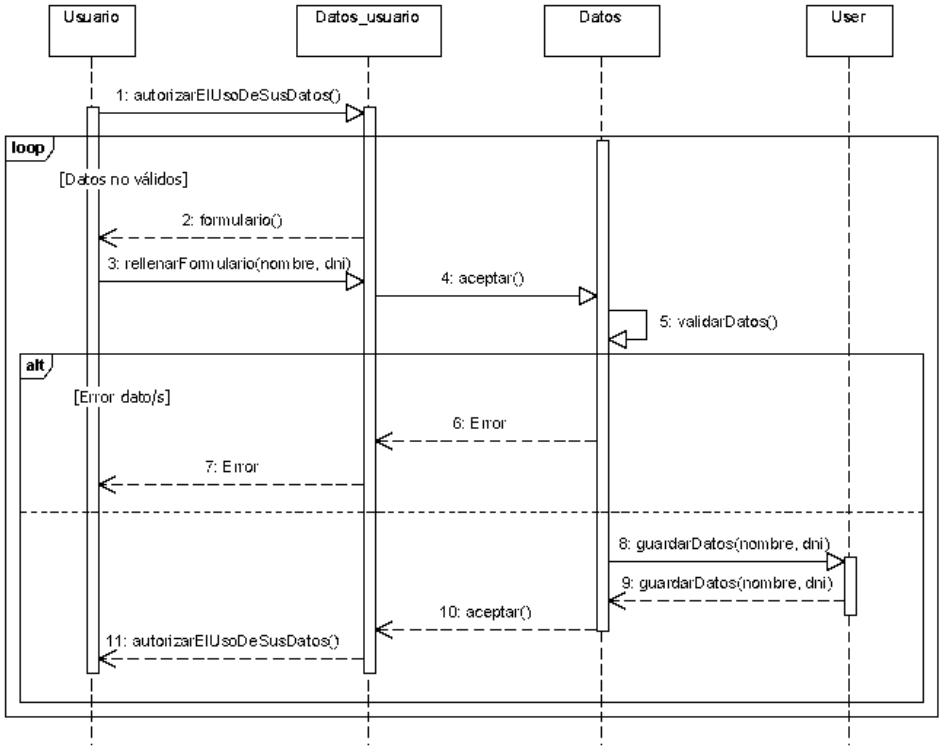


Figura 28. Escenario autorizar el uso de sus datos

Luego se estableció el diseño de datos. A lo largo del proyecto, solo se ha requerido establecer un diseño de datos para guardar la información sobre la actividad llevada a cabo en el PLE móvil en una nueva tabla creada en la base de datos de *Moodle*. La estructura de esta tabla puede verse en la Tabla 2.

ple	
+id	bigint (10)
+nameUser	varchar (20)
+dniUser	varchar (10)
+nameApp	varchar (11)
+fecha	varchar (11)
+hora	varchar (9)

Tabla 2. Tabla ple

- ***id:*** Identificador (Clave primaria)
- ***nameUser:*** Nombre de usuario
- ***dniUser:*** DNI de usuario
- ***nameApp:*** Nombre de la aplicación que ha sido utilizada por el usuario
- ***fecha:*** Fecha de uso de la aplicación por el usuario
- ***hora:*** Hora de uso de la aplicación por el usuario

Y para terminar esta fase se realizó el diseño de las interfaces que aparecerán en el proyecto, para lo que se ha intentado que sean lo más fáciles e intuitivas posibles con interacciones basadas en acciones físicas sobre elementos de código visual y tratamiento de errores.

6.3. Implementación

En este apartado, se explica lo más significativo de la fase de implementación de cada una de las partes del proyecto, así como los lenguajes de programación que se han utilizado en la implementación de las mismas.

6.3.1. Aplicación Android

La implementación de esta aplicación ha requerido el aprendizaje del desarrollo de las aplicaciones en *Android*, un mundo totalmente desconocido para la autora en ese momento.

Este proceso empezó con el conocimiento de *Android* buscando información en Internet para familiarizarse con él. Además, se realizó un curso on-line: "*Avances en el desarrollo de software para Android*" con el que realmente se empezaron a crear pequeñas aplicaciones realizando las distintas tareas para el curso y culminando con la realización de una aplicación final. De esta manera, ya se estaba preparada para desarrollar la aplicación *Android* para el presente proyecto.

El primer paso para construir el PLE móvil fue crear la interfaz de usuario. Todos los archivos que forman parte de la interfaz de una aplicación de *Android* se encuentran en una carpeta llamada *layout*, y son archivos XML que definen el diseño de las interfaces mediante el uso de elementos propios de la GUI.

Debido a la gran variedad de tamaños de pantallas en los múltiples dispositivos *Android* que existen, la interfaz principal que ofrece la aplicación tiene un *ScrollView*. De esta manera, en dispositivos en los que, por su tamaño, no caben todos los elementos que componen la interfaz en la pantalla, se permiten deslizamientos verticales (dedo arriba y abajo). En el caso de la lista de aplicaciones instaladas en el dispositivo, si la extensión de esta supera la pantalla, automáticamente se crea un *scroll* para que el usuario pueda desplazarse por toda la lista.

Por último, se tienen: la lógica de la aplicación, la creación del fichero de log en el que se almacena la información sobre la actividad llevada a cabo en el PLE móvil y la conexión a servicios web desde *Android*. Para implementar esto se hace uso del lenguaje de programación Java.

En la Figura 29, se puede ver un ejemplo de fichero de log.

Anónimo	Anónimo	Dropbox	15/01/2015	11:45:32
Sara	70936724P	Evernote	15/01/2015	11:47:40
Anónimo	Anónimo	Facebook	15/01/2015	11:55:19
Anónimo	Anónimo	Quickoffice	15/01/2015	11:58:03
Anónimo	Anónimo	Skype	15/01/2015	12:00:47
Laura	08557841S	Wikipedia	16/01/2015	13:12:33
Anónimo	Anónimo	Youtube	16/01/2015	13:15:00
Diana	08796236A	Youtube	16/01/2015	13:18:24
Diana	08796236A	Dropbox	16/01/2015	13:23:38
Diana	08796236A	Youtube	16/01/2015	13:28:56
Diana	08796236A	Skype	16/01/2015	16:44:10
Diana	08796236A	Facebook	17/01/2015	11:12:31
María	70996236M	Dropbox	20/01/2015	17:07:02
María	70996236M	Skype	20/01/2015	16:44:10
María	70996236M	Wikipedia	21/01/2015	09:23:59
Anónimo	Anónimo	Evernote	22/01/2015	11:45:32
Anónimo	Anónimo	Facebook	24/01/2015	20:31:00
Anónimo	Anónimo	Quickoffice	24/01/2015	20:45:04
Diana	08796236A	Dropbox	24/01/2015	21:33:18
Manuel	70843965X	Youtube	26/01/2015	10:22:01
Anónimo	Anónimo	Evernote	26/01/2015	11:11:25
Manuel	70843965X	Dropbox	26/01/2015	12:35:17
Diana	08796236A	Dropbox	26/01/2015	12:36:22
Anónimo	Anónimo	Facebook	27/01/2015	20:31:00
Anónimo	Anónimo	Quickoffice	27/01/2015	20:45:04
Diana	08796236A	Dropbox	27/01/2015	21:33:18
Manuel	70843965X	Youtube	27/01/2015	10:22:01
Anónimo	Anónimo	Evernote	28/01/2015	11:11:25
Manuel	70843965X	Dropbox	28/01/2015	12:35:17
Sara	70936724P	Evernote	29/01/2015	11:47:40
Anónimo	Anónimo	Facebook	29/01/2015	11:55:19
Anónimo	Anónimo	Quickoffice	29/01/2015	11:58:03
Anónimo	Anónimo	Skype	29/01/2015	12:00:47
Laura	08557841S	Wikipedia	29/01/2015	13:12:33
Anónimo	Anónimo	Gmail	30/01/2015	17:00:45
Anónimo	Anónimo	Youtube	30/01/2015	17:20:11
Anónimo	Anónimo	Dropbox	30/01/2015	18:31:26
Anónimo	Anónimo	Facebook	01/02/2015	10:06:44
Anónimo	Anónimo	Gmail	01/02/2015	14:03:32
Anónimo	Anónimo	Gmail	02/02/2015	11:58:05
Anónimo	Anónimo	Skype	02/02/2015	12:30:18
Anónimo	Anónimo	Dropbox	02/02/2015	19:47:49
Anónimo	Anónimo	Gmail	03/02/2015	17:00:45
Anónimo	Anónimo	Youtube	03/02/2015	17:20:11
Anónimo	Anónimo	Dropbox	03/02/2015	18:31:26
Anónimo	Anónimo	Facebook	03/02/2015	20:12:33
Anónimo	Anónimo	Gmail	04/02/2015	09:55:52
Anónimo	Anónimo	Gmail	04/02/2015	11:58:05
Anónimo	Anónimo	Skype	04/02/2015	12:30:18
Anónimo	Anónimo	Dropbox	04/02/2015	19:47:49
Anónimo	Anónimo	Gmail	04/01/2015	21:40:00
Anónimo	Anónimo	Youtube	04/01/2015	22:22:31
Anónimo	Anónimo	Dropbox	04/01/2015	22:51:16
Anónimo	Anónimo	Facebook	05/02/2015	10:06:39
Anónimo	Anónimo	Gmail	05/02/2015	10:53:00
Anónimo	Anónimo	Gmail	05/02/2015	11:58:05
Anónimo	Anónimo	Skype	05/02/2015	12:30:18
Anónimo	Anónimo	Dropbox	05/02/2015	13:07:01
Anónimo	Anónimo	Gmail	06/02/2015	20:43:09
Anónimo	Anónimo	Dropbox	06/02/2015	21:02:11

Figura 29: Ejemplo fichero de log

Los problemas surgidos durante la implementación de esta aplicación se derivan del total desconocimiento que se tenía de la programación de aplicaciones para dispositivos móviles con sistema operativo *Android.* Pero, debido a la gran documentación que hay sobre *Android*, estos problemas se han resuelto satisfactoriamente, aunque se debe mencionar que lo más costoso fue realizar la conexión con el servicio web. En la Figura 30, se muestra el código de la función que se ha creado para hacer dicha conexión.

```
private void WsPleConnection() throws ProtocolException, IOException {
    String token = "b34df953498faa3695f3da406f6d358a";
    String domainName = url;
    /// REST RETURNED VALUES FORMAT
    String restformat = "json";

    if (restformat.equals("json")) {
        restformat = "&moodlewsrestformat=" + restformat;
    } else {
        restformat = "";
    }
    String functionName = "local_wsple_insert_data_ple_in_bd";
    String urlParameters =
              "user=" + URLEncoder.encode(name_user, "UTF-8") +
              "&dni_user=" + URLEncoder.encode(dni_user, "UTF-8") +
              "&app=" + URLEncoder.encode(nombre_app, "UTF-8") +
              "&fecha=" + URLEncoder.encode(fecha_app, "UTF-8") +
              "&hora=" + URLEncoder.encode(hora_app, "UTF-8");

    /// REST CALL

    // Send request
    String serverurl = domainName + "/webservice/rest/server.php" + "?wstoken=" +
    token + "&wsfunction=" + functionName;
    HttpURLConnection con = (HttpURLConnection) new  URL(serverurl).openConnection();
    con.setRequestMethod("POST");
    con.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
    con.setRequestProperty("Content-Language", "en-US");
    con.setDoOutput(true);
    con.setUseCaches (false);
    con.setDoInput(true);
    DataOutputStream wr = new DataOutputStream (con.getOutputStream ());
    wr.writeBytes (urlParameters.toString());
    wr.flush ();
    wr.close ();
    //Get Response
    InputStream is =con.getInputStream();
    BufferedReader rd = new BufferedReader(new InputStreamReader(is));
    String line;
    StringBuilder response = new StringBuilder();
    while((line = rd.readLine()) != null) {
        response.append(line);
        response.append('\r');
    }
    rd.close();
}
```

Figura 30: Función WsPleConnection

6.3.2. Servicio web

El servicio web creado para la interacción con *Moodle* se encuentra en la carpeta *wsple*. En primer lugar el servicio procesa la petición recibida y se validan los parámetros (es decir, que lleguen los parámetros name_user, dni_user, name_app, desc_app, fecha y hora). Después se insertan los parámetros recibidos en la tabla de la base de datos de *Moodle* creada para ello. Y por último, devolverá si la inserción ha tenido éxito o no. A continuación, se describen los componentes del sistema y qué se hace en cada ellos.

- **wsple/version.php**. Archivo de configuración del *plugin* de *Moodle*. Facilita ciertas variables para determinar:

- **wsple/externallib.php**. Archivo que contiene la funcionalidad principal del servicio web. Se trata de una clase con el nombre del servicio web que extiende a external_api, clase en la que se define la estructura de los servicios web en *Moodle*. La clase que define los servicios web (*ws_insert_data_ple_in_bd*), en este caso, debe tener tres métodos básicos: uno para especificar los parámetros de entrada, otro para especificar los parámetros de salida y uno adicional con la lógica de negocio. En concreto, tiene los siguientes métodos:
  - *public static function insert_data_ple_in_bd_parameters ()*
  - *public static function insert_data_ple_in_bd (params)*
  - *public static function insert_data_ple_in_bd_returns ()*
- **wsple/lang**. Directorio de idiomas, necesario para los *plugins* pero que en este caso no tiene sentido y por ello aparece como vacío.
- **wsple/db/services.php.** Archivo de descripción del servicio web. Se deben especificar las funciones a incluir como servicio web, y por cada una de ellas la clase que lo implementa, el método concreto, la ruta a la clase, la descripción para la generación automática de documentación y el tipo de acción que permite.

```
$functions = array(
    'local_wsple_insert_data_ple_in_bd' => array(
    'classname'   => 'wsple_insertdata',
    'methodname'  => 'insert_data_ple_in_bd',
    'classpath'   => 'local/wsple/externallib.php',
    'description' => 'Inserta los datos en la base de datos de moodle',
    'type'        => 'write',
    )
);
```

  También debe incluirse la descripción del servicio para su instalación en Moodle. En concreto, el nombre con el que va a aparecer, las funciones que incluye (haciendo referencia a las descritas con anterioridad), si va a estar o no restringido para uno o varios usuarios específicos (independientemente de esto puede restringirse el uso del servicio web por Token) y si está habilitado para su instalación o no.

```
$services = array(
    'Myservice' => array(
    'functions' => array(
        'local_wsple_insert_data_ple_in_bd'
    ),
    'restrictedusers' => 0,
    'enabled' => 1,
    )
);
```

6.3.3. Bloque de Moodle

El bloque de *Moodle* realizado para el presente proyecto se encuentra en la carpeta *ple_activity* dentro de la carpeta *blocks* en *moodle*. Como ya se ha explicado anteriormente, este bloque consta de tres funcionalidades: dos de ellas para el consumo de la información, y una tercera que permite la explotación de la misma. A

continuación, se explica brevemente qué es un bloque en *Moodle*, así como las partes más importantes que constituyen el desarrollo del bloque para este proyecto.

En *Moodle* existen módulos llamados "bloques" que se colocan en las columnas de los laterales y que pueden tener cualquier cosa que se pueda agregar, tales como un calendario, un sitio de autenticación, y un gran número de cosas que no necesariamente tengan relación con lo que hay en el centro de la pantalla.

Para definir un bloque en *Moodle* necesitamos como mínimo cuatro archivos PHP. Estos archivos se encuentran ubicados en la carpeta *blocks/ple_activity*, y son: *block_ple_activity.php*, *db/access.php*, *lang/es/block_ple_activity.php* y *version.php*.

- **block_ple_activity.php:** Archivo que contiene la definición de la clase para el bloque. Sus métodos principales son:
  - *init()*: Método esencial para todos los bloques, cuyo propósito es dar valores a las variables miembro de la clase.
  - *get_content()*: Método es encargado de mostrar el contenido del bloque.
- **db/access.php:** Este archivo contiene las *capabilities* creadas para el bloque. A partir de la versión 2.4 de *Moodle* se introdujeron *addinstance* y *myaddinstance* para los bloques básicos. Además de estas dos *capabilities*, para el bloque del proyecto se han añadido otras tres, una por cada funcionalidad que ofrece el bloque, en la que se da permiso de uso del bloque al profesor ('editingteacher'). De esta manera, el bloque solo podrá ser utilizado por el profesor, y por supuesto, por el administrador de *Moodle*. A continuación, se muestra el código de la *capability* para poder gestionar un informe. Las otras dos son iguales a esta, solo cambia "managereport" por "viewreport" y "viewtodayreport", para ver todas las entradas y los registros de hoy, respectivamente.

```
'block/ple_activity:managereport' => array(
        'captype' => 'write',
        'contextlevel' => CONTEXT_SYSTEM,
        'archetypes' => array(
            'editingteacher' => CAP_ALLOW,
            'manager' => CAP_ALLOW
        )
    ),
```

- **lang/en/block_ple_activity.php:** Este es el archivo de idioma inglés para el bloque en el que se colocan todas las cadenas que se quieran utilizar en él. Se puede poner el idioma que se quiera para el bloque reemplazando "en" por el correspondiente código para cada idioma. Todos los archivos de idioma para los bloques se colocan en la subcarpeta **/lang** de la carpeta de instalación del bloque con su código de idioma adecuado. Por lo tanto, para poner el bloque en español, como es nuestro caso, el archivo de idioma correspondiente es: **lang/es/block_ple_activity.php**. *Moodle* 2.0 y superiores requieren el nombre del *plugin* para mostrarlo en la página de actualización, lo cual se escribe de la siguiente manera:

```
$String ['pluginname'] = Actividad en el PLE;
```

- **version.php:** Archivo que tiene información de la versión del *plugin*. En realidad, este archivo es muy simple, contiene sólo unas pocas definiciones de

campos, en función de sus necesidades. En nuestro caso, el número de versión del bloque y la versión mínima de *Moodle* que se debe instalar para poder utilizarlo:

```
$plugin->version   = 2015070823;
$plugin->requires  = 2013110500;
```

Además de estos cuatro archivos, el bloque contiene otros siete: *viewreport.php, viewtodayreport.php, managereport.php, ple_form.php, pdf_completo.php, pdf_hoy.php y pdf_manager.php*. Los tres primeros se utilizan para gestionar y mostrar la información que se corresponde con cada una de las funcionalidades que ofrece el bloque en la página que corresponda en Moodle; el cuarto, se corresponde con el formulario en el que el profesor pondrá sus condiciones para gestionar un informe; y los tres últimos, son los encargados de generar los PDFs de los distintos tipos de informes: todas las entradas, registros de hoy y gestionados, respectivamente.

6.4. Pruebas

Durante todo el proceso de desarrollo de las distintas partes del proyecto se han ido realizando las pruebas oportunas para cada uno de los hitos que se impusieron durante la fase de implementación.

Todas y cada una de las partes que componen el proyecto han pasado pruebas de datos erróneos, conexión con servidores apagados, falta de conexión a Internet y pruebas similares.

Además, en el caso de la aplicación del PLE móvil, debido a la fragmentación de *Android*, se han realizado pruebas de la aplicación en pantallas de diferentes tamaños (*tablets*), y se han preparado interfaces para su funcionamiento dependiendo de la rotación de la pantalla, como puede verse en la Figura 31. Esta fragmentación, también ha llevado a probar la aplicación en varios dispositivos con distintas versiones de *Android*. Cabe destacar que la aplicación se ha programado para que soporte como mínimo la versión 8 de la API de *Android*, correspondiente a la versión 2.2 del sistema operativo (*Froyo*).

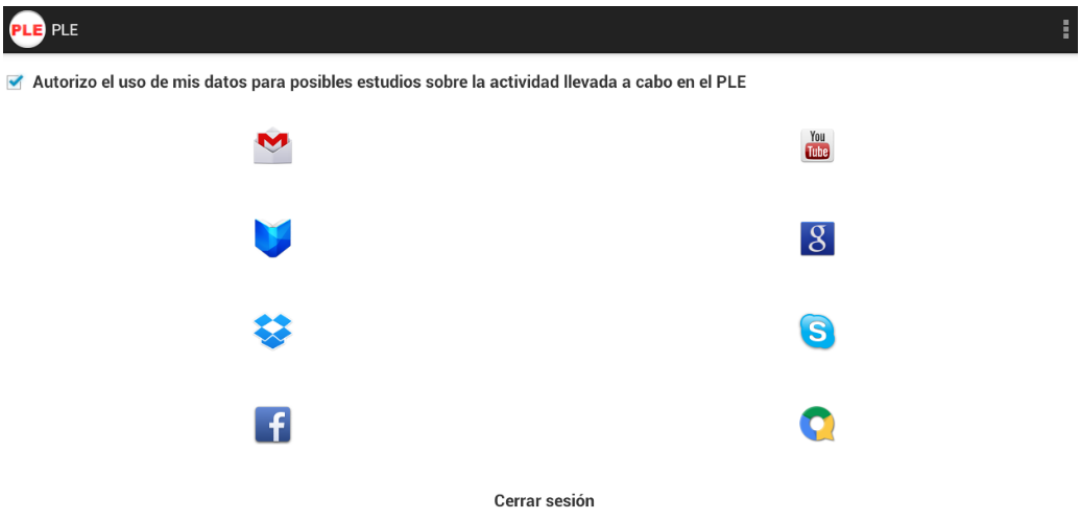


Figura 31. Interfaz PLE en modo *landscape*

7. Trabajos relacionados

Debido al gran auge que han tenido las aplicaciones *Android* en la sociedad, existen multitud de trabajos relacionados con el presente proyecto. Por este motivo, a pesar de que en un proyecto final de carrera de primer ciclo no parece obligada la presencia de este apartado, se ha querido incluir un pequeño resumen de algunos de los trabajos ya realizados que tienen características comunes o están relacionados con el proyecto que se ha desarrollado.

7.1. Implementación de un framework de servicios para facilitar la interoperabilidad entre Moodle y un PLE basado en widgets

Proyecto fin de carrera creado por Alberto del Pozo de Dios para la titulación de Ingeniería Informática (*de Dios, 2012*). Dicho proyecto consiste en la creación de un framework de comunicación que facilita la integración entre LMS y PLE basado en servicios web y especificaciones de interoperabilidad, para lo que el autor ha creado una serie de herramientas, widgets y aplicaciones *Android*. En la Figura 32 puede verse una captura de la interfaz de una de las herramientas que se han creado en este proyecto.

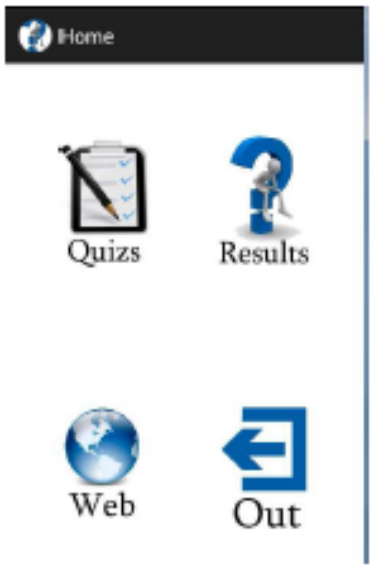


Figura 32. Interfaz de la aplicación de cuestionarios: de Dios, 2012

7.2. Personalización del aprendizaje: Framework de servicios para la integración de aplicaciones online en los sistemas de gestión del aprendizaje

Tesis doctoral realizada por el cotutor de este proyecto, el Dr. D. Miguel Ángel Conde González (*Conde, 2012*). En ella se pretende dar una aproximación para facilitar la interoperabilidad entre LMS y PLE y facilitar el intercambio de información e interacción entre ambos entornos. Dicha tesis doctoral ha sido dirigida por el tutor del presente proyecto, el Dr. D. Francisco José García Peñalvo, director del Grupo GRIAL, y el Dr. Marc Alier Forment, director del grupo SUSHITOS, como resultado de una estrecha relación entre ambos grupos y por ende de la Universidad de Salamanca y la Universidad Politécnica de Cataluña.

7.3. Cliente Android para Moodle

Proyecto fin de carrera creado por Alfonso Bocanegra de Luis para la titulación Ingeniería Técnica en Informática de Gestión en la Facultad de Informática de Barcelona (Universidad Politécnica de Cataluña) (*de Luis, 2012*). Este proyecto consiste en el desarrollo de una aplicación para sistemas móviles *Android* que permita a los usuarios del ambiente educativo virtual Moodle 2.0 acceder a los contenidos y actividades más utilizados desde dispositivos móviles a través de servicios web. En la Figura 33 se puede ver la interfaz inicial de la aplicación creada en este proyecto.

Figura 33. Interfaz inicial de la aplicación del proyecto: de Luis, 2012

7.4. Aplicación para realizar cuestionarios desde dispositivos Android

Proyecto fin de carrera creado por Sarah Bouayad para la titulación Ingeniería Técnica en Informática en la Facultad de Informática de Barcelona (Universidad Politécnica de Cataluña) (*Bouayad, 2013*).. Este proyecto es una ampliación del proyecto citado anteriormente *"Cliente Android para Moodle"* y consiste en permitir a los usuarios contestar a los cuestionarios de Moodle desde dispositivos móviles. Ambos proyectos están englobados dentro del proyecto Moodbile (*Moodbile, 2015*), cuyo objetivo es desarrollar y mantener la API (Application Programming Interface) de los servicios web de Moodle para que aplicaciones externas puedan colaborar con el servidor de Moodle. En la Figura 34 se muestra un ejemplo de pregunta de un cuestionario de este proyecto.

Figura 34. Interfaz de un cuestionario de la aplicación del proyecto: Bouayad, 2013

8. Conclusiones

La evolución de los procesos de aprendizaje en general, y del aprendizaje electrónico (e-learning) en particular, en relación al desarrollo de las tecnologías supone cambios en cuanto a las tecnologías que usan los profesores para enseñar y los alumnos para aprender. Además, el aprendizaje a través del uso de la tecnología móvil genera interés en el alumnado, lo que posibilita el aprendizaje en cualquier momento y en cualquier lugar (m-learning). Dado este contexto y el desarrollo del presente proyecto deberían considerarse dos tipos de entornos de aprendizaje: los LMS y los PLE.

Por un lado, los LMS están centrados en la institución, es decir, el aprendizaje está controlado y está compuesto por un conjunto predeterminado de herramientas. Por otro lado, los PLE son personales, suponen el aprendizaje a lo largo de la vida con las herramientas y recursos que el usuario desea utilizar.

Dos entornos de aprendizaje diferentes suponen dos contextos diferentes a los que el usuario debe acceder. Sin embargo, los PLE no van a reemplazar a los LMS, ambos entornos coexisten y pueden interoperar entre sí. Teniendo en cuenta cómo ambos entornos interoperan, en el presente proyecto se ha realizado un seguimiento de la actividad del usuario en el PLE móvil (Android). Este seguimiento es enviado a un LMS (Moodle) a través de un servicio web, de manera que el profesor pueda acceder a esa información a través de la platafoma. Por lo tanto, los alumnos solo tendrán que acceder al PLE móvil y el profesor accederá al LMS.

En este proyecto fin de carrera se ha conseguido desarrollar un sistema de monitorización de la actividad llevada a cabo en un entorno personalizado de aprendizaje móvil. Para ello, se ha creado una aplicación *Android*, una herramienta para *Moodle* y un servicio web para la interacción entre *Android* y el LMS.

En primer lugar, se ha definido un modelo de monitorización que permite recuperar las peculiaridades de las herramientas que pueden incluirse en un PLE y de la actividad del usuario en ellas. Para llevar a cabo esta tarea, se ha creado una aplicación *Android* que permite reunir un total de ocho aplicaciones, elegidas por el usuario entre las que tenga instaladas en el dispositivo, en una única interfaz, constituyendo así un PLE móvil. Los usuarios en un dispositivo móvil pueden utilizar las distintas aplicaciones incluidas en el PLE de manera anónima ó identificándose. Además, se ha establecido un modelo de gestión y almacenamiento de la actividad, para lo que se ha creado un fichero de log que se almacena internamente en la aplicación y contiene información sobre los usuarios que utilizan el PLE móvil (anónimos e identificados), las aplicaciones que utilizan (nombre) y cuándo las utilizan (fecha y hora).

Después, se ha desarrollado un servicio web que es el encargado de realizar la interacción con el entorno institucional, para lo que se ha elegido *Moodle* como LMS. Dicho servicio inserta la información recibida sobre la actividad llevada a cabo en el PLE móvil en la base de datos de *Moodle*.

Y por último, se ha creado un bloque para la plataforma de aprendizaje *Moodle* que será utilizado por el profesor. Este se ha instalado en *Moodle* como un *plugin* y proporciona una representación flexible de la información, lo que permite al profesor estudiar la influencia de la actividad en el móvil en las notas de los estudiantes y facilita la toma de decisiones. Las funcionalidades que ofrece dicho bloque (todas las entradas, registros de hoy y gestionar informe) permiten al profesor la visualización de distintos

tipos de informes, así como filtrar la información sobre la actividad llevada a cabo en el PLE móvil (por fecha, aplicación, usuario, etc.), de manera que puede acceder a la información concreta que desee. Además de esto, podrá exportar a formato PDF los informes.

Aunque se ha conseguido cumplir con todos los objetivos planteados al comienzo del desarrollo del proyecto, en este tipo de trabajos siempre existen opciones de mejora. Una de estas opciones es mejorar la interfaz gráfica, tanto de la aplicación Android como del bloque de Moodle, ya que es uno de los aspectos de un sistema informático que siempre admite mejora, aunque se ha intentado crear interfaces simples y atractivas.

La realización de todas y cada una de las partes que constituyen este proyecto ha supuesto una gran experiencia, tanto a nivel técnico como personal. En él, se han puesto en práctica muchos de los conocimientos adquiridos a lo largo de la carrera y ha permitido a la autora el aprendizaje de varias tecnologías:

- Iniciación en el mundo del desarrollo de aplicaciones *Android*, un mundo, hasta ese momento, totalmente desconocido para la autora del proyecto.
- Conocimiento y mejora del uso de varios lenguajes de programación: Java, XML, SQL, PHP y HTML.
- Conocimiento del funcionamiento de las plataformas de aprendizaje a nivel de administrador, especialmente de *Moodle*, y los servicios web.

9. Líneas futuras de trabajo

En este apartado se hacen una serie de sugerencias de posibles líneas futuras del proyecto. Estas son:

- **Versión IOS**: creación del PLE móvil para dispositivos móviles que corran bajo el sistema operativo IOS. Para llevar a cabo esta mejora, sólo sería necesario cambiar el cliente, es decir, la implementación del PLE móvil para IOS, el servicio web y el bloque de Moodle no requieren cambios.
- **Planteamiento de actividades en el PLE móvil desde Moodle**: Para llevar a cabo esta mejora, sería necesario añadir una nueva funcionalidad al bloque de Moodle (o crear una actividad) en la que el profesor pueda escribir los planteamientos, crear un servicio web para enviar la información desde Moodle y añadir otra funcionalidad en el PLE móvil para recibirlos y mostrarlos.

10. Lista de acrónimos

En este apartado se hace una lista de todos los acrónimos que aparecen en la memoria del proyecto. Esta lista seguirá la siguiente estructura:

ACRÓNIMO: *Significado en inglés* (traducción al español si está en inglés).

Dicho esto, los acrónimos utilizados en este proyecto ordenados alfabéticamente son los siguientes:

ANSI: ***A****merican* ***N****ational* ***S****tandards* ***I****nstitute* (Instituto Nacional Americano de Estándares)

API: **A**pplication **P**rogramming **I**nterface (Interfaz de Programación de Aplicaciones)

CASE: ***C****omputer* ***A****ided* ***S****oftware* ***E****ngineering* (Ingeniería de Software Asistida por Computadora)

GNU: ***G****NU is* ***N****ot* ***U****nix* (GNU No es Unix)

DDL: ***D****ata* ***D****efinition* ***L****enguage* (Sublenguaje de definición de datos)

DML: ***D****ata* ***M****anipulation* ***L****anguage* (Sublenguaje de manipulación de datos)

DCL: ***D****ata* ***C****ontrol* ***L****anguage* (Sublenguaje de control de datos)

GPL: ***G****eneral* ***P****ublic* ***L****icense* (Licencia Pública General)

GUI: **G**raphic **U**ser **I**nterface (Interfaz Gráfica de Usuario)

HTML: ***H****iper****T****ext* ***M****arkup* ***L****anguage* (Lenguaje de Marcas de HiperTexto)

HTTP: ***H****iper****T****ext* ***T****ransfer* ***P****rotocol* (Protocolo de Transferencia de Hipertexto)

IDE: ***I****ntegrated Development Environment* (Entorno de Desarrollo Integrado)

IEEE: **I***nstitute of* ***E****lectrical and* ***E****lectronic* ***E****ngineers* (Instituto de Ingenieros Eléctricos y Electrónicos)

ISO: ***I****nternational* ***O****rganization for* ***S****tandardization* (Organización Internacional para la Estandarización)

JSON: ***J****ava****S****cript* ***O****bject* ***N****otation* (Notación de Objetos de JavaScript)

LMS: ***L****earning* ***M****anagement* ***S****ystem* (Sistema de Gestión de Aprendizaje)

Moodle: ***M****odule* ***O****bject-****O****riented* ***D****ynamic* ***L****earning* ***E****nvironment* (Entorno Modular de Aprendizaje Dinámico Orientado a Objetos)

PDF: ***P****ortable* ***D****ocument* ***F****ormat* (Formato de documento portátil)

PHP: ***P****HP* ***H****ipertext* ***P****re-processor* (Preprocesador de Hipertexto)

PLE: ***P****ersonal* ***L****earning* ***E****nvironment* (Entorno Personal de Aprendizaje)

REM: ***RE****quirements* ***M****anagement* (Gestión de Requisitos)

REST: ***R****Epresentational* ***S****tate* ***T****ransfer* (Transferencia de Estado Representacional)

SDK: ***S****oftware* ***D****evelopment* ***K****it* (Kit de Desarrollo Software)

SQL: ***S****tructured* ***Q****uery* ***L****anguage* (Lenguaje de Consultas Estructurado)

UML: ***U****nified* ***M****odeling* ***L****anguage* (Lenguaje Unificado de Modelado)

URL: ***U****niform* ***R****esource* ***L****ocator*(Localizador de Recursos Uniforme)

W3C: ***W****orld* ***W****ide* ***W****eb* ***C****onsortium* (Consorcio de la Red Global Mundial)

XAMPP: ***X*** (para cualquiera de los diferentes sistemas operativos) ***A****pache* ***M****ySQL* ***P****hp* ***P****erl*

XML: *e****X****tensible* ***M****arkup* ***L****anguage* (Lenguaje de Marcas eXtensible)

11. Bibliografía

Adell, J., & Castañeda, L. (2010). Los Entornos Personales de Aprendizaje (PLEs): una nueva manera de entender el aprendizaje. In R. Roig Vila & M. Fiorucci (Eds.), *Claves para la investigación en innovación y calidad educativas. La integración de las Tecnologías de la Información y la Comunicación y la Interculturalidad en las aulas. Stumenti di ricerca per l'innovaziones e la qualità in ámbito educativo. La Tecnologie dell'informazione e della Comunicaziones e l'interculturalità nella scuola.* Alcoy, Spain: Marfil – Roma TRE Universita degli studi.

Android. (2015a). Android. Retrieved January 2015, from http://www..android.com/

Android. (2015b). The Android Story. Retrieved January 2015, from http://www..android.com/history/

Android. (2015c). Dashboards. Retrieved January 2015, from http://developer.android.com/about/dashboards/index.html

Android SDK. (2015). Android SDK. Retrieved January 2015, from http://developer.android.com/sdk/1.5_r2/index.html

Blanco, A. (2009). *Introducción a Google API Chart*. Retrieved January 2015, from http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=googleChart

Bouayad, S. (2013). *Aplicación para realizar cuestionarios desde dispositivos Android*. Ingeniería Informática, Universidad Politécnica de Cataluña.

Calle, E. (2009). Un vistazo a XML. Retrieved January 2015, from http://www.oocities.org/infinitobo/temainves/xml.html

Conde, M. Á. (2012). *Personalización del aprendizaje: Framework de servicios para la integración de aplicaciones online en los sistemas de gestión del aprendizaje.*

Doctorado en Informática y Automática Tesis Doctoral, Universidad de Salamanca, Salamanca. Retrieved January from http://grialdspace.usal.es:443/handle/grial/223

de Dios, A. d. P. (2012). *Implementación de un framework de servicios para facilitar la interoperabilidad entre Moodle y un PLE basado en widgets.* Ingeniería Informática Proyecto final de carrera, Universidad de Salamanca, Salamanca.

de Luis, A. B. (2012). *Cliente Android para Moodle*. Ingeniería Técnica Informática de Gestión Proyecto final de carrera, Universidad Politécnica de Cataluña.

e-Learning, P. A. d. (2015). Qué es e-Learning. Retrieved January 2015, from http://prometeo3.us.es/publico/es/quees/MS_25.html

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures.*, University of California, Irvine.

Google. (2015). Google. Retrieved January 2015, from https://www.google.es/intl/es/about

Google Charts. (2015). Google Charts. Retrieved January 2015, from https://developers.google.com/chart/

Jacobson, I., Booch, G., Rumbaugh, J. The Unified Software Development Process. Object Technology Series. Addison-Wesley, 1999

JSON. (2015). Presentación de JSON. Retrieved January 2015, from http://www.json.org/

Moodbile. (2015). Retrieved January 2015, from http://www.moodbile.org/

Moodle. (2015a). Acerca de Moodle. Retrieved January 2015, from http://docs.moodle.org/all/es/Acerca_de_Moodle

Moodle. (2015b). Moodle statistics. Retrieved January 2015, from http://moodle.org/stats

Open Handset Alliance. (2015). Open Handset Alliance. Retrieved January 2015, from http://www.openhandsetalliance.com/

Oracle. (2015). Java Language and Virtual Machine Specifications Retrieved January 2015, from http://docs.oracle.com/javase/specs/

PHP. (2015). Manual de PHP. Retrieved January 2015, from http://php.net/manual/es/

Pardo, A. M. S., & Peñalvo, F. J. G. (2015). Introducción al e-learning. Retrieved January 2015, from http://antia.fis.usal.es/sharedir/TOL/introelearning/

Peñalvo, F. J. G. (2005). Estado actual de los sistemas e-learning. *Teoría de la Educación. Educación y Cultura en la Sociedad de la Información*, *6*(2).

Peñalvo, F. J. G., Raedo, J. M. M., Velthuis, M. G. P., Giner, J. R. G.-B., & García, M. N. M. (2000). Proyecto de Final de Carrera en la Ingeniería Técnica en Informática: Guía de Realización y Documentación (Versión 1.52 ed.).

Raggett, D. (2005). W3C. Getting started with HTML. Retrieved January 2015, from http://www.w3.org/MarkUp/Guide/

Rumbaugh, J., Jacobson, I., & Booch, G. (1999). *The Unified Modeling Language. Reference Manual* (2nd ed.): Pearson Higher Education.

Schaffert, R., & Hilzensauer, W. (2008). On the way towards Personal Learning Environments: Seven crucial aspects. *eLearning papers*, *2*(9), 1-11. doi: citeulike-article-id:8361564

Sevilla, U. d. (2015). Lenguajes y Sistemas Informáticos. Retrieved January 2015, from http://www.lsi.us.es/descargas/descarga_programas.php?id=3

TCPDF. (2015). TCPDF. Retrieved November 2015, from http://www.tcpdf.org

Toro, A. D. (2000). *Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información.* Universidad de Sevilla, Sevilla. Retrieved January 2015, from http://fondosdigitales.us.es/media/thesis/30/O_Tesis-18.pdf

Toro, A. D., & Jiménez, B. B. (2000). Metodología para la Elicitación de Requisitos de Sistemas Software.

W3C. (2015). Guía Breve de Servicios Web. Retrieved January 2015, from http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb

W3Schools (2015). SQL Tutorial. Retrieved January 2015, from http://www.w3schools.com/sql/

Wilson, S., Liber, O., Johnson, M., Beauvoir, P., Sharples, P., & Milligan, C. (2007). Personal Learning Environments: Challenging the dominant design of educational systems *Journal of e-Learning and Knowledge Society, 3*(3), 27-38.