



---

**Universidad de Valladolid**

E.T.S.I. TELECOMUNICACIÓN

# TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS ESPECÍFICAS DE  
TELECOMUNICACIÓN

MENCIÓN EN INGENIERÍA DE SISTEMAS DE TELECOMUNICACIÓN

## **Aplicación Android de comunicación a través de Moodle**

Autor:

**Dña. Elena María Campos Gutiérrez**

Tutor:

**Dña. Miriam Antón Rodríguez**

Valladolid, 18 de Mayo de 2015



**TÍTULO:** APLICACIÓN ANDROID DE COMUNICACIÓN A TRAVÉS DE MOODLE

**AUTOR:** ELENA MARÍA CAMPOS GUTIÉRREZ

**TUTORA:** MÍRIAM ANTÓN RODRÍGUEZ

**DEPARTAMENTO:** TEORÍA DE LA SEÑAL Y COMUNICACIONES E INGENIERÍA TELEMÁTICA

**Miembros del tribunal**

**PRESIDENTE:** MÍRIAM ANTÓN RODRÍGUEZ

**SECRETARIO:** DAVID GONZÁLEZ ORTEGA

**VOCAL:** MARIO MARTÍNEZ ZARZUELA

**SUPLENTE:** MARÍA ÁNGELES PÉREZ JUÁREZ

**FECHA DE LECTURA:** 18 Mayo 2015

**CALIFICACIÓN:**



## **RESUMEN DEL PROYECTO**

La aplicación de las Tecnologías de la Información y la Comunicación (TIC) al sistema educativo, también conocido como *e-Learning*, se desarrolla gracias al empleo de múltiples plataformas. Estas plataformas permiten la formación de los alumnos más allá de las horas lectivas, así como la comunicación entre alumnos y profesores.

El proceso de enseñanza-aprendizaje mejorará y será más eficaz si las herramientas y metodologías empleadas se adaptan de forma adecuada al escenario educativo concreto, teniendo en cuenta las condiciones de contorno impuestas así como los intereses y capacidades de profesores y alumnos. Uno de los fenómenos que está siendo ampliamente aceptado debido a la gran penetración de los dispositivos móviles con amplias funcionalidades es el *Mobile Learning* (aprendizaje a través de un dispositivo móvil).

En este Trabajo Fin de Grado se propone el desarrollo de una aplicación móvil *Android* que permita realizar tanto una comunicación efectiva entre los miembros de un curso mediante foros de discusión, la evaluación de conocimientos a través de cuestionarios y por último la integración de la versión anterior, que incluye autenticación, mensajería, visualización de cursos o asignaturas en los cuales se esté matriculado, creación de nuevo usuario y descarga de recursos. Esta herramienta funcionará haciendo uso de la plataforma de aprendizaje *Moodle*. El dispositivo y la plataforma *Moodle* se comunicarán implementando servicios web.

## **PALABRAS CLAVE**

Android, Base de datos, JSON, Moodle, PHP, REST.

## **ABSTRACT**

The application of information and communications technology (ICT) into the education system, also known as e-Learning, is developed through the use of multiple platforms. These platforms allow the formation of students beyond school hours, as well as communication between students and teachers.

The teaching-learning process will improve and be more effective if the tools and methodologies adapt to the specific educational scenario, taking into account the boundary conditions imposed and the interests and abilities of teachers and students. One phenomenon that is widely accepted due to the high penetration of mobile devices with extensive functionality is the Mobile Learning (learning through a mobile device).

This final project proposes to develop an Android mobile application that allows for effective communication between members of a course through discussion forums, knowledge assessment through questionnaires and finally the integration of the previous version that included authentication, messaging, display courses in which they are enrolled, creating new user and downloading resources. This tool will function using the learning platform Moodle. The device and the Moodle platform shall implement web services.

## **KEYWORDS**

Android, Data Base, JSON, Moodle, PHP, REST.



*Agradecer a mis padres por haber hecho posible que llegue a donde estoy.*

*A mi hermano por el apoyo recibido a lo largo de mi carrera.*

*A mis compañeros de carrera “ Argucieros” que han sido un gran apoyo en este camino.*

*En general a toda mi familia y amigos que han confiado en mi.*

*Por último a mi tutora Míriam que me ha facilitado en todo lo posible la realización de este proyecto.*



# ÍNDICE DE CONTENIDOS

RESUMEN DEL PROYECTO.....	i
PALABRAS CLAVE.....	i
ABSTRACT.....	ii
PALABRAS CLAVE.....	ii
AGRADECIMIENTOS.....	iv
ÍNDICE DE CONTENIDOS.....	vi
ÍNDICE DE FIGURAS.....	xii
ÍNDICE DE TABLAS.....	xv
CAPÍTULO I. INTRODUCCIÓN.....	1
1. INTRODUCCIÓN GENERAL.....	4
2. MOTIVACIÓN.....	3
3. OBJETIVOS.....	5
4. INTRODUCCIÓN AL DOCUMENTO.....	6
CAPÍTULO II. JUSTIFICACIÓN DE LAS TECNOLOGÍAS EMPLEADAS.....	8
1. INTRODUCCIÓN.....	10
2. SISTEMAS OPERATIVOS MÓVILES.....	10
2.1 ANDROID.....	11
2.2 iOS.....	13
2.3 OTROS SISTEMAS OPERATIVOS MÓVILES.....	14
2.3.1 WINDOWS PHONE.....	14
2.3.2 FIREFOX OS.....	14
2.3.3 BLACKBERRY.....	15
2.3.4 UBUNTU TOUCH.....	15

2.3.5 TIZEN.....	15
2.4 ELECCIÓN....	16
3. INTRODUCCIÓN A LAS PLATAFORMAS DE E-LEARNING.....	16
3.1 MOODLE.....	17
3.2 OTRAS PLATAFORMAS DISPONIBLES.....	17
3.3 ELECCIÓN DE LA PLATAFORMA E-LEARNING.....	18
<b>CAPÍTULO III. MOODLE.....</b>	<b>20</b>
1. INTRODUCCIÓN.....	22
2. CARACTERÍSTICAS.....	23
2.1 CARACTERÍSTICAS GENERALES.....	23
2.2 CARACTERÍSTICAS ADMINISTRATIVAS.....	24
2.3 CARACTERÍSTICAS PARA EL DESARROLLO Y GESTIÓN DEL CURSO.....	24
3. ARQUITECTURA.....	26
3.1 INTRODUCCIÓN.....	26
3.2 ESTRUCTURA.....	27
3.3 MÓDULOS DE ACTIVIDADES.....	29
3.4 TEMAS.....	30
3.5 IDIOMAS.....	30
3.6 ESQUEMA DE LA BASE DE DATOS.....	31
4. OTRAS CARACTERÍSTICAS DE MOODLE.....	31
5. VERSIONES DE MOODLE.....	34
6. MOODLE MÓVIL.....	35
<b>CAPÍTULO IV. SERVICIOS WEB.....</b>	<b>38</b>
1. INTRODUCCIÓN A LOS SERVICIOS WEB.....	40
2. XML: EXTENSIBLE MARKUP LANGUAGE.....	41

3. WSDL.....	42
4. UDDI.....	42
5. RELACIÓN DE LOS SERVICIOS WEB CON OTRAS TECNOLOGÍAS.....	43
6. SOAP.....	44
7. RESTful.....	45
8. ELECCIÓN.....	45
9. REPRESENTACIÓN DE LOS DATOS.....	46
<b>CAPÍTULO V. DESARROLLO DEL PROYECTO.....</b>	<b>49</b>
1. CONDICIONES PREVIAS.....	51
2. APLICACIÓN MÓVIL. DISEÑO DE LA INTERFAZ.....	51
2.1 INTRODUCCIÓN.....	51
2.2 INTERFAZ DE LA APLICACIÓN MÓVIL.....	53
3. ESTRUCTURA DE LA BASE DE DATOS DE MOODLE.....	53
Tablas relacionadas con el módulo de foros de Moodle:.....	54
Mdl_forum.....	54
Mdl_forum_discussions.....	55
Mdl_forum_posts.....	55
Tablas relacionadas con el módulo de cuestionarios de Moodle:.....	56
Mdl_quiz.....	56
Mdl_quiz_slots.....	56
Mdl_quiz_attempts.....	57
Tablas relacionadas con el módulo de preguntas de Moodle:.....	58
Mdl_question.....	58
Mdl_question_answers.....	58
Mdl_qtype_multichoice_options.....	58
Mdl_question_attempts.....	59

<b>Mdl_question_attempt_steps .....</b>	<b>59</b>
<b>Mdl_question_attempt_stept_data .....</b>	<b>60</b>
<b>Mdl_question_usages.....</b>	<b>60</b>
<b>Mdl_context.....</b>	<b>61</b>
<b>AnswerQuestion.....</b>	<b>62</b>
<b>4. CONEXIÓN DE LA APLICACIÓN MÓVIL CON EL SERVIDOR WEB.....</b>	<b>64</b>
<b>4.1 INTRODUCCIÓN.....</b>	<b>64</b>
<b>4.2 EL PROTOCOLO HTTP.....</b>	<b>64</b>
<b>Método Get.....</b>	<b>65</b>
<b>Método Head.....</b>	<b>65</b>
<b>Método Post.....</b>	<b>65</b>
<b>Método Put.....</b>	<b>65</b>
<b>Método Delete.....</b>	<b>65</b>
<b>4.3 EL FORMATO DE DATOS JSON.....</b>	<b>65</b>
<b>Objeto JSON.....</b>	<b>66</b>
<b>Array JSON.....</b>	<b>66</b>
<b>Valor JSON.....</b>	<b>67</b>
<b>4.4 DIAGRAMA DE SECUENCIA.....</b>	<b>68</b>
<b>4.4.1 FOROS.....</b>	<b>68</b>
<b>Visualización de discusiones .....</b>	<b>70</b>
<b>Visualización de mensajes de las discusiones.....</b>	<b>70</b>
<b>Creación de respuesta a un mensaje.....</b>	<b>70</b>
<b>Creación de una nueva discusión .....</b>	<b>71</b>
<b>4.4.2 CUESTIONARIOS.....</b>	<b>71</b>
<b>Visualización de cuestionarios.....</b>	<b>73</b>
<b>Visualización de intentos cuestionarios.....</b>	<b>73</b>
<b>Generación de un intento nuevo o actualización de un intento en curso.....</b>	<b>73</b>

Salir de un intento.....	74
Enviar y terminar un intento.....	75
<b>CAPÍTULO VI. MANUAL DE USUARIO.....</b>	<b>77</b>
1. PANTALLA DE REGISTRO DE NUEVO USUARIO.....	79
2. PANTALLA DE INICIO.....	80
2.1. PANTALLA MENÚ PRINCIPAL.....	81
2.1.1. MENSAJERIA.....	82
Pantalla menú mensajes.....	82
Pantalla mensajes no leídos.....	82
Pantalla mensajes leídos .....	83
Pantalla enviar nuevo mensaje.....	84
Pantalla mensajes enviados.....	85
2.1.2. CURSOS O ASIGNATURAS.....	86
Pantalla de secciones.....	86
Pantalla de cursos.....	86
Pantalla de opciones.....	87
Pantalla foros.....	88
Pantalla de discusiones.....	89
Pantalla nueva discusión.....	90
Pantalla de mensajes de discusión.....	91
Pantalla respuesta de un mensaje.....	91
Pantalla cuestionarios.....	92
Pantalla de intentos.....	93
Pantalla de preguntas.....	94
Pantalla dinámica de preguntas .....	95
Pantalla recursos.....	97

<b>CAPÍTULO VII. CONCLUSIONES Y LÍNEAS FUTURAS.....</b>	<b>100</b>
<b>1. CONCLUSIONES.....</b>	<b>102</b>
<b>2. LÍNEAS FUTURAS.....</b>	<b>103</b>
<b>BIBLIOGRAFÍA.....</b>	<b>104</b>
<b>ANEXO I. VERSIONES DE ANDROID.....</b>	<b>108</b>
<b>ANDROID 0.X, PRIMEROS PASOS.....</b>	<b>111</b>
<b>ANDROID 1.0/1.1 – APPLE PIE – NIVEL DE API 1.....</b>	<b>112</b>
<b>ANDROID 1.1 BANANA BREAD, NIVEL DE API 2 .....</b>	<b>113</b>
<b>ANDROID 1.5 CUPCAKE – NIVEL DE API 3.....</b>	<b>113</b>
<b>ANDROID 1.6 DONUT – NIVEL DE API 4.....</b>	<b>114</b>
<b>ANDROID 2.0/2.1 ECLAIR – NIVEL DE API 5 y 7.....</b>	<b>115</b>
<b>ANDROID 2.2 FROYO – NIVEL DE API 8.....</b>	<b>116</b>
<b>ANDROID 2.3 GINGERBREAD, NIVEL DE API 9.....</b>	<b>118</b>
<b>ANDROID 3.0 HONEYCOMB, NIVEL DE API 11.....</b>	<b>120</b>
<b>ANDROID 4.0 ICE CREAM SANDWICH, NIVEL DE API 14.....</b>	<b>121</b>
<b>ANDROID 4.1 – 4.3 JELLY BEAN – NIVEL DE API 16-17.....</b>	<b>124</b>
<b>ANDROID 4.4 – KITKAT – NIVEL DE API 20.....</b>	<b>124</b>
<b>ANDROID 5.0 – LOLLIPOP .....</b>	<b>125</b>
<b>ANEXO II. ESTILO DE CÓDIGO Y NORMAS MOODLE.....</b>	<b>129</b>
<b>ESTILO DE CÓDIGO.....</b>	<b>131</b>
<b>Reglas generales .....</b>	<b>131</b>
<b>Estilo del código.....</b>	<b>132</b>
<b>Reglas en la gestión de la base de datos.....</b>	<b>133</b>
<b>Normas de seguridad.....</b>	<b>135</b>

# ÍNDICE DE FIGURAS

Figura 1 - Arquitectura de Android.....	12
Figura 2 - Arquitectura de iOS.....	13
Figura 3 - Opiniones Moodle Mobile.....	36
Figura 4 - Jerarquía de Vistas en Android.....	52
Figura 5 - Campos de Mdl_forum.....	54
Figura 6 - Campos de Mdl_forum_discussion.....	55
Figura 7 - Campos de Mdl_forum_posts.....	56
Figura 8 - Campos de Mdl_quiz.....	56
Figura 9 - Campos de Mdl_quiz_slots.....	56
Figura 10 - Campos de Mdl_quiz_attempts.....	57
Figura 11 - Campos de Mdl_question.....	58
Figura 12 - Campos de Mdl_question_answers.....	58
Figura 13 - Campos de Mdl_qtype_multichoice_options.....	59
Figura 14 - Campos de Mdl_question_attempts.....	59
Figura 15 - Campos de Mdl_question_attempt_steps.....	60
Figura 16 - Campos de Mdl_question_attempt_step_data.....	60
Figura 17 - Campos de Mdl_question_usages.....	61
Figura 18 - Campos de Mdl_context.....	61
Figura 19 - Relación Base de Datos Sección Cuestionarios.....	62
Figura 20 - Tabla de la base de datos interna. ....	63
Figura 21 - Relación Base de Datos Sección Foros.....	63
Figura 22 - Objeto JSON.....	66
Figura 23- Array JSON.....	66
Figura 24 - Valor JSON.....	67
Figura 25 - Diagrama Foros.....	69
Figura 26 - Diagrama Envío De Mensajes de discusión.....	70
Figura 27 - Diagrama Creación de Nueva Discusión.....	71
Figura 28 - Diagrama Cuestionarios.....	72
Figura 29- Diagrama Generación Intento.....	74
Figura 30 - Diagrama Salir Intento.....	74
Figura 31 - Diagrama Enviar y Terminar Intento.....	75
Figura 32- Pantalla Nuevo Usuario.....	80
Figura 33 - Pantalla de Inicio.....	81
Figura 34 - Pantalla Menú Principal.....	81
Figura 35 - Pantalla Menú Mensajes.....	82
Figura 36 - Pantalla Mensajes No Leídos.....	83
Figura 37 - Pantalla Mensajes No leídos vacía.....	83

<b>Figura 38 - Pantalla Mensajes Leídos.....</b>	<b>84</b>
<b>Figura 39 - Pantalla Responder Mensaje.....</b>	<b>84</b>
<b>Figura 40 - Pantalla Selección destinatario.....</b>	<b>84</b>
<b>Figura 41 - Pantalla Escribir Mensaje.....</b>	<b>84</b>
<b>Figura 42 - Pantalla Escribiendo Mensaje .....</b>	<b>85</b>
<b>Figura 43 - Pantalla Mensaje Enviado.....</b>	<b>85</b>
<b>Figura 44 - Pantalla Mensajes Enviados.....</b>	<b>85</b>
<b>Figura 45 - Pantalla Asignaturas.....</b>	<b>86</b>
<b>Figura 46 - Pantalla Secciones.....</b>	<b>87</b>
<b>Figura 47- Pantalla Menú Opciones.....</b>	<b>88</b>
<b>Figura 48- Pantalla Foros.....</b>	<b>89</b>
<b>Figura 49- Pantalla Discusiones.....</b>	<b>90</b>
<b>Figura 50 - Pantalla Nueva Discusión.....</b>	<b>90</b>
<b>Figura 51 - Pantalla Mensajes de Discusión.....</b>	<b>91</b>
<b>Figura 52 -Pantalla Escribiendo respuesta.....</b>	<b>92</b>
<b>Figura 53 - Pantalla Respuesta a un Mensaje.....</b>	<b>92</b>
<b>Figura 54 - Pantalla Cuestionarios.....</b>	<b>93</b>
<b>Figura 55 - Pantalla de Intentos.....</b>	<b>94</b>
<b>Figura 56 - Pantalla Preguntas de Cuestionario.....</b>	<b>95</b>
<b>Figura 57- Pantalla Verdadero o falso.....</b>	<b>96</b>
<b>Figura 58 - Pantalla con Respuestas Simple.....</b>	<b>96</b>
<b>Figura 59- Pantalla con Respuesta múltiple.....</b>	<b>96</b>
<b>Figura 60 - Pantalla Respuesta corta.....</b>	<b>96</b>
<b>Figura 61 - Pantalla Numérica.....</b>	<b>97</b>
<b>Figura 62 - Pantalla Al Salir del Intento.....</b>	<b>97</b>
<b>Figura 63 - Pantalla Tras enviar y Terminar.....</b>	<b>97</b>
<b>Figura 64 - Pantalla Recursos.....</b>	<b>98</b>
<b>Figura 65 - Pantalla Proceso de Descarga.....</b>	<b>98</b>
<b>Figura 66 - Pantalla Recurso Descargado.....</b>	<b>98</b>
<b>Figura 67- Cuota de Mercado Versiones Android.....</b>	<b>110</b>
<b>Figura 69- Interfaz Android Básica.....</b>	<b>112</b>
<b>Figura 70- Interfaz Android ApplePie.....</b>	<b>113</b>
<b>Figura 71- Comparación Cupcake.....</b>	<b>114</b>
<b>Figura 72 - Comparación Donut.....</b>	<b>115</b>
<b>Figura 73 - Comparación Nexus Iphone.....</b>	<b>116</b>
<b>Figura 74- Comparación Eclair.....</b>	<b>117</b>
<b>Figura 75- Comparación Froyo.....</b>	<b>118</b>
<b>Figura 76 - Diseño Nexus.....</b>	<b>119</b>
<b>Figura 77 - Comparación Gingerbread.....</b>	<b>120</b>

<b>Figura 78 - Interfaz Tablet.....</b>	<b>121</b>
<b>Figura 79 - Interfaz Nexus IV.....</b>	<b>123</b>
<b>Figura 80- Comparación Ice Cream Sandwich.....</b>	<b>123</b>
<b>Figura 81 - Interfaz Nexus V.....</b>	<b>125</b>

## ÍNDICE DE TABLAS

<b>Tabla 1 - Distribución del mercado de sistemas operativos móviles años 2013 y 2014.....</b>	<b>9</b>
<b>Tabla 2 - Cuota de Mercado Versiones Android .....</b>	<b>110</b>

# **CAPÍTULO I. INTRODUCCIÓN**



## 1. INTRODUCCIÓN GENERAL

En la actualidad han tomado gran importancia en nuestra sociedad las Tecnologías de la Información y la Comunicación (TIC). Se trata de las herramientas y programas que administran y comparten información mediante soporte tecnológico. Las TIC forman ya parte de la mayoría de sectores: educación, robótica, administración pública, empleo y empresas, salud...

Las TIC, respecto al sector la educación han ofrecido grandes beneficios y avances contribuyendo de manera significativa a proporcionar una serie de herramientas que aporten a la enseñanza mayor calidad y eficiencia. La aplicación de las TIC al sistema educativo es conocido como *e-Learning*, teniendo como objetivo principal permitir la formación de los alumnos más allá de las horas lectivas, haciendo posible la comunicación o bien entre alumnos o bien entre alumno y docente, mediante foros de discusión o mensajería instantánea, también posibilitando la evaluación de conocimientos del alumnado y el acceso a recursos académicos en cualquier momento.

Uno de los pilares esenciales de las sociedades contemporáneas es la educación, por lo que toda actividad desarrollada con la finalidad de una mayor eficiencia y calidad en este sector, será considerada como un aporte positivo. Con el *e-Learning* lo que se pretende es tener una herramienta más para el proceso educativo. Muchas veces se asocia a la educación a distancia y esto es un error, ya que como mejor desempeño tiene es mezclado con las técnicas de enseñanza tradicionales suponiendo un apoyo y no un sustituto de las mismas.

Lo que se busca por un lado es elaborar mecanismos y herramientas virtuales que permitan apoyar los cursos que actualmente se están dictando, y por otro lado mejorar la comunicación docente-alumno y alumno-alumno, con el fin de desarrollar parte de las enseñanzas desde el lugar de estudio o el hogar de manera que se optimice el tiempo, logrando mayor eficacia durante las reuniones presenciales. Una herramienta consecuencia de lo explicado anteriormente es la plataforma *Moodle*.

*Moodle* es un software diseñado para ayudar a los docentes a divulgar conocimientos en línea. Una de las características principales de esta plataforma es la de estar desarrollada en base a la pedagogía social constructivista, donde la comunicación tiene un espacio relevante en el camino de la construcción del conocimiento, siendo el objetivo final crear una experiencia de aprendizaje enriquecedora.

El objetivo principal del *e-Learning* como ya se ha comentado es permitir la comunicación y formación de los alumnos más allá de las horas lectivas, como por ejemplo permitiendo el acceso a foros de discusión o la evaluación de conocimientos a través de un ordenador con internet. Aquí es donde entra en juego la idea principal en torno a la cual está desarrollado este trabajo, esta idea no es otra que la de fomentar el aprendizaje a través de un dispositivo móvil (*Smartphones, Tablets...*), evitando la necesidad de disponer de un ordenador para poder aprovechar las ventajas del *e-Learning*, en nuestro caso poder utilizar una serie de

prestaciones de la plataforma *Moodle*, las cuales serán explicadas de manera detallada en el presente documento.

El aprendizaje a través de un dispositivo móvil se conoce como *m-Learning*. El objetivo es que el alumno pueda acceder a las funcionalidades y contenidos que ofrece una plataforma virtual de apoyo educativo aparte de en cualquier momento, también en cualquier lugar sin la necesidad de un ordenador.

## 2. MOTIVACIÓN

En la sociedad actual se puede observar como el uso de los dispositivos móviles está completamente instaurado en nuestra sociedad, concretamente el uso de *smarthphones* entre la población joven es algo totalmente extendido.

De manera sencilla, echando un vistazo a plataformas de distribución de aplicaciones móviles como *Play Store* y *App Store* se puede comprobar cómo muchas empresas que utilizan un software online de cara al público, tienen sus versiones móviles del mismo, ya que son conscientes de la importancia de las aplicaciones móviles y el uso que se hace de ellas, cada vez más extendido. Gestiones que antes se hacían desde un ordenador a día de hoy se pueden hacer desde un dispositivo móvil de manera rápida y sencilla.

La motivación principal que tiene este proyecto, es la de dar respuesta a una necesidad promovida por lo citado en el párrafo anterior. Esta no es otra que la de crear una aplicación móvil de acceso a las prestaciones de la plataforma *Moodle*.

El uso de *smartphones* extendido entre los jóvenes sumado a que *Moodle* es un software muy utilizado dentro de la comunidad educativa lleva a pensar que el desarrollo de una aplicación móvil que permita disponer de *Moodle* en cualquier momento puede resultar muy beneficioso de cara al alumnado, permitiendo disfrutar de las ventajas de *Moodle* únicamente con un dispositivo móvil, sin la necesidad de un ordenador.

Esta idea se empezó a desarrollar en Septiembre de 2012 por Fernando Herrero Herrero, como parte de su proyecto fin de carrera como ingeniero de telecomunicaciones en la universidad de Valladolid. En ese momento no existía ninguna aplicación que implementará *Moodle* como aplicación móvil (Herrero Herrero, 2013).

Actualmente en *Play Store* se pueden encontrar varias aplicaciones *Moodle* de uso privado como la disponible para la Liverpool Hope University, dicha aplicación no es de uso libre y no se puede implementar en cualquier *Moodle*, es de uso exclusivo de dicha universidad.

De igual manera se puede encontrar un par de aplicaciones *Android*, que intentan satisfacer esta necesidad, intentando ser compatible para cualquier *Moodle*. Una de ellas se encuentra obsoleta (*Mdroid Legacy*), y luego se encuentra *Moodle Mobile*, la cual atendiendo a las valoraciones y comentarios de los usuarios se cae en la cuenta de que no es una aplicación

completa ni resolutive siendo varias personas las que coinciden en que es más un acceso directo que una aplicación, ya que para muchas funcionalidades se redirige al usuario a la web de *Moodle*, sin dar la posibilidad de realizar una determinada gestión desde la misma aplicación.

Esto es algo que en este proyecto se evitará, todas las funcionalidades implementadas se desarrollarán exclusivamente desde la aplicación móvil, en ningún caso se accederá a la versión web de *Moodle* desde el navegador de nuestro *smartphone* o *tablet*.

Lo que se busca con la aplicación móvil es evitar cargar la versión de PC en un dispositivo móvil, optimizando así el tiempo de carga

### 3. OBJETIVOS

Este proyecto tiene como objetivo principal el realizar una aplicación móvil que permita a los usuarios de la plataforma *Moodle* acceder y participar en los foros de discusión tanto proponiendo nuevas discusiones como aportando respuestas o preguntas a una discusión determinada.

Por otro lado se implementará el servicio que permitirá al alumnado realizar cuestionarios muy diversos propuestos por el docente mediante la actividad de cuestionarios que proporciona *Moodle*.

Esta aplicación parte de un primer proyecto fin de carrera ya desarrollado el cual consistía en una aplicación móvil que permitía el acceso a una serie de funcionalidades de *Moodle* (Herrero Herrero, 2013). Dicha aplicación era útil para la versión 1.9 y anteriores, pero se encuentra obsoleta para la última versión estable (2.7.1).

La aplicación anteriormente mencionada fue actualizada para la versión (2.7.1) incluyendo el servicio de mensajería y la opción de registro de nuevos usuarios (Antolín Blanco, 2015).

En esta aplicación se ha desarrollado los servicios de foros de discusión y el acceso y realización de cuestionarios, y por supuesto la integración con la aplicación enunciada en el párrafo anterior.

Para ello y de manera resumida, la aplicación se desarrollará de la siguiente manera:

- En primer lugar aclarar que tras el proceso de integración la aplicación final contiene el proceso de registro de nuevos usuarios y el proceso de autenticación que tras ser efectivo se muestra un menú con las opciones de mensajería instantánea y asignaturas de las que el usuario previamente autenticado esté matriculado.
- Tras acceder a la opción de asignaturas, se accede los temas correspondientes (corresponden a los recursos denominados etiquetas en *Moodle* con las cuáles el

administrador , normalmente el docente, debe nombrar los temas) y de estos al menú en que ya se puede acceder a foros, cuestionarios y recursos.

- Accediendo a “foros” se muestran en formato de lista los foros de discusión del tema, posteriormente seleccionado, seleccionando un foro se muestran sus discusiones asociadas y la opción de crear una nueva, seleccionando una de las discusiones se muestran la lista de mensajes, y de forma semejante a las discusiones en este paso se puede contestar un mensaje o enviar un mensaje nuevo a la discusión.
- En el menú mencionado en el punto dos ahora se va a acceder a “cuestionarios”, tras esta acción se muestran los cuestionarios disponibles, y accediendo a uno en concreto, se muestran en formato lista las preguntas, seleccionando una de estas se puede contestar, pasar a la siguiente o anterior pregunta en caso de que hubiese, salir del intento quedando este en estado “ en curso” o terminar y enviar las respuestas para obtener la calificación final del cuestionario y cerrar el intento.

## 4. INTRODUCCIÓN AL DOCUMENTO

Este documento se estructura en una serie de seis capítulos como se explica a continuación:

- El primer capítulo sirve de introducción al proyecto y al presente documento. Explicando los objetivos y la motivación del mismo.
- En el segundo capítulo se realizará un análisis de las tecnologías existentes para la realización del proyecto y se expondrán los motivos de elección de cada una.
- El tercer capítulo trata acerca de las características de *Moodle*, plataforma sobre la que gira el desarrollo de este proyecto.
- En el cuarto capítulo se hablará de los servicios Web, gracias a los cuales se implementan las funcionalidades que se abordan en este proyecto.
- El quinto capítulo recoge el desarrollo del proyecto, tanto desde el lado del cliente como del servidor y de la comunicación de los mismos.
- El sexto capítulo es un manual de usuario mostrando la interfaz gráfica.
- El séptimo y último capítulo recogerá las conclusiones y las posibles líneas futuras a seguir en una próxima evolución del proyecto.
- Finalmente se presentará la bibliografía empleada en el desarrollo del proyecto y se adjuntarán una serie de anexos complementarios a la comprensión del mismo.



## **CAPÍTULO II.**

# **JUSTIFICACIÓN DE LAS TECNOLOGÍAS EMPLEADAS**



## 1. INTRODUCCIÓN

En este capítulo se hará un repaso de las tecnologías disponibles para realizar la aplicación móvil planteada, y justificando el por qué se ha seleccionado una u otra.

Por un lado se expondrán los diferentes sistemas operativos móviles, y por otro se repasarán las diferentes plataformas de *e-Learning* disponibles.

## 2. SISTEMAS OPERATIVOS MÓVILES

Un sistema operativo (SO) es el software de una computadora que provee una interfaz entre el resto de programas del ordenador, los dispositivos hardware y el usuario.

Las funciones básicas del Sistema Operativo son administrar los recursos de la máquina, coordinar el hardware y organizar archivos y directorios en dispositivos de almacenamiento.

Actualmente los dispositivos móviles, con el fin de mejorar la interacción con el usuario, y las nuevas necesidades requeridas por los mismos, van equipados con un sistema operativo, con el mismo objetivo que el descrito para los PCs.

En el mercado mundial de los sistemas operativos móviles se encuentran varios tipos. A la cabeza de todos se encuentra *Android*, el sistema operativo de Google (ver Tabla 1).

Su competencia principal es *iOS* el sistema operativo de Apple, aunque lo sigue a larga distancia. Hay otros sistemas con mucha menor aceptación en el mercado como lo son *Windows Phone*, *Firefox OS*, *BlackBerry*, *Ubuntu Touch*, *Tizen*, *WebOS...* (Amate, 2014)

Este apartado se centrará en dar una breve explicación de todos los anteriores, focalizándose principalmente en las características de Android e iOS ya que son los que ocupan entre ambos el 96.5% de la cuota del mercado. Este apartado finalizará con una justificación del sistema operativo elegido para el desarrollo de esta aplicación (Marín, 2014).

Global Smartphone Operating System Shipments (Millions of Units)	Q2 '13	Q2 '14
Android	186.8	249.6
Apple iOS	31.2	35.2
Microsoft	8.9	8.0
BlackBerry	5.7	1.9
Others	0.5	0.5
Total	233.0	295.2

Global Smartphone Operating System Marketshare %	Q2 '13	Q2 '14
Android	80.2%	84.6%
Apple iOS	13.4%	11.9%
Microsoft	3.8%	2.7%
BlackBerry	2.4%	0.6%
Others	0.2%	0.2%
Total	100.0%	100.0%

Total Growth Year-over-Year %	48.9%	26.7%
-------------------------------	-------	-------

Source: Strategy Analytics

**Tabla 1 - Distribución del mercado  
de sistemas operativos móviles años 2013 y 2014**

## 2.1 ANDROID

El sistema operativo número uno en cuanto a popularidad. Con una cuota de mercado cercana al 85% el sistema operativo de *Google* se caracteriza por ser abierto y disponible para cualquier fabricante interesando en utilizarlo para sus dispositivos móviles. Esta característica hace que su cuota de mercado sea muy amplia, ya que no necesita una marca de dispositivos concreta para poder ser implementado (Marín, 2014).

El sistema operativo está compuesto por 12 millones de líneas de código, incluyendo 3 millones de líneas de *XML*, 2,8 millones de líneas de lenguaje *C*, 2,1 millones de líneas de *Java* y 1,75 millones de líneas de *C++*. Se puede observar la arquitectura de este sistema operativo de manera gráfica en la Figura 1:



Figura 1 - Arquitectura de Android

- **Aplicaciones:** Este nivel contiene, tanto las incluidas por defecto de *Android* como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan los servicios, las API y librerías de los niveles anteriores.
- **Framework de Aplicaciones:** Representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para *Android*, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo *framework* representado por este nivel.
- **Librerías:** La siguiente capa se corresponde con las librerías utilizadas por *Android*. Éstas han sido escritas utilizando *C/C++* y proporcionan a *Android* la mayor parte de sus capacidades más características. Junto al núcleo basado en *Linux*, estas librerías constituyen el corazón de *Android*.

- Tiempo de ejecución de *Android*: Al mismo nivel que las librerías de *Android* se sitúa el entorno de ejecución. Éste lo constituyen las *Core Libraries*, que son librerías con multitud de clases Java y la máquina virtual Dalvik.
- Núcleo Linux: *Android* utiliza el núcleo de Linux 2.6 como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes. Siempre que un fabricante incluye un nuevo elemento de hardware, lo primero que se debe realizar para que pueda ser utilizado desde *Android* es crear las librerías de control o drivers necesarios dentro de este *kernel* de Linux incorporado en el propio *Android*.

## 2.2 iOS

Este sistema operativo supone el máximo competidor para *Android*, tal y como se advertía en la Figura 1. Este sistema operativo se encuentra únicamente en los dispositivos *Apple*, ya que son los únicos que lo pueden implementar. Esto limita de manera notable su cuota de mercado.

*Apple* diseñó este sistema para *iPhone* (teléfono de la marca *Apple*), pero más tarde también lo han usado en el *iPod Touch* y en el *iPad*. Es un sistema derivado de *Mac OS X*, el cual está basado a su vez en *Darwin BSD*. Es un sistema basado en capas, como se observa en la Figura 3.

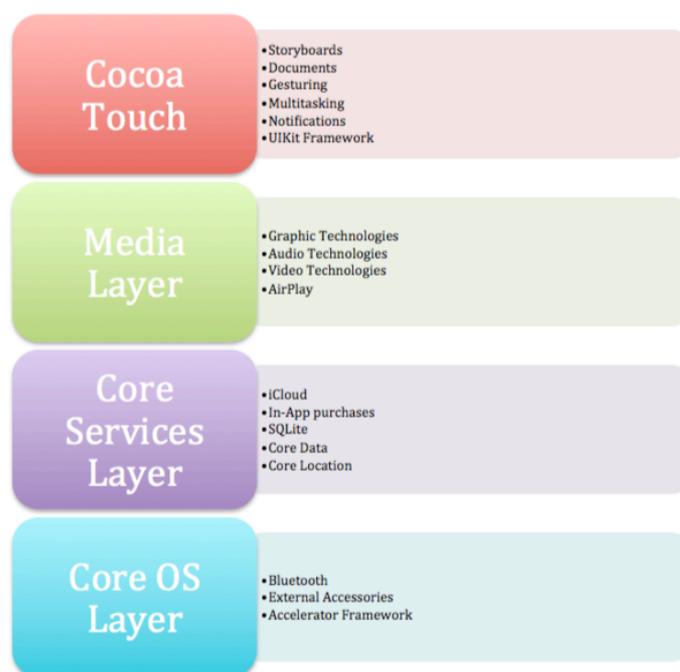


Figura 2 – Arquitectura de iOS

- Cocoa Touch: Se compone de un conjunto de *frameworks* para desarrollar aplicaciones. El lenguaje utilizado para programar esta librería, y por tanto las aplicaciones de *iOS*, es *Objective-C*.
- Media: Esta capa se encarga de los servicios multimedia que pueden usar los usuarios en sus aplicaciones.
- Core Service: provee una abstracción sobre los servicios que proporciona el sistema.
- Core OS: Es la capa base “Core OS” supone el núcleo del sistema operativo, es la que se encarga de la administración de memoria, el sistema de ficheros, las comunicaciones, entre otras tareas. Esta capa interactúa directamente con el hardware.

## 2.3 OTROS SISTEMAS OPERATIVOS MÓVILES

### 2.3.1 WINDOWS PHONE

Es el sistema operativo móvil desarrollado por *Microsoft*. Esta compañía está realizando un gran esfuerzo financiero para posicionar *Windows Phone* como una tercera opción interesante para los consumidores. Su alianza con *Nokia* y su posterior compra le ha ayudado a darse a conocer mejor e ir arañando cuota de mercado a los dos líderes. Los últimos datos hablan de un 2,5% a nivel mundial.

Con un diseño radicalmente distinto a las dos opciones ya comentadas, *Windows Phone* destaca por su pantalla de inicio personalizable que ofrece las notificaciones de las aplicaciones de una manera sencilla y limpia. Además ofrece una experiencia de usuario muy buena independientemente del tipo y gama de terminal en que se esté usando.

Aunque con menos aplicaciones disponibles que en *Android* e *iOS*, *Windows Phone 8.1*, cuenta ya con más de 300.000 *Apps* en su tienda, además de ofrecer aplicaciones propias de la compañía como *Skype*, *OneDrive* o *Xbox Live*.

### 2.3.2 FIREFOX OS

Un sistema operativo basado en *HTML5* con núcleo *Linux*, de código abierto. Desarrollado por *Mozilla Corporation* con apoyo de empresas como Telefónica. El sistema operativo está basado en *Linux* y usa la tecnología de *Mozilla*, *Gecko*. Se basa en estándares abiertos como por ejemplo *HML5*, *CSS3* y *JavaScript*.

Pensado para ser un sistema operativo realmente abierto, a diferencia de *Android*, donde *Google* controla ciertos aspectos del sistema. Esta característica, permite a *Firefox OS* llegar a cubrir el nicho de mercado de la gama baja con mayor facilidad que *Android*. El anuncio hecho en febrero de este año de lanzar un *smartphone* por 25 euros va completamente en esa línea.

Entre las interesantes características de este sistema operativo abierto están las aplicaciones web y pueden ser de dos tipos diferentes: aplicaciones de servidor o empaquetadas. A diferencia de los SO ya comentados, en este caso, las *Apps* de servidor, corren vía web, es decir son páginas webs con la apariencia de aplicaciones y sin conexión a internet no es posible acceder a estas. Las aplicaciones empaquetadas necesitan la descarga de un paquete comprimido y se cargan desde la fuente local cada vez que se accede a la aplicación.

### 2.3.3 BLACKBERRY

*Blackberry* anteriormente conocida como *RIM* no está pasando por sus mejores momentos. Al igual que le pasó a *Nokia*, el cambio de paradigma en los *smartphones* no le fue beneficioso. Acostumbrado a ofrecer terminales con teclado físico, el paso a las pantallas táctiles supuso un problema.

Sin embargo, los esfuerzos realizados por la compañía canadiense para recuperar el terreno perdido fueron grandes y en el año 2012 lanzaron su órdago con un renovado sistema operativo el *Blackberry 10*. Aun así, los últimos estudios sobre cuota de mercado lo dejan en tan solo un 0,5% mundial.

*Blackberry 10* tiene una interfaz más fluida, un teclado inteligente y táctil más depurado y otra serie de opciones que lo acercan a las de la competencia. Al igual que con *iOS*, el SO es software propietario y solamente los teléfonos de la compañía llevan su sistema instalado.

### 2.3.4 UBUNTU TOUCH

Otro sistema operativo basado en Linux pero en esta ocasión bajo la famosa firma *Ubuntu*. Presentado en el 2013, se trata de un proyecto de Canonical.

*Ubuntu Touch* utiliza las mismas tecnologías de la versión de escritorio, por lo que ambas comparten *Apps* sin problemas de compatibilidad. Dispone también de algunas de las aplicaciones más populares como *Facebook* y *Youtube*.

### 2.3.5 TIZEN

Sistema operativo móvil, también basado en *Linux*, patrocinado por *Linux Foundation* y Fundación LiMo. Se ha desarrollado a partir de la plataforma *Linux* de *Samsung*. Aunque en un principio fue presentado como un SO de código abierto, *Tizen 2* funciona con un sistema de licencias no abiertas. El SDK completo fue publicado bajo licencia de *Samsung* de código no abierto.

Aunque pueda parecer que *Tizen* forma parte de la estrategia de *Samsung* a largo plazo, su apuesta errática por este sistema operativo hace que no se sepa muy bien qué pasará con él. De momento algunos de sus dispositivos ya lo incorporan como el caso del famoso *smartwatch Samsung Gear S*.

## 2.4 ELECCIÓN

Como ya se explicó en la introducción, la diferencia del uso en terminales móviles de *Android* e *iOS* es infinitamente superior a la de sus competidores. En la introducción de este documento comentamos que una de sus motivaciones principales de esta aplicación era la de permitir el uso de la misma al mayor número de alumnos y docentes. Parece lógico pensar que tendrá que ser desarrollada para un sistema operativo que abarque la mayor cuota de mercado, por lo tanto parece claro que nuestra elección para desarrollar esta aplicación *Moodle* será *Android*.

La motivación principal que hace que decanta el desarrollo en de esta aplicación con *Android* es la cuota de mercado, para poder abarcar el mayor número de usuarios. Pero hay otros motivos que hacen que la balanza caiga por el lado de *Android*, en detrimento de *iOS* su inmediato rival:

- Para desarrollar en *iOS* los desarrolladores deben abonar una cuota anual de 99\$[], algo que en *Android* resulta mucho más barato, ya que para subir aplicaciones a *Google Play*, únicamente habrá que hacer un único pago de 25\$ para obtener la licencia (Picurelli, 2014).
- La segunda razón es que para desarrollar en *iOS* es necesario disponer de un terminal *Mac* (PCs de la marca *Apple*) y en consecuencia disponer de un terminal móvil *Apple*, ambos de precio elevado. Mientras que para desarrollar *Android* únicamente debemos disponer de un PC normal y un móvil *Android*, los cuales se pueden encontrar a un precio mucho más asequible en el mercado.

## 3.INTRODUCCIÓN PLATAFORMAS E-LEARNING

Como ya se expuso en la introducción de este documento, este proyecto gira en torno a la idea del *e-Learning*. Este nuevo concepto educativo es una revolucionaria modalidad de capacitación que posibilitó Internet, y que hoy se posiciona como la forma de capacitación predominante en el futuro. Este sistema ha transformado la educación, abriendo puertas al aprendizaje individual y organizacional.

La plataforma *e-Learning* por excelencia es *Moodle*, es la más desarrollada y utilizada, pero existen otras alternativas que permiten la educación y capacitación a través de Internet. En este apartado repasaremos brevemente las características básicas de *Moodle* por un lado y las del resto de plataformas por otro.

### 3.1 MOODLE

Una de sus principales características es que es un software Libre. Esto implica que tiene derechos de autor, pero el usuario puede modificar y usar *Moodle* de manera libre siempre que se acepte proporcionar el código fuente a terceros, no modificar en ningún caso la licencia original y/o los derechos de autor, y se deberá aplicar esta licencia a cualquier trabajo derivado de él.

*Moodle*, es adaptable a las necesidades de cada institución o empresa. También es fácilmente integrable a infraestructuras existentes y tiene una enorme capacidad de crecimiento.

Una de las diferencias más importantes es la disponibilidad de opciones. *Moodle* tiene una enorme variedad de herramientas para el armado de cursos, y todas están disponibles en forma libre. No hay una "Licencia Básica" con opciones recortadas y una "Licencia Plus" que incluya todos los módulos, de hecho no hay licencia: *Moodle* incluye todo desde el momento en que lo instalamos. Su arquitectura y herramientas fueron diseñadas para clases en línea, así como también para complementar el aprendizaje presencial.

La instalación requiere una plataforma que soporte *PHP* y la disponibilidad de una base de datos. *Moodle* tiene una capa de abstracción de bases de datos por lo que soporta los principales sistemas gestores de bases de datos. Aunque se recomienda ser implementado con *MySQL*.

Se ha puesto énfasis en una seguridad sólida en toda la plataforma. Todos los formularios son revisados, las cookies cifradas, etc. La mayoría de las áreas de introducción de texto (materiales, mensajes de los foros, entradas de los diarios, etc.) pueden ser editadas usando el editor *HTML*, tan sencillo como cualquier editor de texto.

### 3.2 OTRAS PLATAFORMAS DISPONIBLES

Con mucho menos nivel de desarrollo y uso, en el campo del *e-Learning* nos podemos encontrar con otra serie de plataformas, aquí mostramos una breve lista de las mismas con sus principales características (Fontela Sánchez, 2013):

- *ATutor*: Es otra plataforma de *e-Learning* o *LMS* desarrollada en *PHP* con base de datos *MySQL* y completamente *opensource* (de código abierto). Es una solución bastante efectiva y completamente compatible con paquetes *SCORM* o *IMS*.
- *Claroline*: Otra alternativa desarrollada en *PHP* con base de datos *MySQL*, completamente *opensource* y tan efectiva como *ATutor* y *Moodle*. Dispone de bastantes herramientas colaborativas.

- *eFront*: Un *LMS* que está disponible en 2 versiones, una de pago y otra gratuita. También está desarrollada en *PHP* con base de datos *MySQL*. También ofrece un servicio autohosteado por los desarrolladores.
- *ILIAS*: Este es otro sistema para gestionar cursos y grupos. Permite crear todo tipo de recursos educativos y algunos módulos para realizar funciones avanzadas como integración *WebDav* y *LDAP*.
- *Dokeos*: Este es otro software parecido a *Moodle*, está disponible en dos versiones, una gratuita y otra de pago, pero la versión gratuita tiene algunas limitaciones importantes. Esta desarrollado en *PHP* con base de datos *MySQL*. Dispone de unas capacidades impresionantes, hasta en la versión gratuita.
- *Sakai*: Esta es una herramienta colaborativa con posibilidades de ser utilizada como plataforma *e-Learning* para cursos online. Es un sistema desarrollado para funcionar sobre *Apache Tomcat* y con muy buen rendimiento.

Ninguna de estas plataformas de *e-Learning* está a la altura de *Moodle*, ya que todos los sistemas *e-Learning* anteriormente citados tienen las funcionalidades menos desarrolladas y con menos características.

### 3.3 ELECCIÓN DE LA PLATAFORMA E-LEARNING

*Moodle* es la plataforma *e-Learning* más desarrollada. Se seleccionará esta plataforma para implementarla en versión móvil atendiendo a las tres razones siguientes:

- Es la plataforma que utilizada por defecto en la Universidad de Valladolid, entidad para la que se desarrolla este proyecto.
- Es la plataforma *e-Learning* más completa.
- Disponemos de mucha información y documentación acerca de la misma.

*Moodle* está implementado con *PHP-MySQL* por lo que la parte del servidor de nuestro proyecto, será desarrollada de igual manera.



# **CAPÍTULO III.**

## **MOODLE.**



## 1. INTRODUCCIÓN

*Moodle* es una aplicación web de tipo Entorno de Aprendizaje Virtual (*Virtual Learning Enviroment, VLE*), un Sistema de Gestión de Cursos de Código Abierto (*Open Source Course Management System, CMS*), de distribución libre, también conocido como Sistema de Gestión del Aprendizaje Virtual (*Virtual Learning Enviroment, VLE*) que ayuda a los educadores a crear comunidades de aprendizaje en línea. Este tipo de plataformas tecnológicas también se conoce como LCMS (*Learning Content Management System*).

El diseño y el desarrollo de *Moodle* se basan en una determinada filosofía del aprendizaje, una forma de pensar que a menudo se denomina "pedagogía constructorista social". Este punto de vista mantiene que la gente construye activamente nuevos conocimientos a medida que interactúa con su entorno (Moodle, 2015).

La palabra *Moodle*, en inglés, es un acrónimo para Entorno de Aprendizaje Dinámico Modular, Orientado a Objetos (*Modular Object-Oriented Dynamic Learning Environment*), lo que resulta fundamentalmente útil para los desarrolladores y teóricos de la educación. También es un verbo anglosajón que describe el proceso ocioso de dar vueltas sobre algo, haciendo las cosas como se vienen a la mente... una actividad amena que muchas veces conllevan al proceso de comprensión y, finalmente, a la creatividad.

*Moodle* es un software que permite crear comunidades educativas basadas en Internet. Constituye una potente herramienta de apoyo al docente que le permite impartir sus conocimientos utilizando las nuevas tecnologías.

De una manera más coloquial, podemos decir que *Moodle* es una aplicación para crear y gestionar plataformas educativas, es decir, espacios donde un centro educativo, institución o empresa, gestiona recursos educativos proporcionados por unos docentes y organiza el acceso a esos recursos por los estudiantes, y además permite la comunicación entre todos los implicados (alumnos y profesores).

*Moodle* tiene una distribución gratuita como software libre (*Open Source*, bajo la licencia pública GNU). Tiene derechos de autor, pero el usuario puede copiar, modificar y usar *Moodle* de manera libre siempre que se acepte proporcionar el código fuente a terceros, no modificar en ningún caso la licencia original y/o los derechos de autor, y se deberá aplicar esta licencia a cualquier trabajo derivado de él.

Por último y para resumir lo anterior diremos que *Moodle* es un sistema para el Manejo del Aprendizaje en línea gratuito, que les permite a los educadores la creación de sus propios sitios web privados, llenos de cursos dinámicos que extienden el aprendizaje, en cualquier momento, en cualquier sitio.

### **3. CARACTERÍSTICAS**

Atendiendo a la documentación oficial de *Moodle* (Moodle, 2015), hablaremos de las características generales de la plataforma, más tarde de las características administrativas y las características para desarrollo y gestión de los cursos.

#### **2.1 CARACTERÍSTICAS GENERALES**

- **Modularidad:** Cada uno de los recursos que componen la plataforma se define como una pieza de software independiente que puede ser modificada o eliminada según las necesidades de la institución de enseñanza que la adopte.
- **Interfaz moderna y fácil de usar:** Diseñada para ser responsiva y accesible.
- **Tablero personalizado:** La página inicial dispone de un “tablero” que permite organizar y mostrar los cursos de múltiples formas y permite ver rápidamente el conjunto de mensajes y tareas actuales.
- **Actividades y herramientas colaborativas:** Se dispone de módulos como los foros, las wikis y los glosarios, los cuáles permiten trabajar y aprender de manera conjunta.
- **Calendario todo-en-uno:** Este calendario te ayuda a mantener al día su calendario académico o el de la compañía, fechas de entrega dentro del curso, reuniones grupales y otros eventos personales.
- **Gestión conveniente de archivos:** Se permite la colocación de archivos desde servicios de almacenamiento en la nube, incluyendo MS Skydrive, Dropbox y Google Drive.
- **Editor de texto simple e intuitivo:** Posibilita dar formato convenientemente al texto y añadir multimedia e imágenes con un editor que funciona con todos los navegadores de Internet.
- **Notificaciones:** Cuando se habilitan, los usuarios pueden recibir alertas automáticas acerca de nuevas tareas y fechas para entregarlas, publicaciones en foros y también de los mensajes privados mandados/recibidos entre ellos.
- **Monitoreo del progreso:** Los docentes y los alumnos pueden monitorear el progreso y el grado de finalización con un conjunto de opciones para monitoreo de actividades individuales o recursos, y también a nivel del curso.

## 2.2 CARACTERÍSTICAS ADMINISTRATIVAS

- Diseño personalizable del sitio: fácilmente permite personalizar un tema de *Moodle* con su logo, esquema de colores y mucho más.
- Autenticación e inscripciones (matriculaciones) masivas seguras: Más de 50 opciones para autenticación e inscripción, para añadir e inscribir usuarios a su sitio y cursos *Moodle*.
- Capacidad multilingüe: Se posibilita que los usuarios vean el contenido del curso y aprendan en su propio idioma, o también se da la opción de configurar el sitio para organizaciones y usuarios multilingües.
- Creación masiva de cursos: Permite añadir cursos en lotes.
- Fácil gestión de permisos y roles de usuario: Se atajan las preocupaciones sobre seguridad al definir roles para especificar y gestionar el acceso de los usuarios.
- Soporta estándares abiertos: Se da la posibilidad de importar y exportar fácilmente cursos IMS-LTI, SCORM y más, hacia y desde *Moodle*.
- Alta interoperabilidad: Está habilitada la posibilidad de integrar libremente aplicaciones externas y contenidos, o cree su propio *plugin* para integraciones personalizadas.
- Gestión simple de *plugins* y complementos: Permite instalar y deshabilitar complementos y *plugins* desde adentro de una sola interfaz administrativa.
- Actualizaciones regulares de seguridad: *Moodle* es actualizado regularmente con los últimos parches de seguridad, para ayudar a asegurar que su sitio *Moodle* sea seguro.
- Reportes y bitácoras detalladas: Permite generar y ver reportes sobre actividad y participación a nivel de curso y de sitio.

## 2.3 CARACTERÍSTICAS PARA EL DESARROLLO Y GESTIÓN DEL CURSO

- Rutas directas de aprendizaje: Se da la posibilidad de diseñar y gestionar cursos para cumplir con diversos requisitos. Las clases pueden ser dirigidas por el instructor, auto-reguladas, mixtas o completamente en-línea.

- **Fomento de la colaboración:** Las características incluidas para la publicación colaborativa fomentan que el alumno se comprometa y realice colaboración impulsada por el contenido.
- **Utilización de recursos externos:** Permite enseñar con materiales e incluir tareas provenientes de otros sitios y conectándolos al libro de calificaciones en *Moodle*.
- **Integración multimedia:** El soporte incluido en *Moodle* para multimedia le permite buscar fácilmente e insertar archivos de audio y video en sus cursos.
- **Gestión de grupo:** se recomienda agrupar los alumnos para compartir los cursos y diferenciar actividades, para favorecer así el trabajo en equipo.
- **Flujo gram de puntuación:** Permite por un lado asignar convenientemente a diferentes personas para que califiquen tareas y gestionar la moderación de calificaciones. Y por otro lado permite controlar cuando se liberan las calificaciones a los alumnos individuales.
- **Calificación en línea:** Se da la posibilidad de revisar con facilidad y proporcionar retroalimentación en-línea, al hacer anotaciones directamente sobre archivos PDF dentro del navegador de Internet.
- **Evaluación propia y por pares:** Actividades incluidas, tales como talleres y encuestas, estimulan a los alumnos para que vean, califiquen y evalúen el trabajo de ellos mismos y el de otros participantes del curso como un grupo.
- **Insignias integradas:** Totalmente compatible con las Insignias Abiertas de *Mozilla* (*Mozilla Open Badges*), motive a los estudiantes y recompense la participación y los logros con insignias personalizadas.
- **Competencias y rúbricas:** Permite seleccionar entre métodos avanzados de calificación para personalizar el libro de calificaciones del curso y de acuerdo a sus criterios de exámenes.
- **Seguridad y privacidad:** Posibilita enseñar y compartir conocimientos y recursos dentro de un espacio privado, al que solamente pueden acceder el docente y su grupo de alumnos.

## 3. ARQUITECTURA

### 3.1 INTRODUCCIÓN

Desde la perspectiva de un administrador de sistemas *Moodle* ha sido diseñado de acuerdo con los siguientes criterios (Moodle, 2015):

***Moodle debe poder ejecutarse en la más amplia posible variedad de plataformas:***

La plataforma de aplicaciones Web que funciona en la mayoría de las plataformas es *PHP* combinada con *MySQL*, y este es el entorno en el que *Moodle* ha sido desarrollado (sobre *Linux*, *Windows*, y *Mac OS X*). *Moodle* también usa la librería *ADODB* para la abstracción de bases de datos, lo que significa que *Moodle* puede usar más de diez marcas diferentes de bases de datos.

***Moodle debe de ser fácil de instalar, aprender y modificar:***

Los primeros prototipos de *Moodle* (1999) se construyeron usando *Zope*, un avanzado servidor de aplicaciones Web orientado a objetos. Desafortunadamente resultó que aunque la tecnología era bastante buena, tenía una curva de aprendizaje muy elevada y no era muy flexible en términos de administración del sistema. El lenguaje *PHP*, por otro lado, es muy fácil de aprender (especialmente si el desarrollador ha realizado algo de programación usando cualquier otro lenguaje de script).

La reutilización del código se archiva en librerías con funciones claramente tituladas y con una disposición de los archivos de script, consistente. *PHP* es también fácil de instalar (existen versiones ejecutables para todas las plataformas) y está ampliamente disponible, pues la mayoría de los servicios de alojamiento lo proporcionan como un estándar.

***Debe ser fácil de actualizar desde una versión a la siguiente:***

*Moodle* sabe cuál es su versión (así como las versiones de todos los módulos) y se ha construido un mecanismo interno para que *Moodle* pueda actualizarse a sí mismo de forma apropiada a las nuevas versiones (por ejemplo, puede renombrar las tablas de las bases de datos o añadir nuevos campos). Usando *CVS* en *Unix*, por ejemplo, uno tan sólo tiene que hacer un "cvs update -d" y luego visitar la página principal del sitio para completar la actualización.

***Debe ser modular para permitir el crecimiento:***

*Moodle* tiene una serie de características modulares, incluyendo temas, actividades, interfaces de idioma, esquemas de base de datos y formatos de cursos. Esto le permite a cualquiera añadir características al código básico principal o incluso distribuirlas por separado. Se abordará este tema de manera profunda en la siguiente sección.

### Debe poder usarse junto a otros sistemas:

Una de las cosas que hace *Moodle* es mantener todos los archivos para un curso en un único directorio en el servidor. Esto podría permitir que el administrador de un sistema proporcione similares formas de acceso a un nivel de archivo para cada profesor, tal como *Appletalk*, *SMB*, *NFS*, *FTP*, *WebDAV* y demás. Los módulos de autenticación le permiten a *Moodle* usar *LDAP*, *IMAP*, *POP3*, *NNTP* y otras bases de datos como fuentes de información de los usuarios.

## 3.2 ESTRUCTURA

El directorio principal *Moodle* del servidor tiene almacenados todos los contenidos de la plataforma. A continuación, se detalla la estructura de este directorio y sus elementos principales (Herrero Herrero, 2013):

- **config.php**: Contiene la configuración fundamental. Este archivo no viene con *Moodle*, se crea durante la instalación.
- **install.php**: El script que se ejecuta para crear el archivo *config.php*. Se lanza cuando se instala el paquete de *Moodle*.
- **version.php**: Define la versión actual del código de *Moodle*.
- **index.php**: La página principal del sitio donde hay que autenticarse.
- **help.php**: Muestra la página de ayuda cuando se pulsa sobre el icono de ayuda.
- **admin/**: Código para administrar todo el servidor.
- **auth/**: Módulos para la autenticación de usuarios.
- **blocks/**: Módulos para los pequeños bloques laterales contenidos en la página principal.
- **backup/**: Código para la generación de copias de seguridad.
- **blog/**: Código para la administración de blogs.
- **calendar/**: Código para manejar y mostrar eventos de calendario.
- **course/**: Código para presentar y gestionar los cursos.

- **enrol/**: Código para los módulos de inscripción.
- **error/**: Código sobre los errores de la plataforma.
- **files/**: Código para presentar y gestionar los archivos cargados.
- **filter/**: Código para gestionar los filtros en los módulos.
- **grade/**: Código para calificar los exámenes y trabajos de los alumnos de los cursos.
- **group/**: Código para establecer grupos de alumnos.
- **install/**: Código para la instalación de *Moodle*.
- **iplookup/**: Muestra información sobre la dirección IP.
- **lang/**: Textos en diferentes idiomas, un directorio por idioma.
- **lib/**: Librerías del código fundamental de *Moodle*.
- **login/**: Código para manejar las entradas y creación de cuentas.
- **message/**: Código para gestionar los distintos mensajes que van apareciendo durante todo el uso de *Moodle*.
- **mod/**: Todos los módulos de los cursos de *Moodle*.
- **my/**: Configuración de la página personal de *Moodle*.
- **pix/**: Gráficos genéricos del sitio (íconos).
- **question/**: Código para gestionar el módulo de preguntas.
- **rss/**: Configuración de los canales RSS.
- **search/**: Configuración de las búsquedas internas.
- **tag/**: Etiquetas asignadas a los blog de usuarios y a sus intereses.
- **theme/**: Paquetes de temas para cambiar la apariencia del sitio.

- **user/**: Código para mostrar y gestionar los usuarios.

### 3.3 MÓDULOS DE ACTIVIDADES

Estos son los módulos más importantes, y se encuentran en el directorio mod. Por defecto hay siete módulos: Tarea, Consulta, Foro, Glosario, Cuestionario, Recurso, y Encuesta. Cada módulo está en un subdirectorio separado y consiste en los siguientes elementos obligatorios (más los scripts extra que son únicos para cada módulo):

- **mod\_form.php**: un formulario para establecer o actualizar una instancia de este módulo.
- **version.php**: define alguna meta-información y proporciona código de actualización
- **icon.gif**: se trata de un icono de 16x16 para el módulo
- **db/install.xml**: define la estructura de las tablas para todos los tipos de bases de datos. Se utiliza cuando se instala el módulo.
- **db/upgrade.php**: define los cambios en la estructura de las tablas. Se utiliza cuando se actualiza el módulo
- **db/access.php**: define los permisos.
- **index.php**: es una página para presentar la lista de todas las instancias en un curso
- **view.php**: se trata de una página para ver una instancia en particular
- **lib.php**: cualquiera/todas las funciones definidas para el módulo deben estar aquí. Si el módulo se llama "mod", entonces las funciones requeridas incluyen:
  - **mod\_add\_instance()** - código para añadir una nueva instancia de chisme
  - **mod\_update\_instance()** - código para actualizar una instancia existente
  - **mod\_delete\_instance()** - código para borrar una instancia
  - **mod\_user\_complete()** - dada una instancia, imprime detalles sobre la contribución de un usuario
  - **mod\_user\_outline()** - dada una instancia, devuelve un resumen de una contribución de un usuario

Finalmente, cada módulo tendrá algunos archivos de idioma que contienen cadenas para ese módulo.

### **3.4 TEMAS**

Los temas definen la apariencia de un sitio. Con la distribución básica se proporciona una serie de temas simples, pero el usuario puede querer crear su propio tema, con sus propios colores, logo, estilos y gráficos a través del uso de XHTML y CSS.

- Los temas pueden ser configurados a nivel de sitio, curso y/o usuario.
- Cada página se maneja en forma individual por CSS.
- Las clases CSS que nombra el sistema utiliza inglés simple, es consistente y fácilmente comprensible
- Los nuevos módulos pueden decir a Moodle qué estilos necesitan e incluyen automáticamente éstos en el stylesheet.

Los temas se pueden basar en el tema estándar, que es muy simple pero funcional. Se pueden también basar en cualquier otro tema. Esto permite que crees fácilmente familias de temas, o las variaciones de un tema.

### **3.5 IDIOMAS**

Moodle ha sido diseñado para ser internacional. Cada cadena o página de texto que se presenta como parte de la interfaz surge de una serie de archivos de idioma. Cada idioma es un subdirectorio del directorio lang. La estructura del directorio lang es la que sigue:

- lang/en - directorio que contiene todos los archivos para un idioma.
- moodle.php - cadenas para la interfaz principal
- assignment.php - cadenas para el módulo de tareas
- choice.php - cadenas para el módulo consulta
- forum.php - cadenas para el módulo del foro
- journal.php - cadenas para el módulo del diario
- quiz.php - cadenas para el módulo del cuestionario

- resource.php - cadenas para el módulo de materiales
- survey.php - cadenas para el módulo de encuesta

Se llama a las cadenas desde los archivos usando las funciones: `get_string()` o `print_string()`.

- lang/en/help - contiene todas las páginas de ayuda (para las ayudas emergentes sensibles al contexto)

Las páginas principales de ayuda están situadas aquí, mientras que las páginas específicas de cada módulo están localizadas en subdirectorios con el nombre del módulo. Con la función `helpbutton`, se puede insertar un botón de ayuda en una página.

No obstante también es posible editar los idiomas en línea, usando las herramientas web de Administración bajo "Idioma". Esto hace que sea fácil no sólo crear nuevos idiomas sino también refinar los existentes.

## 3.6 ESQUEMA DE BASES DE DATOS

Dada una base de datos funcionando con tablas definidas, el intencionalmente simple *SQL* usado en *Moodle* debe funcionar bien con una amplia variedad de marcas de bases de datos.

Existe un problema con la creación automática de nuevas tablas en una base de datos, que es lo que *Moodle* intenta hacer tras la instalación inicial. Debido a que cada base de datos es muy diferente de las otras, aún no existe una manera de hacer esto de manera independiente del tipo de plataforma. Para ayudar a la automatización en cada base de datos, pueden crearse esquemas que enumeren el *SQL* requerido para crear tablas en *Moodle* en una base de datos determinada. Estos son los archivos que hay en `lib/db` y dentro del subdirectorio `db` de cada módulo.

## 4. OTRAS CARACTERÍSTICAS DE MOODLE

### ESCALABILIDAD

La infraestructura debe poder ampliarse o escalar para resolver el futuro crecimiento, tanto en términos de volumen de contenidos educativos como del número de estudiantes.

*Moodle* funciona con una amplia variedad de tecnologías de servidores web y bases de datos. Al igual que sucede con cualquier instalación de sistemas de software basados en servidor y con los sistemas de bases de datos, resulta crucial elegir muy cuidadosamente los equipos, el

sistema operativo y el sistema de bases de datos, a fin de asegurar que el sistema puede afrontar un gran rendimiento.

Sería posible lograr un equilibrio de cargas en una instalación *Moodle* usando, por ejemplo, más de un servidor web si fuera necesario. Los servidores web separados consultarían la misma base de datos y apuntarían a la misma área de almacenamiento de archivos pero, en otro caso, la separación de las capas de la aplicación resulta del todo suficiente para hacer viable esta clase de *clustering*. De igual modo, la base de datos podría ser un conglomerado (*cluster*) de servidores.

Todo lo anterior implica que la arquitectura de *Moodle* facilita responder a futuras demandas, adaptando las tecnologías bajo las que se ejecuta. Esto sería posible incluso en un entorno vivo, a fin de mejorar el servicio sin interrupciones importantes.

## **FACILIDAD DE USO**

*Moodle* es una aplicación sencilla y potente. Tiene un interfaz familiar e intuitivo para la navegación que hace fácil crear cursos, y gracias a su diseño modular hace fácil agregar contenidos que motiven al estudiante.

## **ALTA DISPONIBILIDAD**

El *LMS* debe ser lo suficientemente robusto como para satisfacer las diversas necesidades de miles de estudiantes, administradores, creadores de contenidos y profesores simultáneamente.

Los patrones de uso variarán considerablemente dependiendo del contexto específico de la implementación, pero en términos generales *Moodle* presenta una interfaz basada en web de alta disponibilidad, permitiendo a los alumnos, tutores y administradores iniciar sesión de manera permanente y ejecutar sus tareas diarias.

## **COMUNIDAD ACTIVA**

*Moodle* tiene una comunidad activa muy grande de gente que usa el sistema y que desarrolla nuevas funcionalidades y mejoras. Son varios cientos los desarrolladores con acceso a los archivos de la aplicación y muy frecuentados los foros de sugerencias, comentarios y evaluación de cada uno de los módulos.

La comunidad *Moodle* ha sido indispensable para el éxito del sistema. Con tantos usuarios globales, siempre hay alguien que puede contestar a una pregunta o dar un consejo. Al mismo tiempo, los desarrolladores y usuarios de *Moodle* trabajan juntos para asegurar la calidad, añadir nuevos módulos y funcionalidades y proponer nuevas ideas para desarrollar. Como los usuarios son libres de experimentar, muchas personas usan y prueban nuevas características, actuando como un gran departamento de control de calidad.

Las tres ventajas ya mencionadas hacen que *Moodle* sea único dentro del universo de los *LMS*, pero además cumple con los requisitos inherentes a la definición de un buen gestor de contenidos educativos. Se detallarán a continuación.

## **INTEROPERABILIDAD**

Para admitir contenido de diferentes fuentes, y soluciones de equipos de cómputo o programas de diversos proveedores, el *LMS* debería intercambiar información utilizando estándares abiertos de la industria para implementaciones web.

En cuanto a la autenticación, *Moodle* admite autenticación contra *LDAP*, el protocolo estándar más utilizado con este propósito.

En lo referente al contenido, existen otros aspectos:

- *Moodle* admite la importación/exportación de Objetos Reutilizables de Aprendizaje empaquetados de acuerdo a los estándares *IMS Content Packaging* y *SCORM*.
- Las preguntas de los cuestionarios pueden ser exportadas en el formato estándar internacional *IMS QTI 2*.
- En *Moodle*, los canales de noticias RSS pueden integrarse en un sitio web completo o un curso.
- Se puede acceder a las discusiones de los foros como noticias RSS, y por lo tanto integrarse en otros sistemas o sitios web con funcionalidad RSS.

En *Moodle*, el uso de *XML* para importar/exportar información es un procedimiento estándar. El método servicios WEB de intercambio de información con otros sistemas todavía no es estándar, pero está en continuo desarrollo. De hecho, será el empleado para la realización de este proyecto, como se verá más adelante.

### **ESTABILIDAD**

La infraestructura del LMS puede soportar de manera confiable y efectiva una implementación productiva a gran escala las 24 horas del día, los 7 días de la semana.

### **SEGURIDAD**

Al igual que sucede con cualquier solución colaborativa, el LMS puede limitar y controlar selectivamente el acceso de su diversa comunidad de usuarios a los contenidos en línea, recursos y funciones del servidor tanto interna como externamente.

## **5. VERSIONES DE MOODLE**

*Moodle* como ya se ha explicado, se encuentra en continuo desarrollo, por lo que lo normal es que haya una nueva actualización de la última versión, cada 2 o 3 meses.

Para nombrar a sus versiones *Moodle* hace lo siguiente, no presenta una nueva versión, hasta que no considera que los cambios respecto a la anterior sean sustanciales. Esto quiere decir, que si por ejemplo alguien presenta una nueva funcionalidad pero que no implica un cambio importante, dicha funcionalidad se presentará junto con otras en una actualización de la versión actual. Es por esto que si atendemos a la historia de las versiones, vemos como por ejemplo entre las versiones estables 2.6 y 2.7, tenemos hasta 8 actualizaciones de la versión 2.6, pero todas ellas (2.6.1, 2.6.2...) no suponen versiones en sí mismas sino actualizaciones de la versión inmediatamente anterior (Moodle, 2015).

Este proyecto se ha desarrollado sobre *Moodle 2.7.2* ya que era la última actualización estable disponible cuando se empezó a desarrollar el mismo (Julio 2014). Coincidiendo también con el que había instalado en ese momento en la escuela.

El proyecto del que partimos, se había desarrollado sobre *Moodle 1.9* por lo que actualmente se encontraba obsoleto debido a que hay una serie de cambios muy importantes en la configuración de *Moodle*.

Principalmente estos cambios influyen de manera notable en la configuración de la base de datos, debido a cambios en el cifrado de la información y la supresión de muchas tablas, teniendo en cuenta también que la interacción entre dichas tablas ha cambiado en casi todos los casos.

Estos cambios se conocerán en detalle en el capítulo que se explica el desarrollo de este proyecto.

## **6. MOODLE MÓVIL**

Actualmente podemos encontrar varias en Google Play. Las más destacables son las siguientes (atendiendo a sus valoraciones):

- *LHU Moodle Mobile*: Es una aplicación privada, para estudiantes de la Liverpool Hope University. No se puede disponer de ella para cualquier *Moodle*.
- *MDroid Legacy - Moodle mobile*: Es una aplicación que empezó a desarrollarse, pero actualmente se encuentra obsoleta como podemos observar en la descripción de la misma en Google Play.
- *Moodle Mobile*: Es la aplicación más completa de todas las disponibles. Es la oficial de *Moodle*, en la descripción podemos ver como se ofertan casi todos los servicios de la versión para PC, pero atendiendo a los comentarios de los usuarios vemos como esto todavía está lejos de la realidad.

Son muchos los que hacen referencia a que la aplicación está mal programada, y que tarda demasiado en cargar, o en otros casos directamente no funciona. Podemos ver en la Figura 3, los comentarios más recientes. En ellos podemos apreciar la disconformidad de los usuarios con esta App, sobre todo referidos a la lenta navegación debida a la mala optimización de la misma (Google Play, 2015).

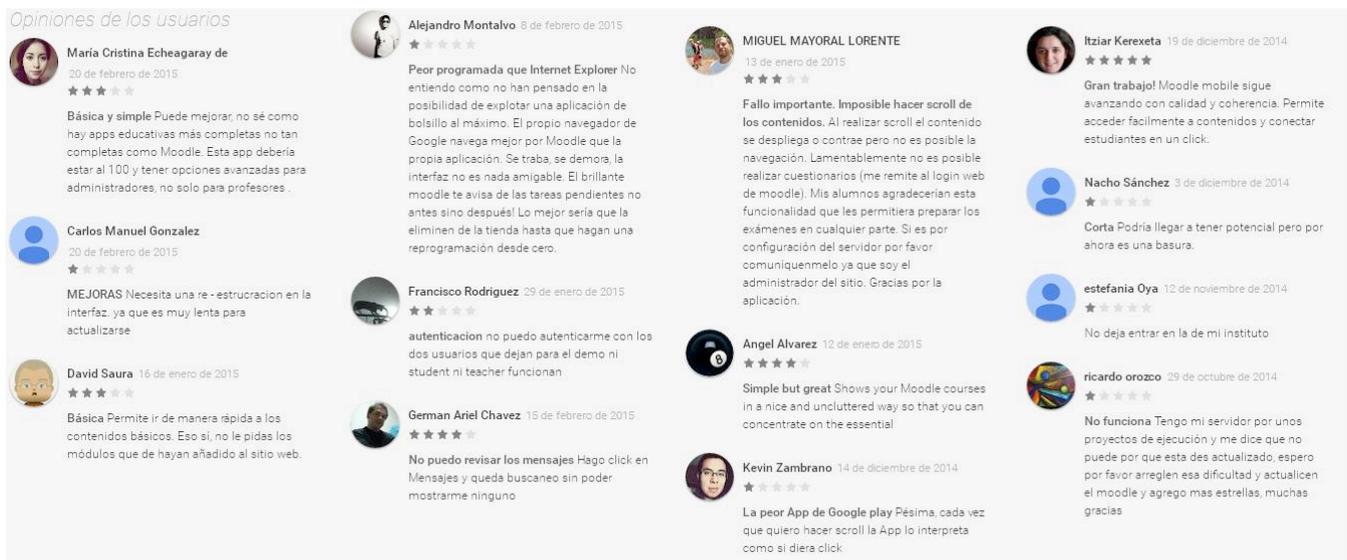


Figura 3 – Opiniones Moodle Mobile

Se va a ofrecer, sin poder ofertar el total de funcionalidades que tiene *Moodle*, el acceso a los recursos, la mensajería (Antolín Blanco, 2015) y tras la integración, la aplicación final contendrá a mayores los servicios de foros y la evaluación de conocimientos a través del acceso a los cuestionarios que ofrece *Moodle*, de una manera rápida, sencilla y optimizada. Evitando así los problemas de los que hablan los usuarios de la app ya existente. Como desventaja no tendremos todas las posibilidades que nos da *Moodle* desarrolladas en nuestra App pero las que tendremos estarán 100% integradas funcionando de manera correcta. Dispondremos una interfaz simple pero efectiva, y realizaremos una optimización de las funciones para lograr una navegación rápida.



# **CAPÍTULO IV. SERVICIOS WEB**



## 1. INTRODUCCIÓN A LOS SERVICIOS WEB

Los Servicios Web (Web Service, o Web Services), atendiendo a la definición que da el W3C se pueden definir como “Un conjunto de aplicaciones o de tecnologías con capacidad de interoperar en la web. Estas tecnologías intercambian datos entre sí con el fin de ofrecer un servicio. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la red” (World Wide Web Consortium, 2013).

Como resumen podríamos decir que un Servicio Web es un tipo de *middleware* que permite la comunicación entre aplicaciones remotas que interactúan entre sí para presentar información dinámica al usuario.

Es necesaria una arquitectura de referencia estándar para poder proporcionar interoperabilidad entre estas aplicaciones y que al mismo tiempo sea posible su combinación para realizar operaciones complejas.

Las características deseables para poner el Servicio Web a disposición para su uso son las siguientes:

- Es necesario definir en primer lugar el Servicio Web. Para favorecer la interacción con otros se debe utilizar un lenguaje común que permita estructurar los datos que componen el servicio. Para ello se suele hacer uso de XML (*eXtensible Markup Language*).
- Un servicio debe poder ser accesible a través de la Web. Para ello debe utilizar protocolos de transporte estándares como HTTP, y codificar los mensajes en un lenguaje estándar que pueda conocer cualquier cliente que quiera utilizar el servicio.
- Un servicio debe contener una descripción de sí mismo. De esta forma, una aplicación podrá saber cuál es la función de un determinado Servicio Web, y cuál es su interfaz, de manera que pueda ser utilizado de forma automática por cualquier aplicación, sin la intervención del usuario. El lenguaje que se utiliza para la descripción del servicio es WSDL (*Web Service Description Language*), y la publicación se del mismo se realiza mediante UDDI (*Universal Description, Discovery and Integration*), indiferentemente del carácter del servidor (público o privado).
- Debe poder ser localizado. Deberemos tener algún mecanismo que nos permita encontrar un Servicio Web que realice una determinada función. De esta forma tendremos la posibilidad de que una aplicación localice el servicio que necesite de forma automática, sin tener que conocerlo previamente el usuario.

Desde el punto de vista del usuario, cuando se quiera solicitar un Servicio Web, se debe de disponer de un directorio que tenga las referencias a los servicios disponibles. Para intercambiar información entre el proveedor y el solicitante se debe de hacer uso de un protocolo de comunicaciones, como SOAP (*Simple Object Access Protocol*), que transmitirá los datos sobre HTTP, FTP o SMTP en formato XML, o REST, que intercambiará la información sobre HTTP en formato XML o JSON.

## 2. XML: eXtensible Markup Language

XML (*eXtensible Markup Language*, Lenguaje de Marcado eXtensible) es un lenguaje desarrollado por W3C (*World Wide Web Consortium*) que está basado en SGML (*Standard Generalized Markup Language*, Lenguaje de Marcado Generalizado Estándar).

XML es un lenguaje utilizado para el almacenamiento e intercambio de datos estructurados entre distintas plataformas. Es un metalenguaje, es decir, puede ser empleado para definir otros lenguajes, llamados dialectos XML, como por ejemplo GML, MathML, RSS...

Al igual que HTML, XML es un lenguaje de marcado, pero a diferencia del primero que fue diseñado para mostrar datos, XML fue diseñado para transportar y almacenar esos datos, centrándose en el contenido.

Normalmente diferentes sistemas independientes trabajan con formatos incompatibles, XML se encarga de simplificar el intercambio de estos datos, ya que almacena los mismos en formato de texto plano, proporcionando un software independiente de la forma de almacenamiento de los datos.

La unidad fundamental de un documento XML es el elemento, o lo que es lo mismo: Un par de etiquetas de inicio y fin coincidentes y el contenido que se encuentra en el medio de las dos. En un documento XML debe de haber un único elemento raíz que contenga al resto de elementos, y los elementos deberán estar correctamente anidados.

Un ejemplo de una correcta anidación es el siguiente:

```
<etiqueta1><etiqueta2></etiqueta2></etiqueta1>
```

Un ejemplo incorrecto por el contrario sería este otro:

```
<etiqueta1><etiqueta2></etiqueta2></etiqueta1></etiqueta2>
```

En un documento XML, todos los nombres de los elementos son *case sensitive*, es decir, sensibles a letras minúsculas y mayúsculas, teniendo que cumplir las siguientes normas:

- Pueden contener letras minúsculas, letras mayúsculas, números, puntos ".", guiones medios "-" y guiones bajos "\_".
- Asimismo, pueden contener el carácter dos puntos ":". No obstante, su uso se reserva para cuando se definan espacios de nombres.
- El primer carácter tiene que ser una letra o un guión bajo "\_".

Las etiquetas en XML no están definidas sino que cada uno debe definir sus propias etiquetas.

Por otra parte, hay que tener en cuenta que, detrás del nombre de una etiqueta se permite escribir un espacio en blanco o un salto de línea. Por ejemplo, sintácticamente es correcto escribir:

```
<etiqueta1 >Nombre</etiqueta1>
```

Otro concepto importante es el de atributo. Estos pueden aparecer dentro de las etiquetas de inicio como pares "nombre=valor". Además, un mismo atributo solo puede aparecer una vez en una etiqueta.

Las letras no inglesas (á, Á, ñ, Ñ...) están permitidas. Sin embargo, es recomendable no utilizarlas para reducir posibles incompatibilidades con programas que puedan no reconocerlas.

En cuanto al carácter guión medio "-" y al punto ".", aunque también están permitidos para nombrar etiquetas, igualmente se aconseja evitar su uso; el guión medio porque podría confundirse con el signo menos, y el punto porque, por ejemplo al escribir nombre.completo, podría interpretarse que completo es una propiedad del objeto nombre (Pes, 2014).

### 3. WSDL

WSDL (Web Services Description Language) es un lenguaje basado en XML utilizado para describir la funcionalidad que proporciona un servicio Web. Una descripción WSDL (fichero WSDL) de un servicio web proporciona una descripción entendible por la máquina (*machine readable*) de la interfaz del servicio Web, indicando cómo se debe llamar al servicio, qué parámetros espera, y qué estructuras de datos devuelve (Dept. Ciencia de la Computación, 2014).

WSDL describe un servicio utilizando varios elementos (etiquetas xml). Dichos elementos podemos clasificarlos como abstractos o concretos. La parte WSDL abstracta describe las operaciones y mensajes con detalle. En otras palabras, la parte abstracta de un WSDL especifica qué hace el servicio:

- Qué operaciones están disponibles
- Qué entradas, salidas, y mensajes de error tienen las operaciones
- Cuáles son las definiciones de los tipos para los mensajes de entrada, salida y error

En el mundo Java, podemos pensar en la parte abstracta de un WSDL como en la definición de una interfaz o una clase abstracta, con la definición de sus métodos, pero no sus implementaciones. La parte abstracta de un WSDL contiene dos componentes principales:

- Las operaciones que forman la definición de la interfaz
- Los tipos de datos para los parámetros de entrada, salida y error, de las operaciones

La parte WSDL concreta describe el cómo y dónde del servicio:

- Cómo tiene que llamar un cliente al servicio
- Qué protocolo debería usar
- Dónde está disponible el servicio

En el mundo Java podemos pensar en la parte concreta de un WSDL como en la implementación de la parte abstracta, aunque en términos de servicios Web, solamente describe dónde se encuentra dicha implementación para utilizarse. La parte concreta de un WSDL contiene dos componentes principales:

- Información de enlazado (binding) sobre el protocolo a utilizar
- La dirección en donde localizar el servicio

## **4. UDDI**

UDDI nos permite localizar Servicios Web. Para ello define la especificación para construir un directorio distribuido de Servicios Web, donde los datos se almacenan en XML. En este registro no sólo se almacena información sobre servicios, sino también sobre las organizaciones que los proporcionan, la categoría en la que se encuentran, y sus instrucciones de uso (normalmente WSDL). Tenemos por lo tanto 3 tipos de información relacionados entre sí:

- Páginas blancas: Datos de las organizaciones (dirección, información de contacto, etc).
- Páginas amarillas: Clasificación de las organizaciones (según tipo de industria, zona geográfica, etc).
- Páginas verdes: Información técnica sobre los servicios que se ofrecen. Aquí se dan las instrucciones para utilizar los servicios. Es recomendable que estas instrucciones se especifiquen de forma estándar mediante un documento WSDL.

## **5. RELACIÓN DE LOS SERVICIOS WEB CON OTRAS TECNOLOGÍAS**

Los servicios web son, principalmente, una tecnología de integración. Sin embargo, son independientes de la forma propiamente dicha. Las tecnologías de componentes para servicios web son definidas en forma común e interactúan en XML, según se mencionó anteriormente. Sin embargo, ya que el propio XML es independiente de lenguaje, los servicios web también lo son. Por tanto, los servicios web se pueden desarrollar en varios lenguajes de programación, como Java, Python, Android, Perl, C#, Basic y otros.

El caso que a nosotros nos interesa es la compatibilidad con Android, vamos a conocer las características más básicas de los dos protocolos de comunicación más usados a la hora de crear un servicio web y finalizaremos este apartado eligiendo uno de ellos, añadiendo una explicación de por qué nos resulta más útil en detrimento de los protocolos rechazados.

## **6. SOAP (Simple Object Access Protocol):**

Se trata de un protocolo derivado de XML que nos sirve para intercambiar información entre aplicaciones.

Normalmente utilizaremos SOAP [www.w3.org/TR/soap/](http://www.w3.org/TR/soap/) para conectarnos a un servicio e invocar métodos remotos, aunque puede ser utilizado de forma más genérica para enviar cualquier tipo de contenido. Podemos distinguir dos tipos de mensajes según su contenido:

- Mensajes orientados al documento: Contienen cualquier tipo de contenido que queramos enviar entre aplicaciones.
- Mensajes orientados a RPC: Este tipo de mensajes servirá para invocar procedimientos de forma remota (Remote Procedure Calls). Podemos verlo como un tipo más concreto dentro del tipo anterior, ya que en este caso como contenido del mensaje especificaremos el método que queremos invocar junto a los parámetros que le pasamos, y el servidor nos deberá devolver como respuesta un mensaje SOAP con el resultado de invocar el método.

Puede ser utilizado sobre varios protocolos de transporte, aunque está especialmente diseñado para trabajar sobre HTTP.

En primer lugar hay que empezar diciendo que Android no incluye en su SDK ningún tipo de soporte para el acceso a servicios web de tipo SOAP. Es por esto por lo que vamos a utilizar una librería externa para hacernos más fácil esta tarea. Entre la oferta actual, la opción más popular y más utilizada es la librería ksoap2-android. Esta librería es un fork, especialmente

adaptado para Android, de la antigua librería kSOAP2. Este framework nos permitirá de forma relativamente fácil y cómoda utilizar servicios web que utilicen el estándar SOAP (Paszniuk, 2014).

## **7. RESTful**

Los servicios del tipo REST son los más utilizados en la actualidad a la hora de crear un servicio Web.

Se trata de un protocolo cliente/servidor sin estado. Esto quiere decir que cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en HTTP utilizan cookies y otros mecanismos para mantener el estado de la sesión (algunas de estas prácticas, como la reescritura de URLs, no son permitidas por REST).

Un concepto importante en REST es la existencia de recursos (elementos de información), que pueden ser accedidos utilizando un identificador global (un Identificador Uniforme de Recurso). Para manipular estos recursos, los componentes de la red (clientes y servidores) se comunican a través de una interfaz estándar (HTTP) e intercambian representaciones de estos recursos (los ficheros que se descargan y se envían)

Este protocolo tiene una serie de operaciones bien definidas como son, que se aplican a todos los recursos de información: HTTP en sí define un conjunto pequeño de operaciones, las más importantes son POST, GET, PUT y DELETE. Las cuales veremos con más detenimiento en el capítulo de desarrollo del proyecto. Para poder realizar peticiones http en Android deberemos de usar la librería http-request (Amatellanes, 2014).

## **8. ELECCIÓN**

REST y SOAP pueden utilizarse para implementar una funcionalidad similar, pero en general SOAP debe utilizarse cuando se necesita una característica concreta de SOAP, y REST generalmente es la mejor opción en caso contrario.

REST y SOAP muchas veces se utilizan una en lugar de la otra, pero son totalmente diferentes enfoques. REST es un estilo de arquitectura para generar aplicaciones de cliente-servidor. SOAP es una especificación de protocolo para intercambiar datos entre dos extremos.

Para el desarrollo de este proyecto se ha decidido realizar la implementación utilizando RESTful, atendiendo a los siguientes motivos:

- Como ya hemos explicado en su apartado, se trata de un tipo de implementación basada totalmente en especificaciones del protocolo HTTP, donde todo actúa como un recurso. Todos los mensajes se tratan de igual manera, sin diferenciar el contenido, esto nos reporta una sencillez evidente.
- Dichos mensajes son ligeros, lo cual nos ayuda a que la escalabilidad y el rendimiento no nos suponga un problema. Esta característica nos resulta especialmente útil, ya que hay que recordar que nuestra aplicación va a ser una aplicación para dispositivos móviles, y aunque los procesadores de los mismos cada vez son más potentes, todavía se encuentran lejos de los procesadores de los ordenadores convencionales o servidores.
- SOAP es una implementación mucho más estricta, se basa en un estándar oficial, mientras que RESTful tiene recomendaciones de implementación, pero no resultan tan rigurosas.
- RESTful a la hora de la representación de los datos, permite XML o JSON, mientras que SOAP especifica que deben de ser representados en XML.
- Por último cabe recordar que este proyecto parte de otro ya empezado, en el que RESTful era el protocolo elegido y por lo tanto atendiendo a las anteriores razones y a esta última, creemos que continuar el proyecto utilizando las mismas tecnologías que el proyecto del que se parte suponen la mejor opción.

## **9. REPRESENTACIÓN DE LOS DATOS**

Como se ha explicado en el apartado anterior a este, RESTful permite representar los datos en formato XML o JSON, lo cual nos lleva a otra discusión, teniendo que elegir entre una de estas dos opciones. XML se explicó al comienzo de este capítulo como parte fundamental para poder entender bien los Servicios Web, JSON en cambio no ha sido explicada hasta ahora.

JSON (JavaScript Object Notation) es un formato para el intercambio de datos, básicamente JSON describe los datos con una sintaxis dedicada que se usa para identificar y gestionar los datos. JSON nació como una alternativa a XML, el fácil uso en javascript ha generado un gran número de seguidores de esta alternativa.

Una de las mayores ventajas que tiene el uso de JSON es que puede ser leído por cualquier lenguaje de programación. Por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías.

La principal ventaja de JSON frente a XML es que escribir un analizador sintáctico mediante JSON es mucho más sencillo que utilizando XML, y su procesado es más rápido en cualquier ordenador.

En entornos donde el flujo de datos es alto entre cliente y servidor y donde los tiempos de procesado son más bien pequeños, hace que sea más utilizado JSON, ya que los tiempos de respuesta se optimizan debido a su ligero peso.

La representación de datos elegida será JSON. Al igual que se ha explicado en la elección de RESTful, el proyecto del que partimos utiliza JSON como representación de datos, lo que ya nos supone una razón de peso para utilizarlo de nuevo. A parte de por esta razón, hemos elegido este tipo de representación por los siguientes motivos:

- Es mucho más simple que XML.
- JSON es la mejor herramienta para compartir datos. Esto es porque los datos están almacenados en vectores y registros mientras que XML almacena los datos en árboles. Ambos tienen sus ventajas, pero las transferencias de datos son mucho más fáciles cuando los datos se almacenan en una estructura que está familiarizada a los lenguajes orientados a objetos. Esto hace que sea muy sencillo importar datos desde un fichero JSON a Android, Perl, Ruby, Javascript, Python, y otros muchos lenguajes. Para hacer lo mismo con XML, necesitaría primero transformar los datos antes de que puedan ser importados. Por este motivo, JSON es un formato de fichero superior para las APIs web.
- Ambos ficheros XML y JSON son legibles para el ser humano. Al menos, son entendibles para los programadores que trabajan con estos formatos de ficheros. Sin embargo, los ficheros JSON son más restrictivos y por lo tanto ligeramente más legibles. Esto se debe a que el número de formatos de datos permitidos por JSON es mucho menor que XML. Además, la estructura de los datos está más estandarizada con los ficheros JSON debido al hecho de que existen menos opciones cuando se compara con el formato XML.



# **CAPÍTULO V.**

## **DESARROLLO DEL PROYECTO**



Como se ha explicado en capítulos anteriores se han analizado las diferentes posibilidades de cada tecnología. A modo de resumir las elecciones realizadas se concreta que, para la parte móvil la mejor opción es desarrollar para *Android*, así como para la parte del servidor, dado que *Moodle* está implementado con *PHP-MySQL*, se ha seguido por esta línea. Por último, para la parte de servicios web la elección ha sido la arquitectura REST, con representación JSON.

A continuación, se verá cómo se ha ido realizando el desarrollo de la aplicación móvil, así como las comunicaciones con el servidor.

## 1. CONDICIONES PREVIAS

La implementación de este proyecto se ha llevado a cabo siempre teniendo presente que se trata de un complemento de la plataforma *Moodle*. El objetivo final ya mencionado en capítulos anteriores es:

1. Dar la posibilidad al usuario de la interacción con otros usuarios mediante el uso total de foros de discusión.
2. Evaluación de conocimientos mediante cuestionarios con posibilidad de preguntas de varios tipos.
3. Por último, la integración de la anterior versión que incluye formulario de nuevos registros, la autenticación, la mensajería de *Moodle* la visualización de cursos o asignaturas y la visualización y descarga de Recursos (Antolín Blanco, 2015).

En la actividad de foros el usuario podrá visualizar tanto foros de discusión, discusiones y mensajes de las discusiones como crear nuevas discusiones y responder o enviar nuevos mensajes a las discusiones.

En la actividad de cuestionarios, el usuario podrá realizar todos los intentos que desee, sin límite de tiempo y obtendrá la calificación correspondiente a las respuestas que haya realizado.

Por último, el objetivo es crear una única aplicación que integre todo lo desarrollado hasta el momento.

## 2. APLICACIÓN MÓVIL. DISEÑO DE LA INTERFAZ

### 2.1 INTRODUCCIÓN

En una aplicación *Android* la interfaz de usuario es construida usando objetos *View* y *ViewGroup*, u objetos que sean herederos de estos. De esta forma, *View* sirve de base para una serie de subclases llamadas *widgets*, que no son otra cosa que objetos de la interfaz de usuario completamente desarrollados, como campos de texto o botones. Por su parte, *ViewGroup* hace otro tanto de lo mismo sirviendo de base a las subclases llamadas *layouts*,

las cuales ofrecen arquitecturas de disposición de elementos tipo *Linear*, *Tabular*, *Relative* y un largo etcétera.

Un objeto *View* es una estructura de datos cuyas propiedades almacenan los parámetros de disposición y contenido de un área específica de la pantalla del dispositivo. Además, este objeto es capaz de gestionar sus propias medidas, su disposición, el *scroll* si es que lo hay y prácticamente cualquier interacción por parte del usuario en el área en el que está definido. Los objetos *ViewGroup* son utilizados como puntos de ramificación del árbol de jerarquía de vistas definido en un archivo *XML layout*, y de ellos colgarán distintos elementos *View* o *ViewGroup*, tal y como se puede observar en la Figura 4:

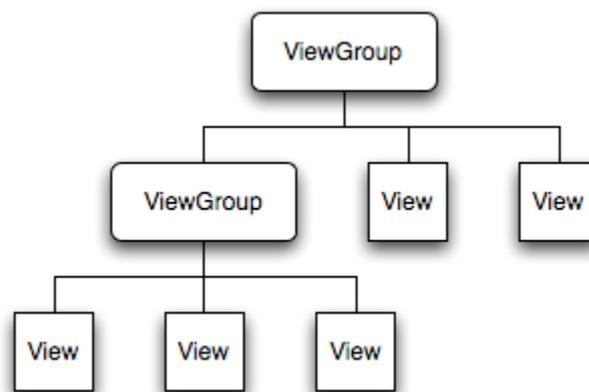


Figura 4 - Jerarquía de Vistas en Android

La jerarquía puede ser tan simple o compleja como el desarrollador desee y puede ser construida a partir de *layouts* y *widgets* predefinidos por *Android* o implementando objetos *View* personalizados. Para unir el árbol definido en la jerarquía de vistas a la pantalla del dispositivo para el renderizado es necesario que la *Activity* a la que se quiere adjuntar la jerarquía de vistas llame al método *setContentview()* pasando una referencia al nodo del objeto raíz. El sistema *Android* utilizará dicha referencia para dibujar el árbol de vistas en la pantalla.

Un *widget* es un objeto que extiende la clase *View* y que sirve como interfaz para la interacción con el usuario. *Android* adjunta un conjunto de *widgets* completamente implementados, como botones, *checkboxes* y campos de entrada de texto entre muchos otros que permiten construir de una forma rápida y eficaz la interfaz de usuario de cualquier aplicación.

En cuanto a los *layouts*, es importante destacar que esta palabra tiene dos acepciones en el entorno de *Android*. Por una parte, los *layouts* son los archivos XML utilizados para definir la jerarquía de vistas asociada a una *Activity*. Cada elemento del archivo XML será un objeto

*View* o *ViewGroup* (o un descendiente de los mismos). La segunda acepción de *layout* hace referencia a todas las subclases de *ViewGroup* que pueden ser añadidas a la jerarquía de vistas.

El nombre de los elementos del archivo XML representa la clase de Java a la que hace referencia el nodo, por lo que un elemento con el nombre `<TextView>` hará referencia a un objeto heredero de *View TextView*, mientras que un nodo `<LinearLayout>` representará a la clase heredera de *ViewGroup LinearLayout*. En cualquier caso, y tal y como se puede observar en la Figura 3.1, el nodo padre del *layout* siempre ha de ser heredero de *ViewGroup*.

## 2.2 INTERFAZ DE LA APLICACIÓN MÓVIL

A la hora de desarrollar la interfaz de esta aplicación móvil, se ha optado por seguir uno de los principios de diseño de *Moodle*: la simplicidad. Por lo tanto se va a utilizar el mínimo de interfaz necesario para obtener la funcionalidad que se desea implementar.

Esta aplicación muestra la gran mayoría de los *layouts* están representados por *ListViews*, como puede ser el listado de cursos, temas, foros, discusiones, mensajes del foro, cuestionarios y lista de preguntas de cada cuestionarios.

Los *layouts* que no están incluidos en la anterior enumeración son menús que están formados por *Buttons* y la representación de cada pregunta perteneciente a un cuestionario ya que tendrá un formato dinámico específico para mostrar las posibles respuestas (*Checkbox*, *RadioButton*) o la solicitud de la respuesta (*Edittext*) dependiendo el tipo de pregunta que sea.

En un capítulo posterior le dedicaremos a explicar todas las funcionalidades implementadas con detenimiento a modo de manual de usuario.

## 3. ESTRUCTURA DE LA BASE DE DATOS DE MOODLE

La base de datos de *Moodle* tiene alrededor de 250 tablas, dato que de primeras nos puede resultar abrumador y del cual podríamos deducir una gran complejidad. Nada más lejos de la realidad, ya que cada funcionalidad de *Moodle* tiene de manera intrínseca una interacción con un gran número de dichas tablas.

Todas las tablas de la base de datos de *Moodle* comienzan con el prefijo “mdl\_” continuado de una palabra clave que nos ayuda a ubicar dicha tabla. Por ejemplo para el caso de los foros, la tabla principal será “mdl\_forum”.

Refiriéndonos a este proyecto en concreto, es buen momento para recordar que funcionalidades tienen relación con la base de datos del servidor *Moodle*:

- Visualización de foros.

- Visualización de discusiones.
- Visualización de mensajes del foro
- Inserción de nuevas discusiones y nuevos mensajes
- Visualización de cuestionarios.
- Visualización de las preguntas del cuestionario.
- Visualización de cada pregunta con sus posibles respuestas.
- Inserción de intentos al acceder a responder un cuestionario.
- Inserción y visualización de la nota obtenida al responder el cuestionario.

De manera muy resumida, acabamos de enumerar las funcionalidades que desarrollaremos en este proyecto, cada una de ellas está relacionada con una serie de tablas de la base de datos. Por lo tanto lo que vamos a hacer no es explicar las 250 tablas, sino únicamente, explicaremos aquellas tablas con las que hemos tenido que interactuar para desarrollar nuestra aplicación. Y más concretamente explicaremos los campos de cada tabla que nos han resultado de interés. No es necesario conocer el resto de tablas para poder entender perfectamente el desarrollo de nuestro trabajo.

### Tablas relacionadas con el módulo de foros de Moodle:

#### Mdl\_forum

Esta tabla contiene información de los foros en el sistema, es la tabla más general de este modulo.

Los campos

- *Course*: Recogen el curso al cual pertenece el foro.
- *Name*: El nombre con el que se le identifica al foro.
- *Intro*: La introducción que informa para que va destinado el uso de dicho foro.
- *Timemodified*: el tiempo de modificación del foro que es actualizado cuando se introduce tanto una discusión como un nuevo mensaje en el foro.

Estos son los campos que se usan en las consultas a esta tabla.

mdl_forum	
ID BIGINT	A P
NAME VARCHAR(255)	
INTRO LONGTEXT	
TIMEMODIFIED DECIMAL	
COURSE BIGINT	U

Figura 5 - Campos de Mdl\_forum

## Mdl\_forum\_discussions

Recoge la información de las discusiones de los foros.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Course*: identificador del curso al que pertenece la discusión.
- *Forum*: identificador del foro al que pertenece la discusión.
- *Name*: nombre de la discusión.
- *Firstpost*: identificador del post del mensaje con el que se crea la discusión.
- *Userid*: identificador del usuario quien crea la discusión.
- *Timemodified*: identifica el tiempo en segundos de la última modificación de la discusión.
- *Usermodified*: identificador del último usuario que ha modificado la discusión.

mdl_forum_discussion	
ID BIGINT	A P
COURSE BIGINT	F
FORUM BIGINT	F
NAME VARCHAR(255)	
FIRSTPOST BIGINT	F
USERID BIGINT	F
TIMEMODIFIED DECIMAL	
USERMODIFIED BIGINT	F

Figura 6 - Campos de Mdl\_forum\_discussion

## Mdl\_forum\_posts

Recoge la información de los mensajes de los foros.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Discussion*: Identificador de la discusión a la que pertenece.
- *Parent*: Si es distinto de '0' tendrá el valor del identificador del mensaje que ha sido contestado, y si es '0' es que el mensaje no contesta a nadie, es decir, es un nuevo mensaje en la discusión.
- *Userid*: Identificador del usuario que ha enviado el mensaje.
- *Created*: Tiempo en segundos de la creación del mensaje.

- *Subject*: Tema del mensaje.
- *Message*: Contenido del mensaje.

mdl_forum_posts	
ID BIGINT	(A) (P)
DISCUSSION BIGINT	(F)
PARENT BIGINT	
USERID BIGINT	(U)
CREATED DECIMAL	
SUBJECT VARCHAR(255)	
MESSAGE LONGTEXT	

Figura 7 - Campos de Mdl\_forum\_posts

### Tablas relacionadas con el módulo de cuestionarios de Moodle:

#### Mdl\_quiz

Recoge la información de los cuestionarios.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Course*: Identificador del curso al que pertenece el cuestionario.
- *Name*: Nombre del cuestionario.
- *Sumgrades*: Suma de las notas máximas de las preguntas que lo forman.

mdl_quiz	
ID BIGINT	(P)
NAME VARCHAR(255)	
COURSE BIGINT	
SUMGRADES DECIMAL	

Figura 8 - Campos de Mdl\_quiz

#### Mdl\_quiz\_slots

Enlaza las preguntas con el cuestionario al que pertenecen.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Slot*: Posición que tiene la pregunta en el cuestionario.
- *Quizid*: Identificador del quiz.
- *Questionid*: Identificador de la pregunta.
- *Maxmark*: Máxima nota de la pregunta.

mdl_quiz_slot	
ID BIGINT	(A) (P)
SLOT BIGINT	
QUIZID BIGINT	(F)
QUESTIONID BIGINT	(F)
MAXMARK DECIMAL	

Figura 9 - Campos de Mdl\_quiz\_slots

### Mdl\_quiz\_attempts

Almacena cada intento realizado a los cuestionarios.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Quiz*: Identificador del cuestionario.
- *Userid*: Identificador del usuario.
- *Attempts*: Numero del intento.
- *Uniqueid*: Identificador del la tabla *mdl\_question\_usage* (que se crea una fila a la vez que se crea en la presente tabla).
- *State*: Estado en el que se encuentra el intento (*inprogress, finished*).
- *Timestart*: Tiempo en segundos del inicio del intento.
- *Timefinish*: Tiempo en segundos del final del intento.
- *Timemodified*: Tiempo en segundos de la modificación del intento.
- *Sumgrades*: NULL si el intento esta en curso, o la numero de la nota si el intento esta terminado.

mdl_quiz_attempts	
ID BIGINT	(A) (P)
QUIZ BIGINT	(U)
USERID BIGINT	(U)
ATTEMPTS MEDIUMINT	
UNIQUEID BIGINT	(U)
STATE VARCHAR(255)	
TIMESTART BIGINT	
TIMEMODIFIED BIGINT	
TIMEFINISH BIGINT	
SUMGRADES DECIMAL	

Figura 10 - Campos de Mdl\_quiz\_attempts

## Tablas relacionadas con el módulo de preguntas de Moodle:

### Mdl\_question

Recoge la información de las preguntas que forman los cuestionarios.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Name*: Nombre de la pregunta.
- *Questiontext*: Contenido de la pregunta.
- *Defaultmark*: Nota de la pregunta.
- *Qtype*: Tipo de la pregunta (*truefalse*, *numerical*, *multichoice*, *shortanswer*)

mdl_question	
ID	BIGINT (A P F)
NAME	VARCHAR(255)
QUESTIONTEXT	LONGTEXT
DEFAULTMARK	DECIMAL
QTYPE	VARCHAR(255)

Figura 11 – Campos de Mdl\_question

### Mdl\_question\_answers

Recoge la información de las respuestas posibles para las preguntas de la anterior tabla.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Question*: Identificador de la pregunta a la que esta asociada la respuesta.
- *Answer*: Contenido de la respuesta.
- *Fraction*: Nota asociada a la respuesta.

mdl_question_answers	
ID	BIGINT
ANSWER	LONGTEXT
FRACTION	DECIMAL
QUESTION	BIGINT (F)

Figura 12 – Campos de Mdl\_question\_answers

### Mdl\_qtype\_multichoice\_options

Indica si la pregunta que es de tipo opción múltiple es de una solo opción (*single=1*), o de varias (*single=0*).

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Questionid*: Identificador de la pregunta.
- *Single*: Determina si la pregunta es de una sola respuesta o de varias.

mdl_qtype_multichoice_options	
ID	BIGINT
QUESTIONID	BIGINT
SINGLE	SMALLINT

Figura 13 - Campos de Mdl\_qtype\_multichoice\_options

### Mdl\_question\_attempts

Recoge la información de los intentos a las preguntas y cuando se finaliza el cuestionario se almacena la respuesta final enviada.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Questionusage*: es el identificador generado en la tabla *mdl\_question\_usage*.
- *Slot*: La posición que tiene la pregunta en el cuestionario.
- *Questionid*: Identificador de la pregunta.
- *Questionsummary*: Texto de la pregunta y si la pregunta es de tipo multichoice este texto va seguido de todas las posibles respuestas separadas por punto y coma.
- *Rightanswer*: Texto de la respuesta correcta.
- *Responsesummary*: Texto de la respuesta respondida que solo se almacenara cuando se finalice el cuestionario.

mdl_question_attempts	
ID	BIGINT
QUESTIONUSAGEID	BIGINT
SLOT	BIGINT
QUESTIONID	BIGINT
QUESTIONSUMMARY	LONGTEXT
RIGHTANSWER	LONGTEXT
RESPONSESUMMARY	LONGTEXT

Figura 14 - Campos de Mdl\_question\_attempts

### Mdl\_question\_attempt\_steps

Recoge información tanto de los intentos parciales, es decir, cuando se responde a un cuestionario y se sale de él sin enviar como cuando se finaliza el intento.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Questionattemptid*: Identificador de la tabla anterior (*mdl\_question\_attempts*).
- *Fraction*: Nota obtenida en la pregunta, solo se rellena este campo en el intento que se finaliza.
- *Timecreated*: Tiempo en segundo de la creación del intento.
- *Userid*: Identificador del usuario quien realiza el intento.

mdl_question_attempt_steps	
ID BIGINT	(A) (P)
ATTEMPTSTEPID BIGINT	
NAME VARCHAR(255)	
VALUE LONGTEXT	
USERID BIGINT	(F)

Figura 15 - Campos de Mdl\_question\_attempt\_steps

### Mdl\_question\_attempt\_step\_data

Recoge información tanto de los intentos parciales, es decir, cuando se responde a un cuestionario y se sale de él sin enviar como cuando se finaliza el intento.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Attemptsepid*: Identificador de la tabla anterior (*mdl\_question\_attempt\_steps*).
- *Name*: Almacena diferentes textos en función del tipo de pregunta y número de intento, diferenciando el primero del resto hasta que se envía y termina el intento.
- *Value*: Almacena tanto el orden de las respuestas si la pregunta es multichoice, como los datos que han sido respondidos.

mdl_question_attempt_step_data	
ID BIGINT	(A) (P)
ATTEMPTSEPID BIGINT	
NAME VARCHAR(255)	
VALUE VARCHAR(255)	

Figura 16 - Campos de Mdl\_question\_attempt\_step\_data

### Mdl\_question\_usages

Recoge el “uso” de una pregunta en un intento, se genera una nueva fila por cada intento.

Los campos más significantes y que han sido útiles para las sucesivas consultas son:

- *Contextid*: Identificador de la tabla *mdl\_context*.

- *Component*: En esta aplicación este campo siempre va a contener “mod\_quiz”, ya que es el módulo que estamos utilizando.

mdl_question_usages	
ID BIGINT	A P F
CONTEXTID BIGINT	U
COMPONENT VARCHAR(255)	

Figura 17 - Campos de Mdl\_question\_usages

### Mdl\_context

El campo *id* de esta tabla es el mismo que el campo *contextid* de la tabla anterior. El campo *contextlevel* nos dice de qué tipo de módulo se trata, si es un “50” nos indica que es un curso, si es “70” que es un cuestionario. Nos vamos a centrar en este tipo ya que es el que nos interesa para desarrollar nuestra aplicación el dato del módulo curso ha sido utilizado en la aplicación que se ha integrado (Antolín Blanco, 2015) en la visualización de cursos y temas de cada curso.

Por último el campo *instanceid* nos proporciona la relación con el campo *id* de la tabla *mdl\_course*, identificando cada curso en concreto (Pueyo, 2009).

mdl_context	
ID BIGINT	A P F
CONTEXTLEVEL BIGINT	
INSTANCEID BIGINT	

Figura 18 - Campos de Mdl\_context

A continuación se muestran las relaciones con las tablas de la base de datos de *Moodle*, se ha dividido esta relación con finalidad de simplificar la comprensión de las mismas.

En la figura 19 se puede observar la relación entre las tablas relacionadas con el apartado de cuestionarios de esta aplicación, en él se muestran todas aquellas que tienen función en esta sección.

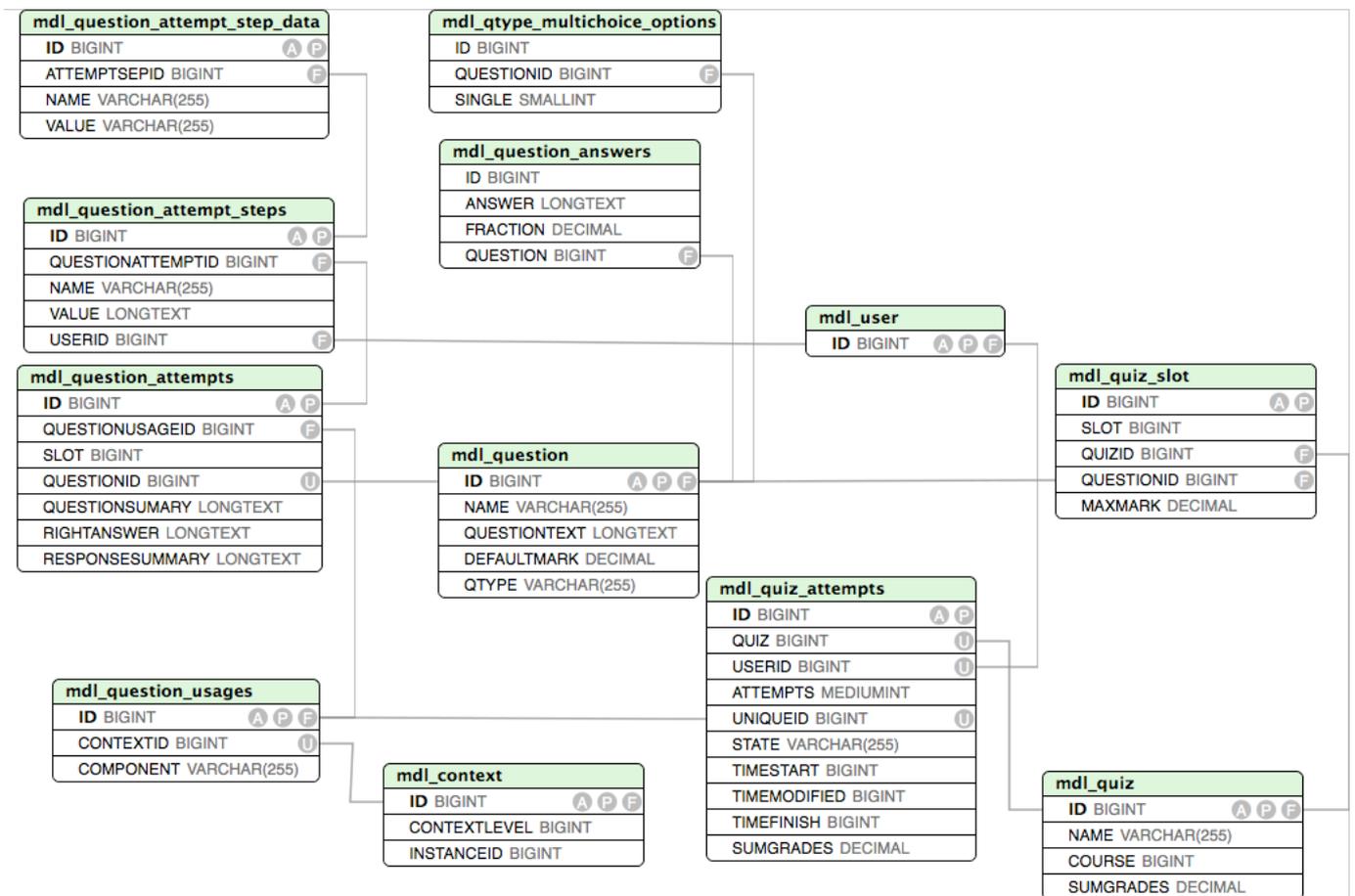


Figura 19 – Relación Base de Datos Sección Cuestionarios.

En este proyecto también consta de una base de datos interna en la que se almacenan las respuestas en el momento de pasar a la siguiente o anterior pregunta de un determinado cuestionario, esta base de datos se consultará cuando se sale de un intento o se envían las respuestas para ser evaluadas. Esta base de datos esta compuesta solamente por una tabla denominada **AnswerQuestion** y contiene los campos:

- *Id*: clave primaria de la tabla.
- *Responsesummary*: contiene el índice de la respuesta seleccionada valido para preguntas de tipo opción múltiple.
- *ResponsesummaryText*: contiene el texto de la respuesta aportada o seleccionada por el usuario.
- *Idquiz*: contiene el identificador del cuestionario al que pertenece la pregunta.
- *Qtype*: almacena el tipo de pregunta.
- *Single*: en las preguntas de opción múltiple informa si es de una sola respuesta (1) o de varias (0).

AnswerQuestion	
id	INTEGER (A P)
RESPONSESUMMARY	INTEGER
RESPONSESUMMARYTEXT	TEXT
IDQUIZ	INTEGER
QTYPE	TEXT
SINGLE	INTEGER

Figura 20 - Tabla de la base de datos interna.

Por otra parte se va a mostrar la base de datos de Moodle en relación a la sección de foros de la aplicación, en la figura 21 se observan todas las tablas que están implicadas.

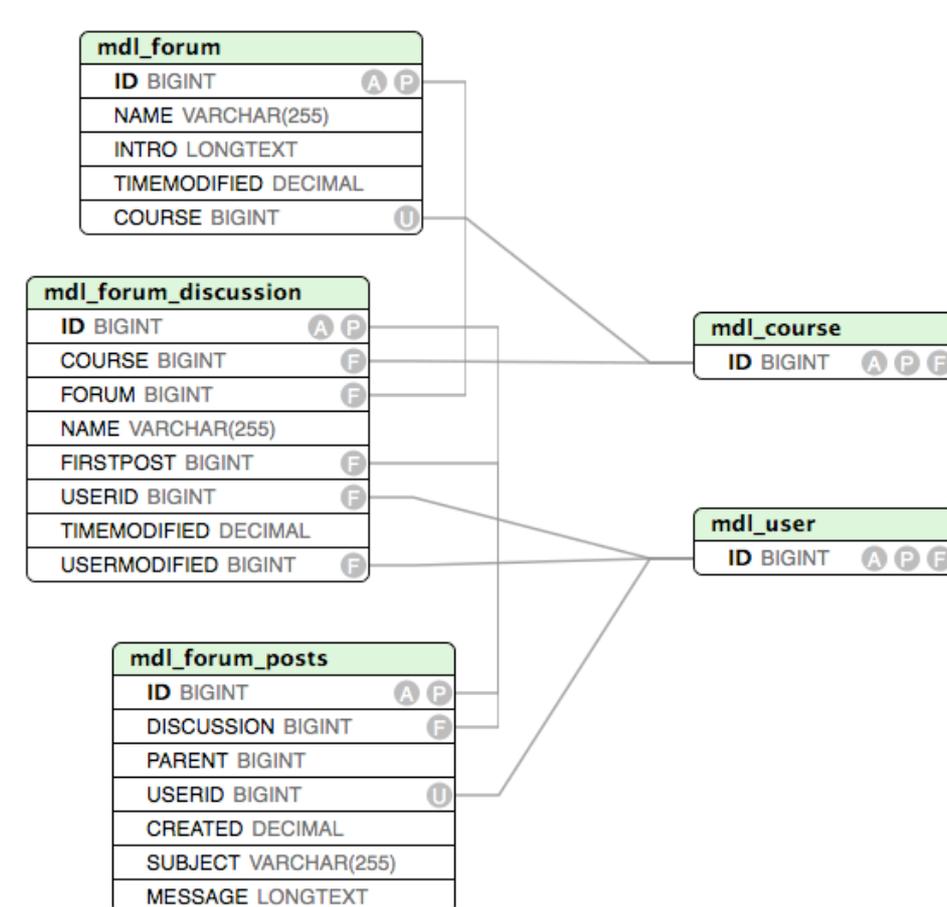


Figura 21 - Relación Base de Datos Sección Foros.

Los datos que tienen la conexión con las tablas mdl\_user y mdl\_course son datos que se van arrastrando de vistas anteriores y a este nivel de la aplicación no implica hacer consultas a estas tablas, por lo que no están desglosadas, explicando sus campos. La explicación de los campos de las tablas anteriormente nombradas y algunas más se hace en la versión que ha sido integrada (Antolín Blanco, 2015) en este proyecto

## 4. CONEXIÓN DE LA APLICACIÓN MÓVIL CON EL SERVIDOR WEB

### 4.1 INTRODUCCIÓN

Como ya se indicó a lo largo de este documento, las conexiones necesarias entre la aplicación móvil y el servicio web, se va a llevar a cabo mediante un servicio RESTful. De esta manera a través de conexiones HTTP se podrán realizar comunicaciones en las que cada mensaje va a contener toda la información necesaria para comprender una petición o una respuesta.

El formato elegido para la representación de los mensajes ha sido JSON, al ser mucho más sencillo en cuanto a estructura que XML, proporcionando más flexibilidad a la hora de representar cualquier estructura de datos o añadir nuevos campos.

### 4.2 EL PROTOCOLO HTTP

El protocolo HTTP (*HyperText Transfer Protocol*) es un protocolo de la capa de aplicación que se emplea para la transferencia de información entre sistemas, de manera clara y rápida. Es el protocolo que utiliza el *World-Wide Web* desde 1990.

HTTP se basa en un paradigma de peticiones y respuestas. Un cliente envía una petición en forma de método, una URI y una versión del protocolo, seguida de los modificadores de la petición, información sobre el cliente y adicionalmente un posible contenido. Por su parte, el servidor envía una respuesta con una línea de estado que incluye la versión del protocolo y un código de éxito o error, seguido de la información del servidor y un posible contenido.

Generalmente, es el cliente el que inicia la comunicación HTTP, y ésta consiste en la petición de un recurso del servidor. Éste es el caso que ocupa a este proyecto.

La sintaxis de una petición es la siguiente:

“http://” dirección [ “:” puerto ] [ path ]

Donde dirección es el nombre de un dominio de Internet o una dirección IP, puerto es el número que indica el puerto al que se envía la petición y path hace referencia al recurso a que se quiere acceder. Por defecto, se supone que se accede al puerto 80 y si no se indica el path será “/”.

A la hora de realizar las peticiones se ha de incluir el método que se aplica al recurso. Los principales métodos que se utilizan son los que se explican a continuación (Herrero Herrero, 2013).

### **Método GET**

Este método requiere la devolución de información al cliente identificada por la URI. Si la URI se refiere a un proceso que produce información se devuelve la información, y no la fuente del proceso.

### **Método HEAD**

Se trata de un método similar a GET, con la diferencia de que el servidor no tiene que devolver el contenido sino solo las cabeceras. Estas cabeceras deberían ser las mismas que las que se devolverían empleando el método GET.

### **Método POST**

El método POST se utiliza para realizar peticiones en las que el servidor acepta el contenido de la petición como nuevos parámetros del recurso pedido. Este método se creó para cubrir funciones como enviar mensajes a grupos de usuarios, dar un bloque de datos como resultado de un formulario a un proceso o añadir nuevos datos a una base de datos.

La función que lleve a cabo el método POST está determinada por el servidor, y depende de la URI de la petición. El resultado de la acción realizada por el método POST puede ser un recurso que no sea identificable mediante una URI.

### **Método PUT**

Este método permite guardar el contenido de la petición en el servidor bajo la URI de la misma. Si esta URI ya existe, el servidor considera que la petición es una actualización del recurso. Si no existe, el servidor puede crear el recurso con esa URI.

La diferencia entre POST y PUT se halla en el significado de la URI. En el caso de POST, la URI identifica el recurso que va a manejar el contenido de la petición, mientras que en el caso de PUT la URI identifica el contenido de la petición.

### **Método DELETE**

Este método se emplea para que el servidor borre el recurso indicado por la URI de la petición. Aunque la respuesta sea satisfactoria, no se garantiza al cliente que la operación de borrado haya tenido éxito.

De todos estos métodos, para la realización de este proyecto basta con utilizar GET y POST, tal como se verá más adelante.

## **4.3 EL FORMATO DE DATOS JSON**

JSON (*JavaScript Object Notation*) es un formato ligero de intercambio de datos. Se trata de un formato simple, fácil de leerlo y escribirlo para humanos e igualmente fácil de interpretarlo y generarlo para máquinas. JSON es un formato de texto completamente

independiente del lenguaje, pero utiliza convenciones ampliamente conocidas por programadores de diferentes lenguajes de programación. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos. JSON está constituido por dos estructuras:

- Una colección de pares nombre-valor. Es lo que se conoce como un objeto.
- Una lista ordenada de valores. Es lo que se conoce como un array.

Estas estructuras son universales, es decir, todos los lenguajes de programación las soportan de una forma u otra, por lo que es bastante interesante que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

### Objeto JSON

Un objeto es un conjunto desordenado de pares nombre-valor. Comienza con el carácter “{” y termina con el carácter “}”. Cada nombre es seguido por el carácter “:” y cada par nombre-valor está separado por el carácter “,”. En la Figura 28 se puede ver una representación de los objetos JSON (JSON, s.f.).

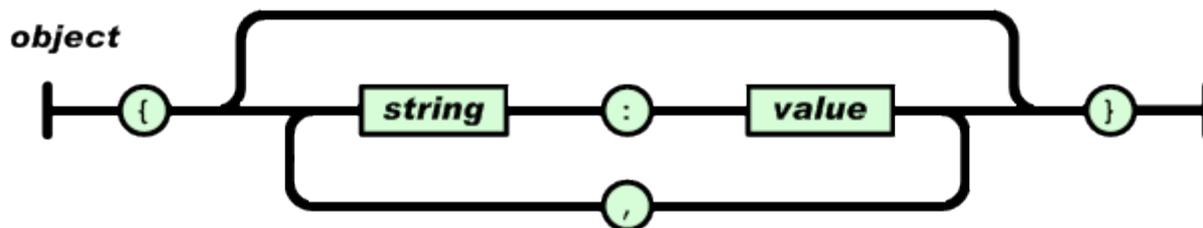


Figura 22 - Objeto JSON

### Array JSON

Un array es una colección de valores. Comienza con el carácter “[” y termina con el carácter “]”. Los valores de esta lista se separan por el carácter “,”. En la Figura 29 se puede ver una representación de los arrays JSON.

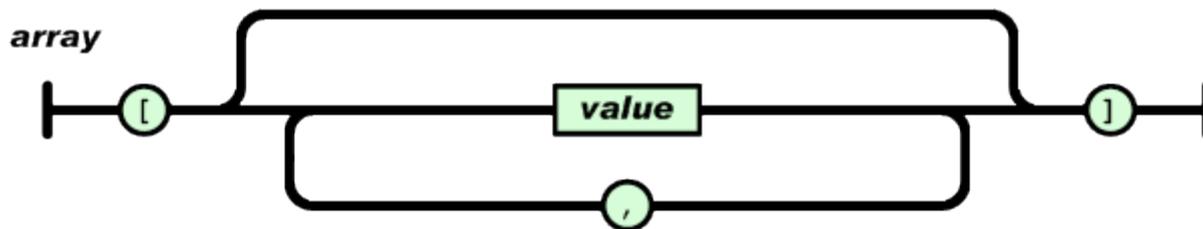


Figura 23- Array JSON

## Valor JSON

Un valor JSON puede ser una cadena de caracteres con comillas dobles, un número, una constante *true*, *false* o *null*, un objeto o un array. En la Figura 30 se puede ver una representación de los valores JSON.

### value

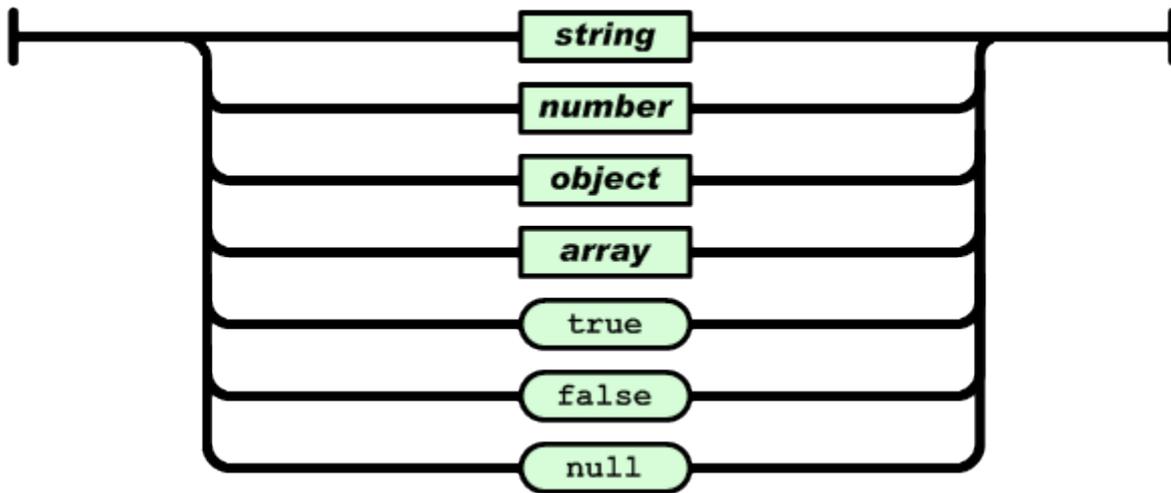


Figura 24 - Valor JSON

Un ejemplo de cómo se estructuran los datos en JSON podría ser el siguiente:

```
{
  - materias: [
    - {
      id: "1",
      nombre: "Dispositivos Moviles"
    },
    - {
      id: "2",
      nombre: "Computo Integrado"
    },
    - {
      id: "3",
      nombre: "Sistemas Distribuidos y paralelos"
    },
    - {
      id: "4",
      nombre: "Modelado y Simulado de sistemas Dinamicos"
    }
  ]
}
```

Nótese que el contenido dentro de los corchetes “[” y “]” es un array JSON, mientras que las llaves “{” y “}” delimitan objetos JSON.

## **4.4 DIAGRAMAS DE SECUENCIA**

A continuación se va a describir los diagramas de secuencia de este proyecto, estos serán los correspondientes a los apartados de foros y cuestionarios de la aplicación, por lo que se debe tener en cuenta los pasos previos que se deben haber realizado en la aplicación hasta llegar a ellos, estos pasos a los que se refiere son el proceso de autenticación, la visualización de cursos y el de etiquetas que fueron explicados en el proyecto previo a la integración (Antolín Blanco, 2015).

Para describir los diagramas de secuencia de este proyecto, es necesario involucrar a tres actores. En primer lugar, se tiene la aplicación *Android*, la cual actuará como cliente y es la encargada por tanto de realizar las peticiones al servidor. En segundo lugar, se tiene el servidor, dentro del cual se va a poder diferenciar claramente, por un lado los recursos que están alojados en el servidor (ficheros PHP que atenderán las peticiones de los clientes) y por otro lado la base de datos de *Moodle*, que almacena toda la información necesaria. A continuación se explicará cada una de las partes en que se ha dividido el proyecto.

### **4.4.1 FOROS**

A este punto se ha llegado previa selección del curso y visualización de un menú en el cual tenemos la opción de visualizar los foros asociados a el curso en el que se encuentra.

El proceso de visualización de los foros viene representado en el siguiente diagrama:

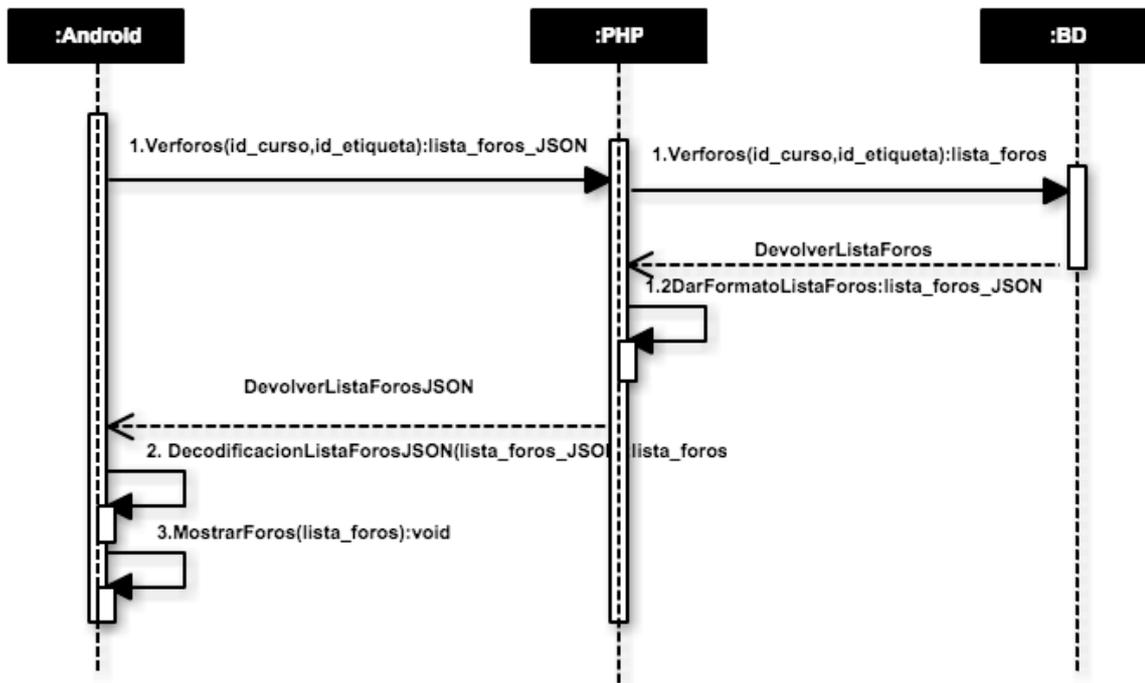


Figura 25 - Diagrama Foros

El cliente *Android* realiza una petición al servidor. Lo que se demanda es el identificador, nombre y introducción que se añade a modo de aclaración sobre la utilización del foro.

Estos datos son los correspondientes a los parámetros id, name e intro de la tabla *mdl\_forum*. Para poder realizar esta petición el cliente *Android* utiliza el método POST para pasar el identificador del curso y de la etiqueta que identifica al tema del curso al recurso PHP del servidor que se ha implementado para llevar a cabo esta función.

A continuación se realizan una serie de consultas a la base de datos para acabar obteniendo una lista de todos los foros asociados al tema identificado con la etiqueta.

Una vez obtenida esta lista de foros, en el recurso PHP se realizara un formateo de los datos de manera que tomen la apariencia JSON para enviar la respuesta al cliente *Android*. En consecuencia, en el cliente se recibirá esta lista formateada bajo JSON y se decodificara generando una nueva lista dispuesta para ser presentada por pantalla.

En esta lista, como se comentó al comienzo de este apartado, se visualizará el nombre e introducción de los foros del tema consultado y respecto al identificador de los foros se mostrará en la pantalla pero de modo oculto. Los identificadores, en este caso de los foros, pasarán de pantalla en pantalla en modo oculto ya que en posteriores peticiones al servidor puede ser necesario. Esto es lo que se ha hecho así como con los identificadores de los cursos y etiquetas como con el identificador del usuario autenticado.

### Visualización de discusiones

En la visualización de discusiones se sigue el mismo patrón que para la de foros, con la diferencia que el parámetro que envía el cliente *Android* por el método POST es el identificador del foro.

### Visualización de mensajes de las discusiones

En la visualización de mensajes de las discusiones se sigue el mismo patrón que para la de foros, con la diferencia que el parámetro que envía el cliente *Android* por el método POST es el identificador de la discusión.

### Creación de respuesta a un mensaje

El cliente *Android* realiza una inserción de un mensaje de discusión con su título a la base de datos, este mensaje será en respuesta a otro de esa discusión, el cliente enviará al servidor, por el método POST, el identificador del mensaje al cual se esta contestando, ya que este dato se guarda en la tabla mdl\_forum\_post.parent, haciendo referencia al padre del que procede el mensaje. Otros datos que envía el cliente *Android* al servidor para llevar a cabo este cometido son el identificador de la discusión a la cual va a pertenecer el mensaje, el identificador del usuario que esta escribiendo, el título y el contenido del mensaje.

Todos los datos comentados son recibidos por el recurso PHP implementado para llevar a cabo la inserción del mensaje en la base de datos de *Moodle*.

A continuación, se realizarán las consultas necesarias.

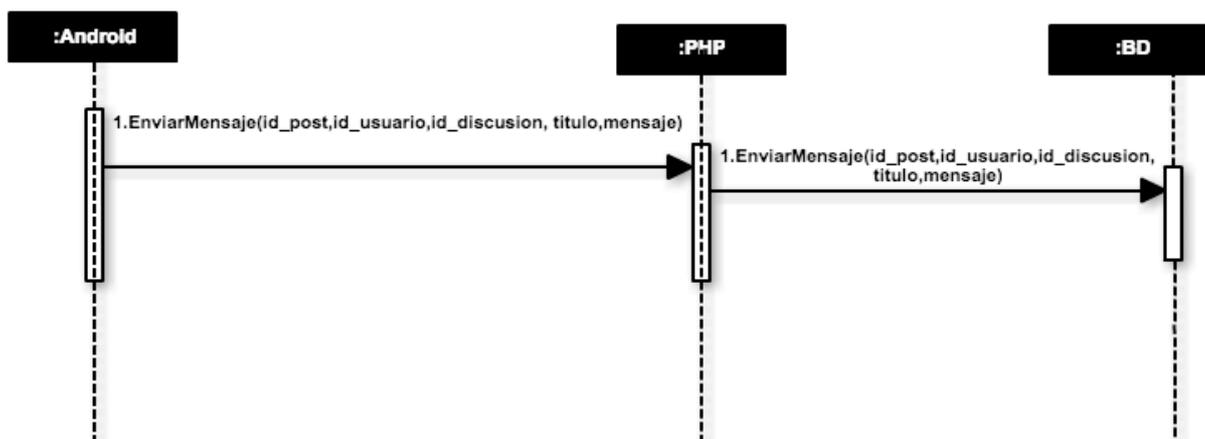


Figura 26 - Diagrama Envío De Mensajes de discusión.

### Creación de una nueva discusión

De forma similar a la creación de mensajes sucede para la creación de discusiones a continuación sin hacer mas explicación se muestra el diagrama de secuencia.

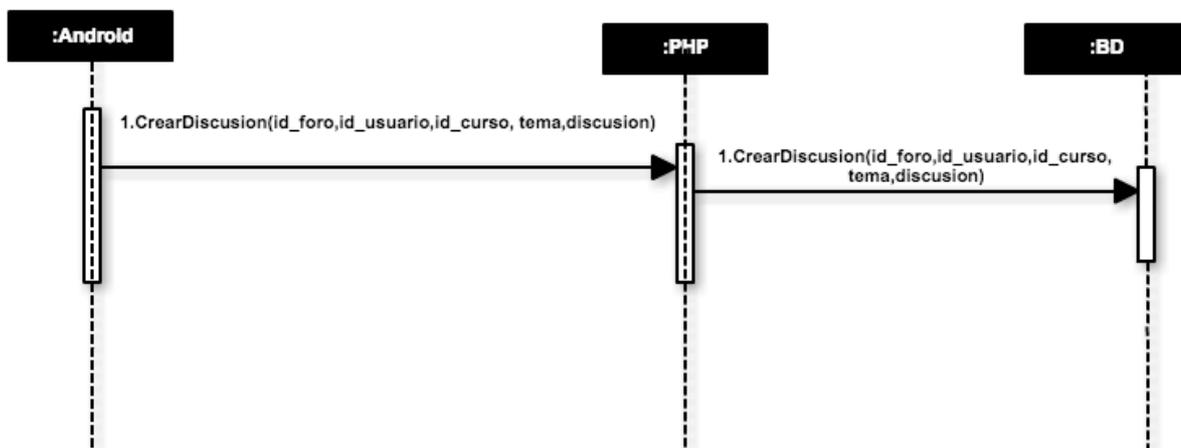


Figura 27 – Diagrama Creación de Nueva Discusión.

## 4.4.2 CUESTIONARIOS

### Visualización de cuestionarios

A este punto se ha llegado previa selección del curso y visualización de un menú en el cual tenemos la opción de visualizar los cuestionarios asociados a el curso en el que se encuentra.

El proceso de visualización de los cuestionarios viene representado en el siguiente diagrama:

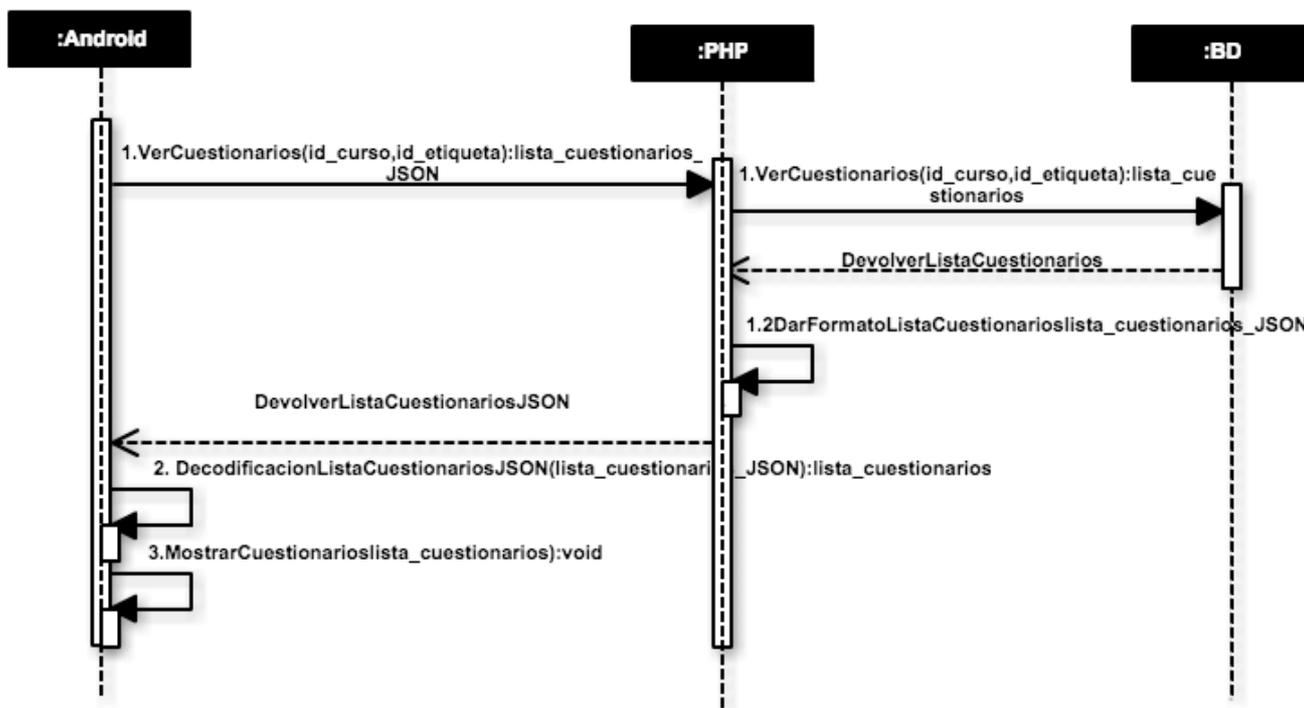


Figura 28 - Diagrama Cuestionarios

El cliente *Android* realiza una petición al servidor. Lo que se demanda es el nombre, el identificador y número de preguntas que contiene el cuestionario.

Estos datos son los correspondientes a los parámetros `name`, `id` de la tabla `mdl_quiz` y para saber las preguntas que contiene lo que se consulta a la base de datos es el número de slots que tiene ese cuestionario en la tabla `mdl_quiz_slots`, estos slots no son más que una enumeración que se hace a las preguntas que forman un cuestionario a la hora de crearlo.

Para poder realizar esta petición el cliente *Android* utiliza el método `POST` para pasar el identificador del curso y de la etiqueta que identifica al tema del curso al recurso PHP del servidor que se ha implementado para llevar a cabo esta función.

A continuación se realizan una serie de consultas a la base de datos para acabar obteniendo una lista de todos los cuestionarios asociados al tema identificado con la etiqueta.

Una vez obtenida esta lista de cuestionarios, en el recurso PHP se realizará un formateo de los datos de manera que tomen la apariencia JSON para enviar la respuesta al cliente *Android* de forma semejante a como se hace para el apartado anterior de la visualización de los foros. En consecuencia, en el cliente se recibirá esta lista formateada bajo JSON y se decodificará generando una nueva lista dispuesta para ser presentada por pantalla.

En esta lista, se visualizará el nombre del cuestionario y el número de que preguntas que lo forma, por supuesto del tema consultado y respecto al identificador de los cuestionarios se

mostrará en la pantalla pero de modo oculto. Los identificadores de los cuestionarios como ya se explico para los foros pasará de pantalla en pantalla de modo oculto para el posible uso en posteriores peticiones al servidor.

### **Visualización de intentos a cuestionarios**

Tras visualizar los cuestionarios disponibles y pulsar en el deseado, se procede a la visualización de intentos anteriores, tanto los que han sido terminados como si hay algún intento abierto y sigue en curso.

El proceso de visualización de los intentos a un mismo cuestionario por un usuario, tienen un diagrama igual que en los apartados anteriores (foros y cuestionarios) con algunas variaciones en los datos enviados por POST del cliente *Android* en la petición PHP y en la lista en formato JSON que recibe, estas variaciones se van a comentar a continuación.

Respecto al cliente *Android* pasará al recurso PHP implementado por el método POST el identificador del cuestionario en cuestión y el identificador del usuario que está autenticado.

Lleva a que se realice una serie de consultas a la base de datos para acabar obteniendo una lista de todos los intentos al cuestionario.

El recurso PHP igual que en los anteriores apartados se encarga del formateo de los datos en formato JSON enviando la respuesta al cliente, estos pasos son exactamente igual que se ha explicado antes.

La lista que se visualiza, contiene el número de intento, el estado (en curso o terminado), la nota sobre el valor máximo, que es la suma de las notas máximas de cada pregunta, y por último también se verá la nota obtenida sobre diez. Los dos últimos campos tendrán valor y sentido cuando el estado del intento sea 'terminado' ya que es cuando el usuario envía las respuestas definitivas para ser evaluadas.

### **Generación de un intento nuevo o actualización de un intento en curso**

A este proceso se llega tras pulsar el botón de "Intentar" de la pantalla de intentos de cuestionarios, generando un nuevo intento si el último intento realizado ha sido terminado o bien actualizando el intento si el último realizado sigue en curso.

El cliente *Android* pasará al recurso PHP mediante el método POST el identificador de curso del cuestionario, del usuario y el numero de preguntas que contiene. El recurso PHP para llevar a cabo lo enunciado en el primer párrafo de este apartado.

Tras realizar esta acción se puede ver desde la plataforma para PC el intento en curso.

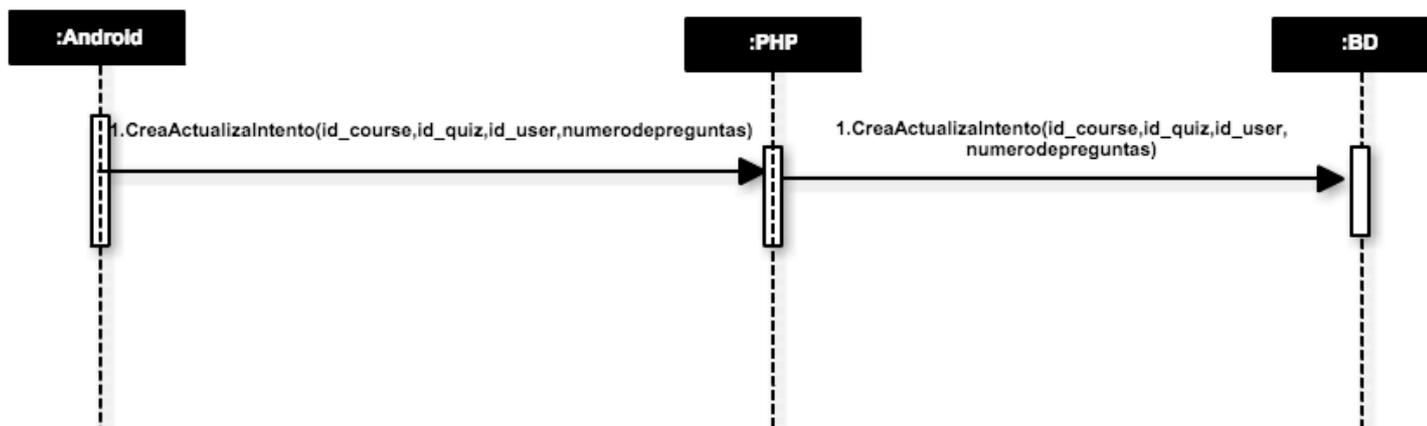


Figura 29- Diagrama Generación Intento

### Salir de un intento

En cada pregunta se tiene un botón que permite salir del intento, este botón al ser pulsado, obtiene la información de la base de datos interna en la que se ha ido guardando las respuestas en el momento que se pasa a la siguiente pregunta, y envía una por una las respuestas a las preguntas del cuestionario, dejando reflejado en la base de datos todos los valores que se hayan contestado, repito sin haber dado a enviar para proceder a la calificación del test, de tal manera que accediendo desde la plataforma para PC a este mismo intento podemos ver y modificar las respuestas realizadas en el último intento en el que se salió sin enviar.

A continuación se muestra el proceso que sigue el cliente *Android* para cada pregunta-respuesta.

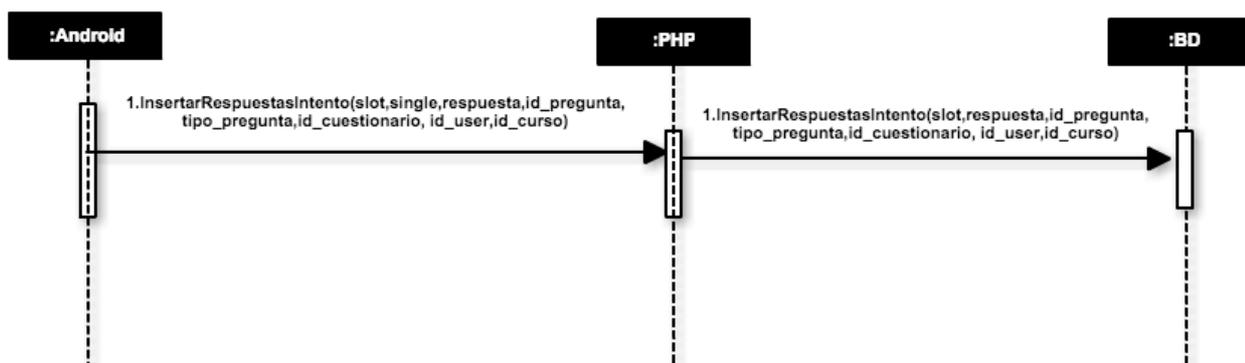


Figura 30 - Diagrama Salir Intento

Como se aprecia en la figura dos parámetros que el cliente envía al recurso PHP son 'slot' y 'single', el primero es la posición que ocupa la pregunta en el cuestionario, es decir, posición 1, 2, 3... y el segundo parámetro es el que indica en preguntas de opción múltiple si la respuesta es única o de respuesta múltiple.

### Enviar y terminar un intento

Esta acción se realiza al pulsar al botón para enviar las respuestas para calificar, en ese momento el cliente *Android* envían de igual modo que en el diagrama anterior, es decir, se escriben de una en una desde la base de datos interna a la base de datos de *Moodle*, a continuación se cambia de estado a terminado confirmando las respuestas finales y por último se solicita la nota al último recurso PHP.

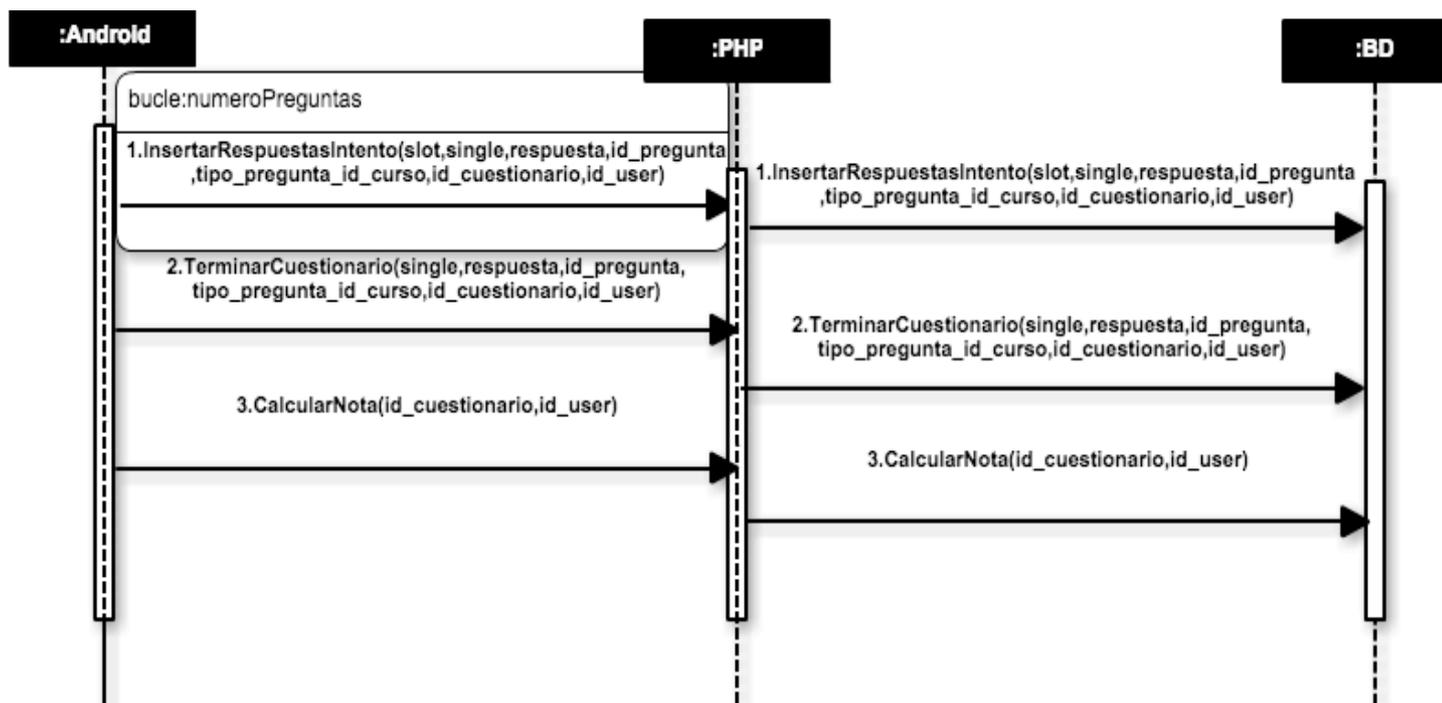


Figura 31 - Diagrama Enviar y Terminar Intento



# **CAPÍTULO VI.**

## **MANUAL DE USUARIO.**



Para el desarrollo de la interfaz se ha optado por seguir uno de los principios básicos de diseño del estilo global de *Moodle*: la simplicidad. Según este principio, se debe utilizar el mínimo de interfaz necesario para obtener la funcionalidad que se desee implementar. En este punto cabe recordar cuáles son los objetivos de este proyecto: permitir el acceso a foros de discusión con la creación de discusiones, envío y respuesta a mensajes de una discusión en concreto, permitir la visualización de cuestionarios, de intentos anteriores al cuestionario, la visualización de preguntas en lista, y por último visualizar las preguntas en su formato correspondiente con la opción de salir del intento o terminar y enviar las preguntas. Por último y no menos importante, el último objetivo consiste en la integración de la anterior versión (Antolín Blanco, 2015) que contiene el módulo de mensajería de *Moodle*, la opción de crear nuevo usuario, el proceso de autenticación, la visualización de cursos y la descarga de recursos.

En la gran mayoría los *layout* de la aplicación están formados en su gran mayoría por *ListViews*, los cuales van a permitir visualizar muy fácil e intuitivamente los foros, las discusiones, los mensajes de discusión, los cuestionarios, los intentos a cuestionarios o las preguntas del cuestionario. Por otro lado, la visualización de las preguntas de forma individual se realiza de forma dinámica desde JAVA en función del tipo de pregunta (verdadero o falso, respuesta corta, respuesta numérica y opción múltiple de una o varias respuestas correctas). Otro tipo de *layout* son Los menús que permitirán el acceso a las diferentes secciones están desarrollados con *buttons* con imágenes indicativas de cada sección, las cuales permitan una navegación lo más intuitiva posible.

A continuación, se va a proceder a detallar todas las funcionalidades de la aplicación final de este proyecto, a modo de manual de usuario.

## 1. PANTALLA DE REGISTRO DE NUEVO USUARIO

Si el usuario selecciona en la pantalla principal el rótulo de “Regístrese” accederá a esta pantalla. Dicha pantalla no es más que un formulario de registro, el cual una vez completado y tras pulsar e botón de “Registrar nuevo usuario” se lanzará un mensaje toast indicando que el nuevo usuario queda registrado a la espera de ser validado por el administrador de *Moodle*.



Figura 32- Pantalla Nuevo Usuario

## 2. PANTALLA DE INICIO

Cuando se inicia la aplicación, en primer lugar aparece una pantalla de autenticación de usuarios, tal y como podemos observar en la Figura 5. Esta pantalla consta por un lado de dos campos a rellenar: el nombre de usuario y la contraseña, más un botón de login el cual el usuario pulsará cuando crea que ha introducido los datos correctamente. Por otro lado debajo de este botón hay un rótulo que indica “Nuevo usuario”. Al pulsar sobre este último rótulo se accederá a una pantalla que permite el registro de nuevos usuarios.

Los campos a introducir en los campos antes mencionados se tratan de los datos que cada usuario utilice para acceder a su cuenta de Moodle vía web. Para acceder a la aplicación es necesario introducir el valor correcto de estos dos campos. En caso de introducir mal alguno de los dos campos, saltará un cuadro de diálogo de aviso indicando al usuario que debe introducir correctamente los datos. Si el usuario está registrado en la aplicación pero no confirmado por el administrador, de igual manera aparecerá otro cuadro de diálogo toast avisando de la incidencia.



Figura 33 - Pantalla de Inicio

## 2.1 PANTALLA MENÚ PRINCIPAL

El menú principal, al que se accede tras ser autenticado con éxito, consta de dos *Buttons*, uno que permite acceder a la mensajería representado por un sobre y otro que permite el acceso a los cursos o asignaturas en los que el usuario está matriculado, representado por el icono de una libreta y rotulado con la palabra asignaturas. Pulsando cualquiera de los dos iconos se accederá a una sección u otra. En la parte superior derecha, aparece el nombre y apellidos del usuario que ha sido autenticado y pulsando en él se podrá volver a la pantalla principal para autenticarse de nuevo.



Figura 34 - Pantalla Menú Principal

## 2.1.1 MENSAJERIA

### Pantalla menú mensajes

Al igual que en la pantalla menú principal, en este caso se muestran una serie de buttons que conforman un menú referido al apartado de mensajería de Moodle. En dicho menú se observan 5 opciones:

1. Mensajes no leídos: representado por un sobre cerrado, si el usuario pulsa sobre este botón accederá a una lista de mensajes no leídos.
2. Mensajes leídos: representado por un sobre abierto, si el usuario pulsa sobre este botón se accederá a una lista de mensajes previamente leídos. Tanto por a través de la aplicación móvil o a través de la plataforma *Moodle* del PC.
3. Enviados: representados por un sobre con una flecha, permite al usuario consultar una lista completa de los mensajes que ha enviado.
4. Nuevo mensaje: representado por una hoja en blanco, permite al usuario acceder a una lista de destinatarios a los cuales puede enviar un nuevo mensaje.
5. Menú principal: Permite al usuario volver a la pantalla de menú principal.



Figura 35 – Pantalla Menú Mensajes

### Pantalla de mensajes no leídos

Al seleccionar en el menú de mensajes la opción de mensajes no leídos, el usuario visualizará únicamente aquellos mensajes que no se hayan leído, o bien a través de la aplicación móvil o a través de la plataforma para ordenador. Si el usuario no dispone de ningún mensaje no leído, en esta pantalla aparecerá un mensaje del tipo “No tiene mensajes nuevos”.

En la parte inferior el usuario dispone de un botón para vaciar la bandeja de entrada como se puede observar en la Figura 35, al seleccionarlo se accederá a la pantalla de la Figura 14, y los mensajes que aparecían como no leídos al volver a entrar en esta pantalla ya no aparecerán y se encontrarán en el apartado de mensajes leídos.

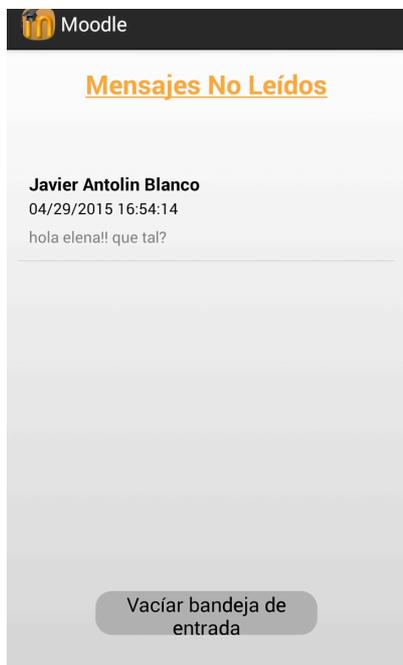


Figura 36 - Pantalla Mensajes No Leídos

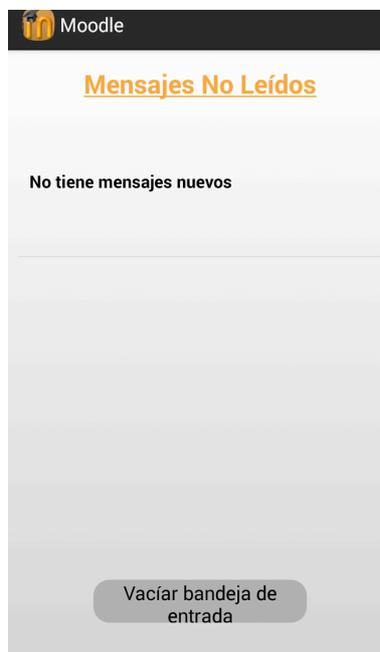


Figura 37 - Pantalla Mensajes No leídos vacía

### Pantalla mensajes leídos

Es igual que la pantalla de mensajes no leídos que se observaba en la Figura 35, pero en este caso aparecerán los mensajes en los que el usuario sea destinatario pero que ya hayan sido leídos por el mismo. En ambos casos cabe destacar que si el usuario selecciona cualquier mensaje (tanto en la pantalla de leídos como no leídos), se accederá a una vista individual como se observa en la Figura 37 a través de la que se podrá contestar a dicho mensaje. Si seleccionamos la opción de responder accederemos a la pantalla escribir nuevo mensaje. En la que el receptor por defecto del mensaje que va a escribir el usuario será el emisor del mensaje que está seleccionado.

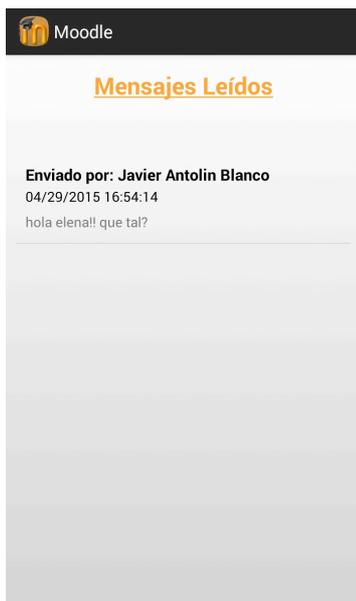


Figura 38 – Pantalla Mensajes Leídos

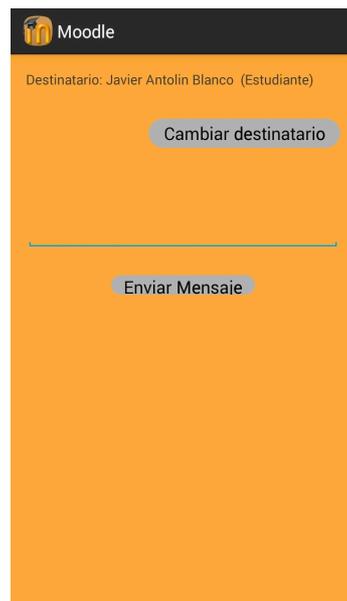


Figura 39 – Pantalla Responder Mensaje

### Pantalla enviar mensaje nuevo

Al seleccionar esta opción en el menú de mensajería, el usuario verá una lista formada por aquellas personas a la que tiene permiso para enviar un mensaje.

Esta lista de nombres la forman, alumnos y profesores que coincidan con el usuario en alguna asignatura, a continuación se escribe el mensaje en el cuadro de texto y se envía pulsando el botón de enviar, como se muestra en las figuras siguientes.

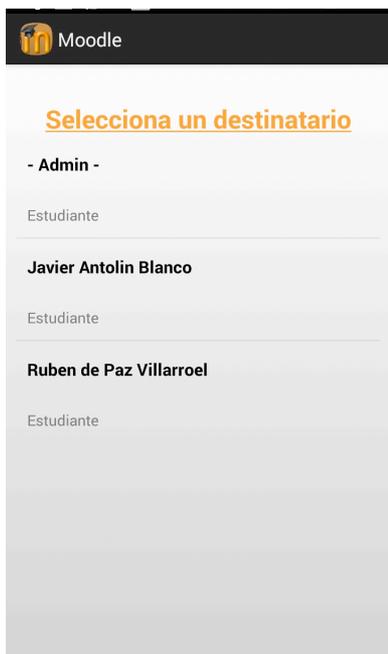


Figura 40 – Pantalla Selección destinatario

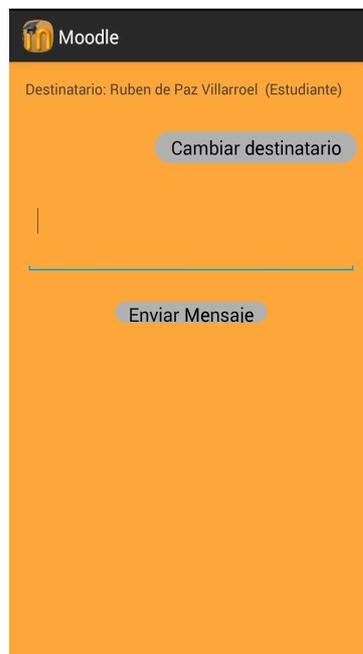


Figura 41 – Pantalla Escribir Mensaje



Figura 42 – Pantalla Escribiendo Mensaje

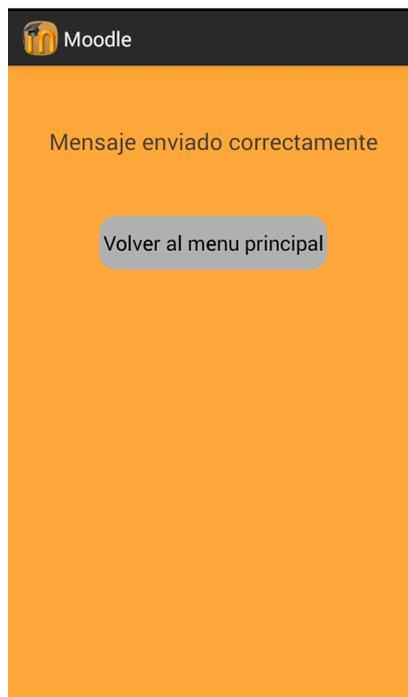


Figura 43 – Pantalla Mensaje Enviado

### Pantalla mensajes enviados

Es una pantalla del mismo tipo que la de recibidos y no leídos, está formada por un *listview* y permite ver los mensajes que ha enviado el usuario, junto a información como la fecha y el receptor, pulsando sobre un determinado mensaje el usuario tendrá acceso a una vista individualizada del mismo. En la siguiente figura se muestra el mensaje enviado en las pantallas anteriores.

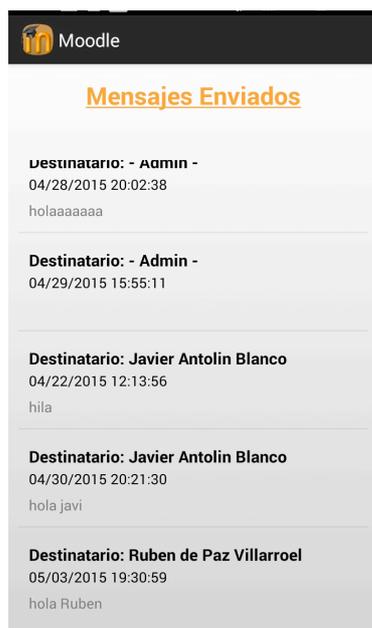


Figura 44 – Pantalla Mensajes Enviados

## 2.1.2 CURSOS O ASIGNATURAS

### Pantalla de cursos o asignaturas

Una vez autenticado en la aplicación, y tras pulsar el icono asignaturas en la pantalla de menú principal aparecerá un listado de los cursos o asignaturas en que el usuario esté matriculado, además se muestra el usuario autenticado y la opción de volver a la pantalla principal para volver a autenticarse, por otro lado se muestra la ruta donde se encuentra en este caso, se sitúa dentro de la sección en asignaturas como bien se visualiza en la figura siguiente. Simplemente seleccionando uno de ellos se podrá acceder a las secciones en que está dividido el curso, de la manera que se observa en la Figura 44.

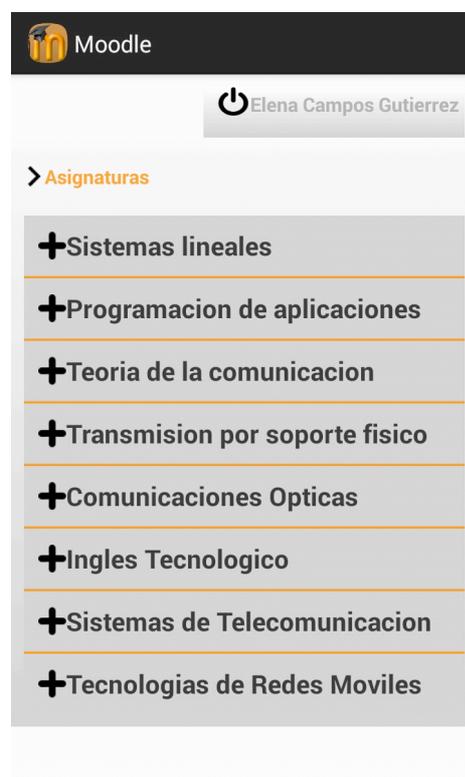


Figura 45 - Pantalla Asignaturas

### Pantalla secciones

De la misma manera que en la pantalla de cursos, en esta pantalla se mostrará un listado de las secciones en que está dividido el curso que se ha seleccionado en la pantalla anterior, además se sigue visualizando el usuario autenticado, la ruta que nos indica donde se encuentra el usuario y un botón que permite volver al menú principal. Al seleccionar una sección se accederá al menú de opciones que permitirá acceder a los foros, cuestionarios, recursos o volver al menú principal.

En este caso no siempre tiene por qué haber foros, cuestionarios o recursos disponibles en una sección. Atendiendo a la Figura 45, se observa cómo hay disponibles varios temas y pulsando accederemos al menú mencionado anteriormente.



Figura 46 - Pantalla Secciones

### Pantalla opciones

En esta pantalla se visualiza el usuario autenticado, la ruta que nos indica en que sección se encuentra el usuario y cinco botones, el primero valido para volver al menú inicial (mensajería y asignaturas), el segundo permite el acceso a foros, el tercero el acceso a los cuestionarios, el cuarto a los recursos y por último el botón de etiquetas que lleva a los temas de la asignatura.

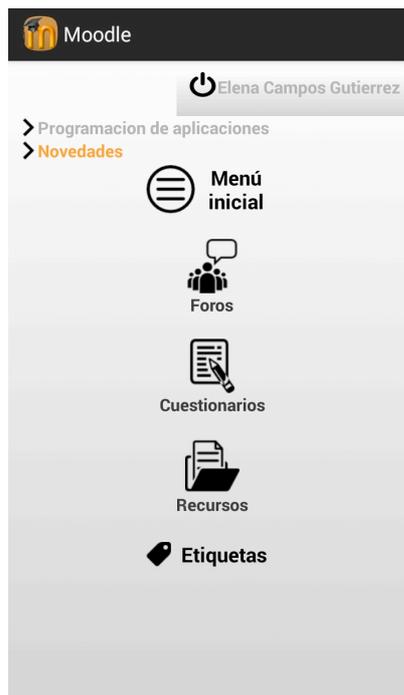


Figura 47- Pantalla Menú Opciones

### Pantalla foros

En esta pantalla se visualiza el nombre de usuario autenticado, la ruta en la que se encuentra dicho usuario, el botón de etiquetas con la idéntica función explicada en la pantalla anterior, y por último el listado de foros asociados a la etiqueta en la que se encuentra. En este listado se muestra el nombre del foro y una breve introducción de este.

En la siguiente figura se observa lo mencionado anteriormente, aunque en este caso el numero de foros del listado solo sea uno.

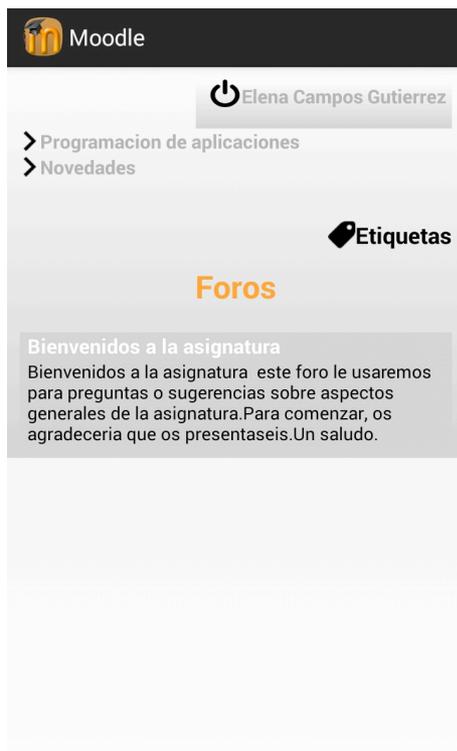


Figura 48- Pantalla Foros

### Pantalla discusiones

Al igual que en pantallas se visualiza el usuario autenticado, la ruta en la que se encuentra y el botón de etiquetas.

Se tiene un botón para crear un nuevo tema de discusión y el listado de las discusiones, mostrando el tema de discusión y quien lo ha creado, de tal manera que pulsando en alguna de ellas accederemos a los mensajes que contenga.

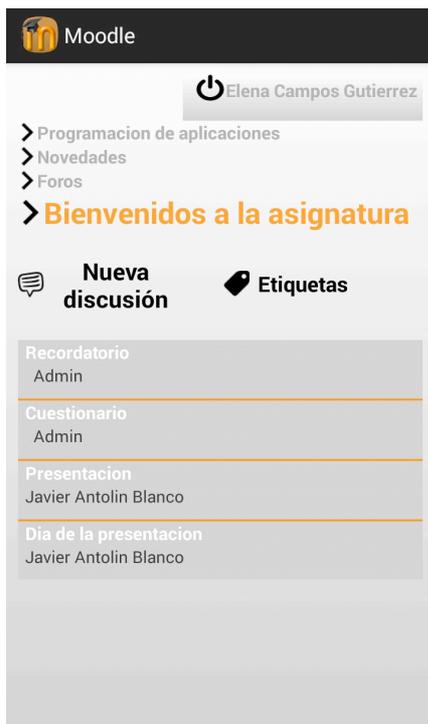


Figura 49- Pantalla Discusiones

### Pantalla nueva discusión

A esta pantalla se accede tras pulsar el botón anterior “Nueva discusión”. De igual manera se tiene el usuario y la ruta en la que se encuentra, acompañado de dos cuadros de texto en los que se espera que se añada el tema y mensaje de discusión y por último el botón de enviar al foro la nueva discusión.

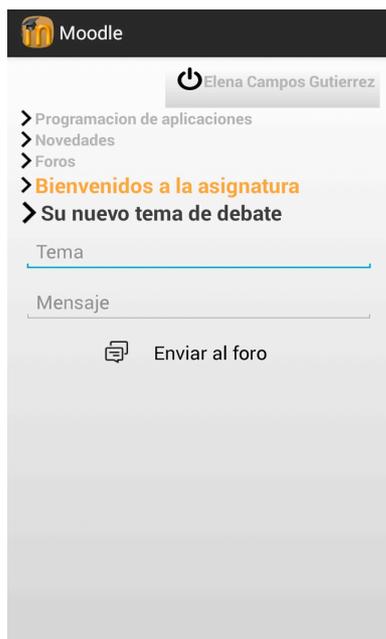


Figura 50 - Pantalla Nueva Discusión

## Pantalla mensajes de discusión

Esta pantalla es muy similar a la pantalla de discusiones, excepto en que se sustituye el botón de nueva discusión por el de nuevo mensaje y la lista de visualización de mensajes que cada ítem muestra el tema del mensaje, quién lo ha escrito, la fecha, y el contenido del mensaje.

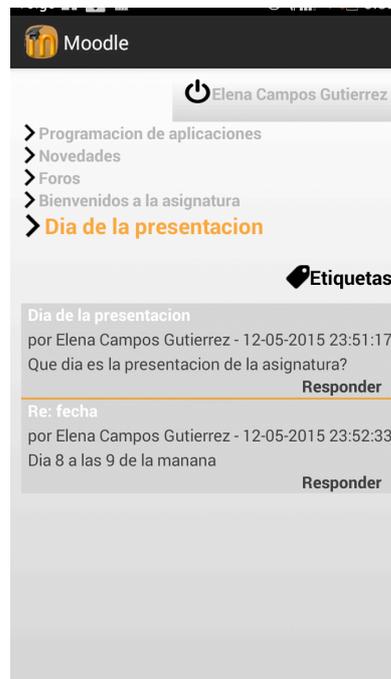


Figura 51 – Pantalla Mensajes de Discusión

## Pantalla respuesta a un mensaje

La pantalla de nuevo mensaje se obtendrá tras pulsar el botón mencionado anteriormente denominado “Nuevo mensaje” y la pantalla de respuesta a un mensaje sucederá tras pulsar un mensaje de la lista.

Ambas pantallas, contienen dos cuadros de texto que esperan el título y contenido del mensaje, tras rellenas estos campos se debe pulsar en el botón “Responder” para enviar el mensaje al foro.

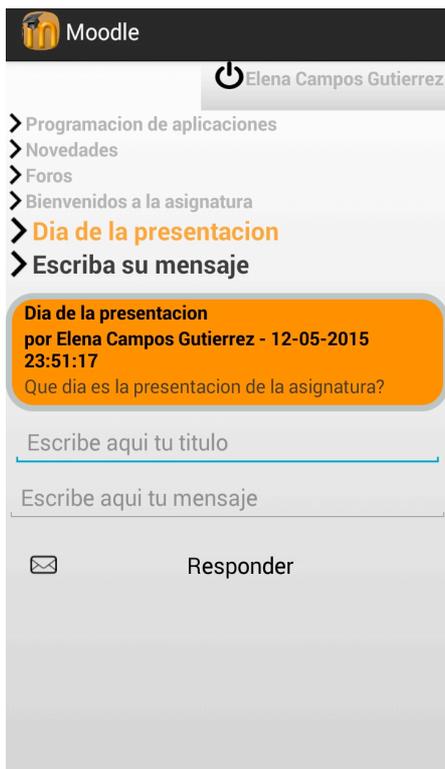


Figura 53 – Pantalla Respuesta a un Mensaje

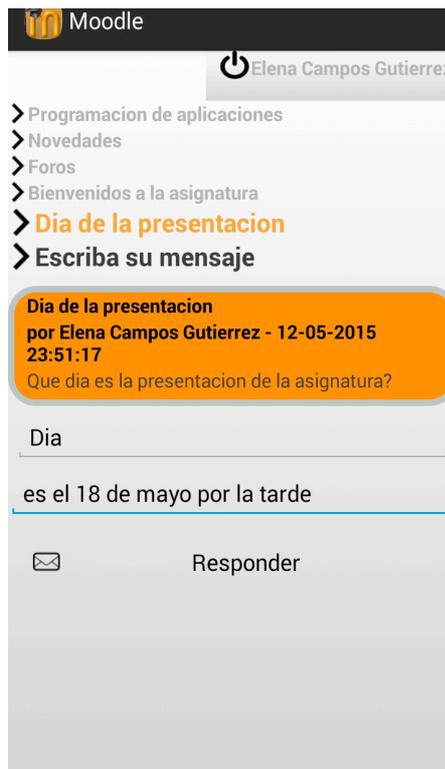


Figura 52 –Pantalla Escribiendo respuesta

### Pantalla de cuestionarios

A esta pantalla se llega desde el menú de opciones pulsando en el botón de “cuestionarios” .

Esta pantalla guarda gran similitud con la pantalla de foros, excepto en la lista que se muestra que contiene el nombre de cuestionarios y el número de preguntas que lo compone.

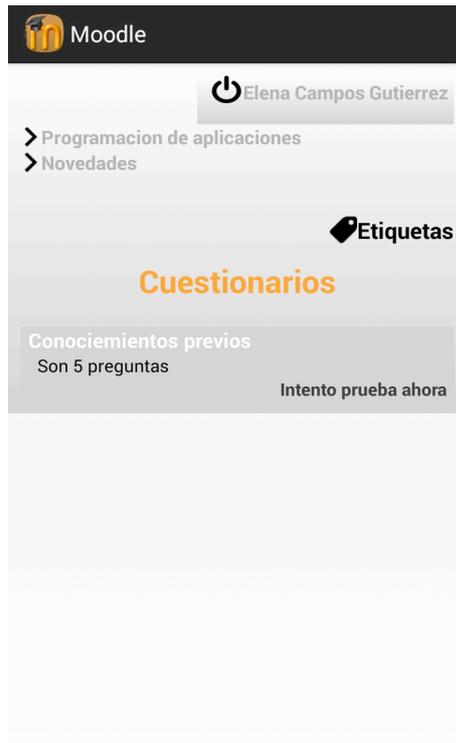


Figura 54 – Pantalla Cuestionarios

### Pantalla de intentos

Incluye como es habitual el usuario autenticado, la ruta que indica donde se sitúa y el botón de etiquetas que devuelve a la lista de temas del curso en el que está.

Se muestra un listado de los intentos anteriores, estén en estado terminado o en curso, si el estado es terminado se mostrará la nota sobre diez y la nota sobre la suma de las notas máximas de las preguntas que lo forman.



Figura 55 – Pantalla de Intentos

### Pantalla de preguntas

Se visualizan el nombre y el contenido de las preguntas que forman el cuestionario, pulsando en cada una de ellas, se podrá ver el nombre y el contenido junto con el formato correspondiente para la respuesta. Además, incluye como en las anteriores pantallas, el usuario autenticado, la ruta que indica dónde se sitúa y el botón de etiquetas que devuelve a la lista de temas del curso en el que está.

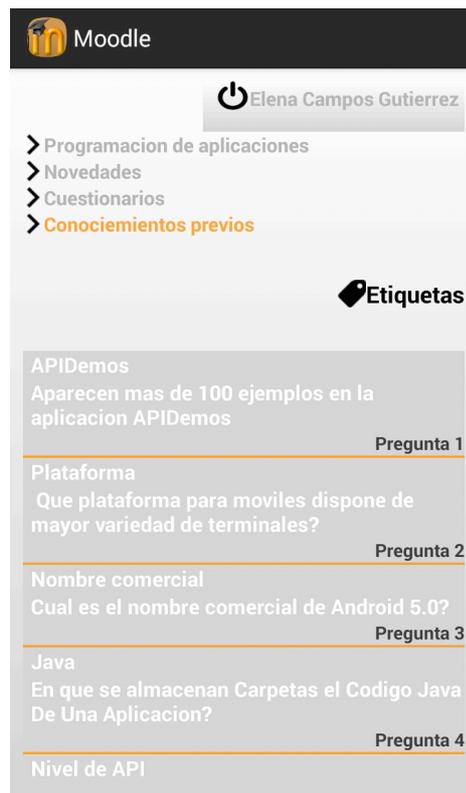


Figura 56 - Pantalla Preguntas de Cuestionario

### Pantalla dinámica de preguntas

Esta pantalla se adapta a cada tipo de pregunta, como pueden ser de verdadero o falso, opción múltiple con respuesta múltiple o simple, de respuesta corta o respuesta numérica.

Esta pantalla irá acompañada de un botón que permite salir del intento, guardando en la base de datos del teléfono y de Moodle las respuestas respondidas, de tal forma que si entramos en el intento desde la plataforma para PC se visualizará las respuestas, por otro lado también se cuenta con flechas en ambos sentidos (excepto en la primera y última pregunta) para desplazarse por las preguntas. Además, otro botón que se tiene es el de terminar y enviar las respuestas del formulario para su calificación y volver a la pantalla de intentos con la calificación oportuna, este botón solo aparece si el usuario se encuentra en la última pregunta.

A continuación, se muestra un ejemplo de cada tipo de pregunta y la visualización tanto de las flechas como del botón terminar en los casos señalados.

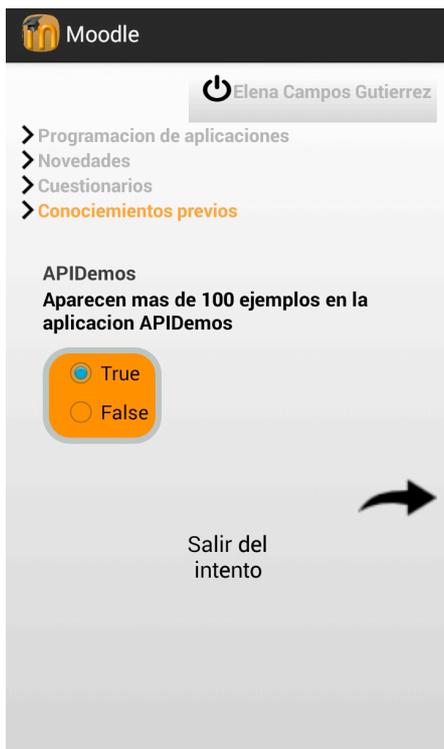


Figura 57- Pantalla Verdadero o falso

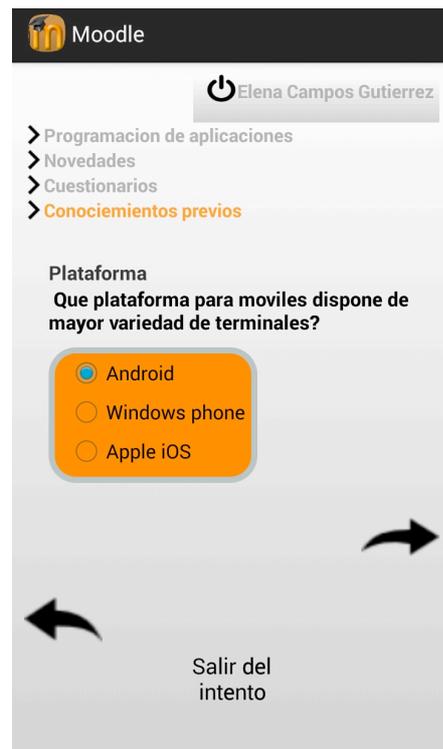


Figura 58 - Pantalla con Respuestas Simple

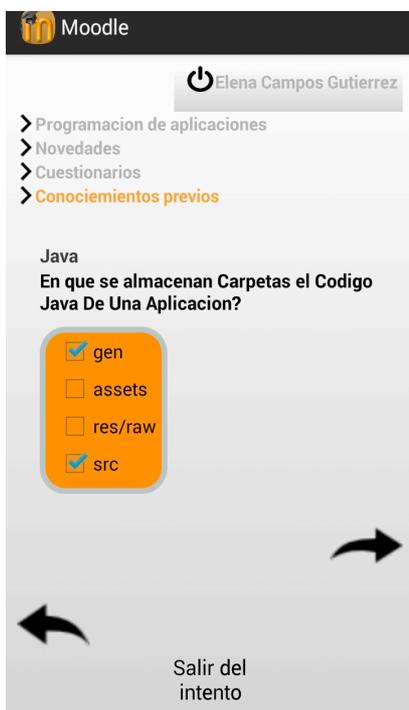


Figura 59- Pantalla con Respuesta múltiple

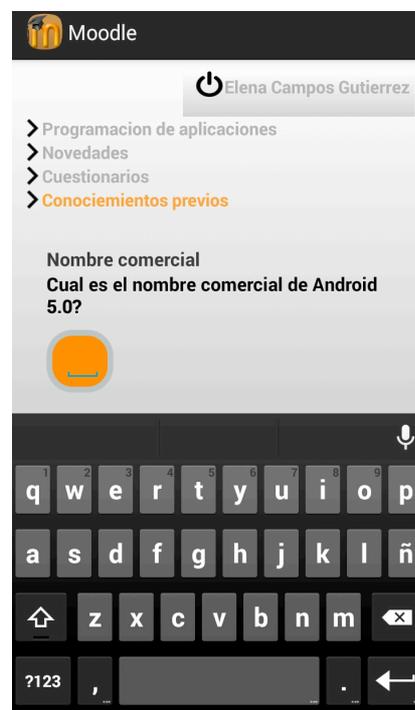


Figura 60 - Pantalla Respuesta corta



Figura 61 - Pantalla Numérica



Figura 62 - Pantalla Al Salir del Intento



Figura 63 - Pantalla Tras enviar y Terminar

## Pantalla de recursos

A esta pantalla se llega desde el menú de opciones pulsando en el botón de "cuestionarios".

Esta pantalla guarda gran similitud con las pantallas de foros, y cuestionarios excepto en la lista que se muestra el nombre del recurso disponible al que pulsando sobre el comienza su descarga.



Figura 64 – Pantalla Recursos



Figura 65 – Pantalla Proceso de Descarga

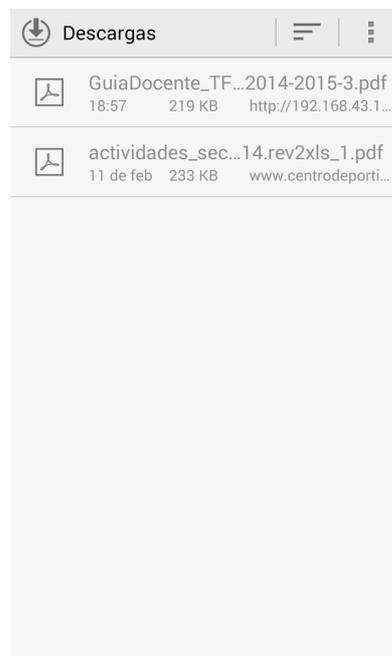


Figura 66 – Pantalla Recurso Descargado



## **CAPÍTULO VII.**

# **CONCLUSIONES Y LÍNEAS FUTURAS**



## 1. CONCLUSIONES

En este trabajo fin de grado, se ha realizado una tercera versión de una aplicación móvil para dispositivos *Android* destinada al uso académico, en la cual se incorporan algunas de las funcionalidades de *Moodle*.

La aplicación final que se consigue permite realizar las siguientes funcionalidades y actividades listadas a continuación, registro de nuevos usuarios, autenticación de usuarios registrados, mensajería instantánea, visualizar asignaturas o cursos en los que se este matriculado, secciones de los cursos nombradas con etiquetas, descarga de recursos, completa actividad en foros de discusión, evaluación de conocimientos mediante cuestionarios

El primer objetivo a realizar era el de la actividad de foros de discusión de *Moodle*, esta funcionalidad se consigue de forma completa en esta versión, ya que permite visualizar foros, discusiones, mensajes de discusión, crear nuevas discusiones, enviar nuevos mensajes y responder a otros ya existentes.

El segundo objetivo era la evaluación de conocimientos mediante la actividad que proporciona *Moodle* que son los cuestionarios. Estos pueden ser muy diversos, el tipo de cuestionario que se admite en esta aplicación es de tiempo y número de intentos ilimitado, y los tipos de pregunta que admite son preguntas de verdaderos y falso, de respuesta corta, numérica, o de respuesta múltiple de una o varias respuestas correctas. Estos tipos de preguntas han sido elegidos respecto a que son los tipos más usados y más comunes en los cuestionarios realizados por los docentes. Para la evaluación de las respuestas se ha optado por la creación de una base de datos interna en la que se almacenan las repuestas de los intentos y no se envían para calificar hasta que no sea ordenado por el usuario.

El tercer objetivo a realizar era la integración de la segunda versión que incluía las funcionalidades y actividades nombradas al comienzo de este apartado.

Para esta versión se ha seguido utilizando todas las tecnologías de las anteriores versiones: *Android* para el desarrollo de la aplicación móvil, *REST* para la implementación de los servicios Web y *JSON* para la representación de datos. Se ha optado por continuar con dichas tecnologías ya que lo que se buscaba es que el procesado de información fuese lo más ligero posible, y esa es la ventaja principal que presentan *REST* y *JSON* frente a *SOAP* y *XML* respectivamente. El utilizar de nuevo estas tecnologías también es favorable desde el punto de vista de reutilizar código.

Por último añadir que esta versión queda abierto la opción de ampliar la misma a nuevas versiones ya que *Moodle* contiene multitud de actividades y funcionalidades que pueden ser útil añadir a una aplicación móvil .

## 2. LÍNEAS FUTURAS

Teniendo en cuenta el trabajo desarrollado en este proyecto, se proponen las siguientes líneas futuras para la evolución y mejora de las prestaciones de la aplicación:

- Permitir a los distintos usuarios manejar funcionalidades características de su rol, es decir, al profesor permitirle añadir recursos, cursos, foros..., o al alumno cambiar la foto de perfil.
- Mejorar actividades ya desarrolladas. En la actividad de cuestionarios, permitir que estos tengan un límite de tiempo para desarrollarse y un límite de intentos, también visualizar las respuestas correctas tras enviar el cuestionario para calificar y en los foros visualizar desde el menú que hay mensajes nuevos.
- Desarrollar notificaciones que permitan al usuario estar informado de los recursos, mensajes en foros, cuestionarios que se añadan al instante, o de los mensajes nuevos que ha recibido sin la necesidad de entrar en la aplicación a verlos.
- Otra línea futura a considerar sería la de desarrollar esta aplicación en *iOS* para los usuarios de iPhone atendería a las necesidades de bastantes usuarios que usan estos teléfonos.
- Cambio en la interfaz de la pantalla de secciones de una lista a una lista expandible en la que se puedan visualizar los foros, cuestionarios y recursos de cada sección.

## BIBLIOGRAFÍA

- Amate, C. (2014). Conoce los Principales Sistemas Operativos Móviles. Recuperado el 24 noviembre de 2014, de <http://blogthinkbig.com/sistemas-operativos-moviles/>
- Amatellanes (2014). Integrando un RestFul Web Service en Android. Recuperado el 25 de Enero de 2015, de <https://amatellanes.wordpress.com//01/02/android-integrando-un-restful-web-service-en-android/>
- Android (2015). History. Recuperado el 10 de Marzo de 2015, de [https://www.android.com/intl/es\\_es/history/](https://www.android.com/intl/es_es/history/)
- Cosmo, J. (2014). Android 4.4 (KitKat) ya es la versión más usada con el 30,2%. Recuperado el 10 de Marzo de 2015, de <http://www.xatakandroid.com/mercado/android-4-4-kitkat-ya-es-la-version-mas-usada-con-el-30-2>
- Cosmo, J. (2015). El 1,6% de los dispositivos Android ya están actualizados a Lollipop. Recuperado 10 de Marzo de 2015, de <http://www.xatakandroid.com/mercado/el-1-6-de-los-dispositivos-android-ya-estan-actualizados-a-lollipop>
- Dept. Ciencia de la Computación. (Junio de 2014). Introducción a los Servicios Web. Invocación de servicios web SOAP. Universidad de Alicante. Recuperado el 15 de Enero de 2015, de <http://expertojava.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.pdf>
- E-abclearning (2011a). Definición E-Learning. Recuperado el 12 de Marzo de 2015, de <http://www.e-abclearning.com/definicion-e-learning>
- E-abclearning (2011b). Que es una plataforma E-Learning. Recuperado el 12 de Marzo de 2015, de <http://www.e-abclearning.com/queesunaplataformadeelearning>
- Esquiva Rodríguez, A. (2013). JSON qué es y para qué sirve. Recuperado el 20 de Marzo de 2015, de <https://geekytheory.com/json-i-que-es-y-para-que-sirve-json/>
- Fontela Sánchez, A. (2013). 6 alternativas a Moodle para elearning. Recuperado el 18 de diciembre de 2014, de <http://openwebcms.es/2013/6-alternativas-moodle-para-elearning/>
- Google Play (2015). Moodle Mobile. Opiniones. Recuperado el 24 de Marzo de 2015, de <https://play.google.com/store/apps/details?id=com.moodle.moodlemobile&hl=es>
- Hernández, D. (2010). Donde se Guardan los Contenidos Académicos que Suben a la Plataforma en un Curso Especifico. Recuperado el 12 de Octubre de 2014, de <https://moodle.org/mod/forum/discuss.php?d=164482&parent=721142>
- Herrero Herrero, F. (2013). Desarrollo de una Aplicación m-Learning Android con Acceso a Moodle. Proyecto Fin de Carrera. Valladolid: Universidad de Valladolid.

Antolin Blanco, J. (2015). Aplicación Android de acceso a los contenidos de Moodle. Proyecto Fin de Carrera. Valladolid: Universidad de Valladolid.

JSON (s.f.). Introducing JSON. Recuperado el 15 de Octubre de 2014, de <http://json.org/>

Marín, M. (2014). Android domina el mercado mundial de 'smartphones' con un 85% de cuota. Recuperado el 24 de noviembre de 2014, de <http://www.ticbeat.com/tecnologias/android-domina-mercado-mundial-smartphones/>

Moodle (2015a). Antecedentes. Recuperado el 10 de Marzo de 2015 de <https://docs.moodle.org/all/es/Antecedentes>

Moodle (2015b). Arquitectura. Recuperado el 10 de Marzo de 2015 de <https://docs.moodle.org/all/es/Arquitectura>

Moodle (2015c). Características. Recuperado el 10 de Marzo de 2015 de <https://docs.moodle.org/all/es/Características>

Moodle (2015d). Filosofía. Recuperado el 10 de Marzo de 2015 de <https://docs.moodle.org/all/es/Filosofía>

Moodle (2015e). Manual de Estilo de Código. Recuperado el 12 de Marzo de 2015, de [https://docs.moodle.org/all/es/Manual\\_de\\_Estilo\\_de\\_Codigo](https://docs.moodle.org/all/es/Manual_de_Estilo_de_Codigo)

Moodle (2015f). Paquetes de Idioma. Recuperado el 10 de Marzo de 2015 de [https://docs.moodle.org/all/es/Paquetes\\_de\\_idioma](https://docs.moodle.org/all/es/Paquetes_de_idioma)

Paszniuk, R. (2014). Acceso a Web Service SOAP en Android. Recuperado el 20 de Enero de 2015, de <http://www.programacion.com.py/moviles/android/acceso-a-web-service-soap-en-android>

Pes, C. (2014). Tutorial de XML. Recuperado el 20 de diciembre de 2014, de <http://www.abrirllave.com/xml/>

Php (2013). Funciones de hashing de contraseñas. Recuperado el 20 de Noviembre de 2014, de <http://php.net/manual/es/function.password-hash.php>

Picurelli, L. (2014). Cómo darse de alta como desarrollador iOS. Recuperado el 15 de diciembre de 2014, de <http://www.yeeply.com/blog/como-darse-de-alta-como-desarrollador-ios/>

Pueyo, J. (2009). Ejemplo con roles, usuarios y contextos. Recuperado el 12 Octubre de 2014, de <http://moodle-chile.cl/blog/noticias/moodle-foro/103-para-administradores/3050-ejemplo-con-usuarios-roles-y-contextos.html>

World Wide Web Consortium(2013). Guía Breve de Servicios Web. Recuperado el 20 de Noviembre de 2014, de <http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>

Zamora, J.A. (2014), Aprende Android en 20 conceptos. Recuperado el 20 de Octubre de 2014, de <http://www.elandroidelibre.com/2014/02/aprende-android-en-20-conceptos-empezando-a-programar-para-android.html>



# **ANEXO I.**

## **VERSIONES DE ANDROID**



Android es el sistema operativo móvil que utilizan más de mil millones de smartphones y tablets. Como se puede observar en la página oficial la compañía hace referencia a mejorar o endulzar la vida de sus usuarios a través de su sistema operativo, por lo que las diferentes versiones del mismo llevan nombre de un dulce en inglés, ordenados alfabéticamente, ordenación que coincide con su fecha de aparición: *Apple Pie, Banana Bread, Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, KitKat y Lollipop*(Android, 2015). En la imagen siguiente se puede ver la distribución de las diferentes versiones en el panorama actual. En la imagen no se recogen los dispositivos actualizados a la versión Lollipop, los cuáles únicamente conforman en la actualidad el 1,6% (Cosmo, 2015).

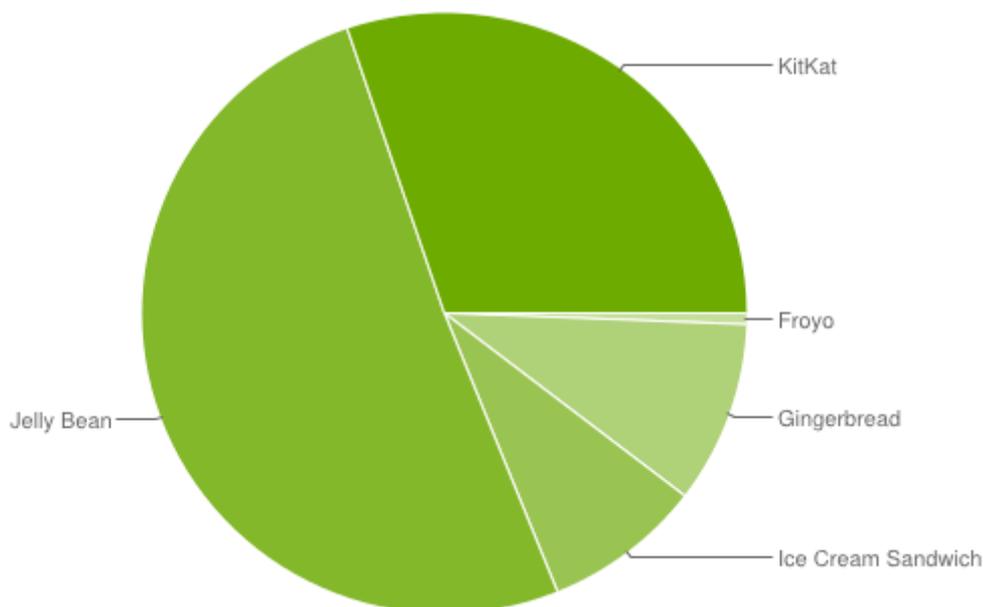


Figura 67- Cuota de Mercado Versiones Android

Version	Codename	API	Distribution
2.2	Froyo	8	0.6%
2.3.3 - 2.3.7	Gingerbread	10	9.8%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	8.5%
4.1.x	Jelly Bean	16	22.8%
4.2.x		17	20.8%
4.3		18	7.3%
4.4	KitKat	19	30.2%

Figura 68- Cuota de Mercado Versiones Android Tabla

### Android 0.X, primeros pasos

Hace ya más de seis años que apareció la primera forma de Android. No estaba pensada para pantallas táctiles y su aspecto difiere mucho del Android que la mayoría de nosotros pudo apreciar por primera vez. De hecho estas versiones únicamente estaban disponibles en formato beta y no aptas para su venta comercial.

Lo que tenemos ante nosotros es un clon de una Blackberry con una versión de Android primigenia, ni estaba Google ni el iPhone había sido presentado todavía. Tenemos una resolución de 320×240, sin widgets, sin el botón de atrás, sin notificaciones y con una configuración mínima.

A medida que pasó el tiempo con las sucesivas betas se fueron añadiendo y mejorando todos los menús, contactos, mensajes, llamadas, navegador, maps... se añadieron las notificaciones, algunos widgets. De las primeras betas a la versión 0.9 pasó mucho tiempo. Los cambios fueron sucesivos y muy diversos.



Figura 69- Interfaz Android Básica

### Android 1.0/1.1 – APPLE PIE – Nivel de api 1

Android 1.0, la versión que introdujo las Google Apps. Además de ellas, también representó la llegada de la app más importante de nuestro días; el Android Market, que luego se convertiría en lo que hoy día es Google Play.

Esta nueva versión añadía nuevas opciones, como el patrón de desbloqueo, las alarmas, las notificaciones de batería y algunas otras aplicaciones como Gmail, Google Calendar, el reproductor de videos YouTube...



**Figura 70- Interfaz Android ApplePie**

#### ANDROID 1.1 BANANA BREAD, NIVEL DE API 2

Aparece en febrero de 2009. Es una versión en la que apenas se añadieron funcionalidades. Básicamente, se trató de corregir algunos errores de la versión anterior.

#### Android 1.5 Cupcake – NIVEL DE API 3

Fue la primera versión Android con apodo propio, al menos que se hiciera público. Fue lanzada en abril de 2009. Se basó en el núcleo Linux 2.6.27.

La mayor novedad fue el teclado táctil con las opciones de mayúsculas, minúsculas y adaptado a los distintos fabricantes. El panel de notificaciones fue rediseñado y se permitió por primera vez widgets de terceros.

En Cupcake también hubo grandes actualizaciones referentes a las aplicaciones móviles, entre las que se encuentra la posibilidad de subir vídeos a Youtube por primera vez.

Otras mejoras importantes fueron:

- *Widgets* de escritorio y *live folders*.
- Grabación avanzada de audio y vídeo.
- Copiar y pegar.
- Subir vídeos a *YouTube* de forma directa.
- Transiciones animadas entre pantallas.
- Función de auto-rotación de la pantalla.

En la Figura 71 se puede observar una comparación gráfica de esta versión respecto a la versión 1.1

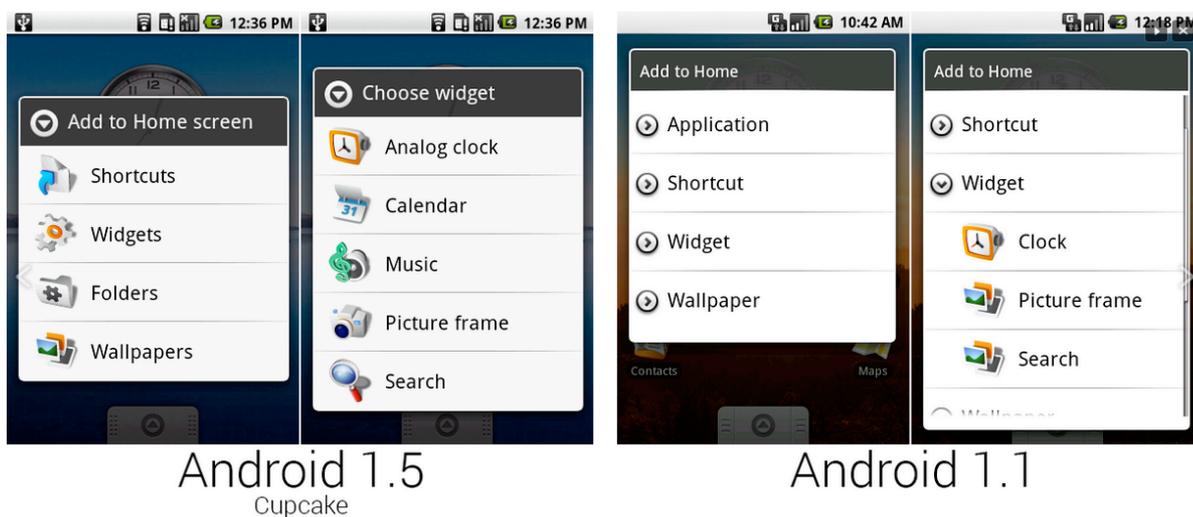


Figura 71 – Comparación Cupcake

#### Android 1.6 Donut – NIVEL DE API 4

Esta versión se lanzó en septiembre de 2009. Con Donut llegó el soporte para múltiples tamaños de pantalla distintos, soporte CDMA, y el TTS. Más allá de esto, fue una actualización menor que a pesar de corregir bugs y mejorar algún diseño no aportó grandes cambios. Los más reseñables son los siguientes:

- Capacidades de búsqueda avanzada en todo el dispositivo, incluso por voz.
- Puede trabajar con diferentes densidades de pantalla.

- Soporte para pantallas WVGA.
- Soporte para CDMA/EVDO, 802.1x y VPN.
- Mejoras en la aplicación de la cámara.
- Mejoras en el *Android Market* permitiendo realizar al usuario búsquedas de aplicaciones de forma más sencilla.
- Aparece un nuevo atributo, *onClick*, que puede especificarse en una vista.

En la imagen, se puede observar los cambios producidos a nivel de interfaz en la tienda de aplicaciones Google Market respecto a la versión 1.5. Podemos observar como el entorno está más trabajado y la carga gráfica es mayor.

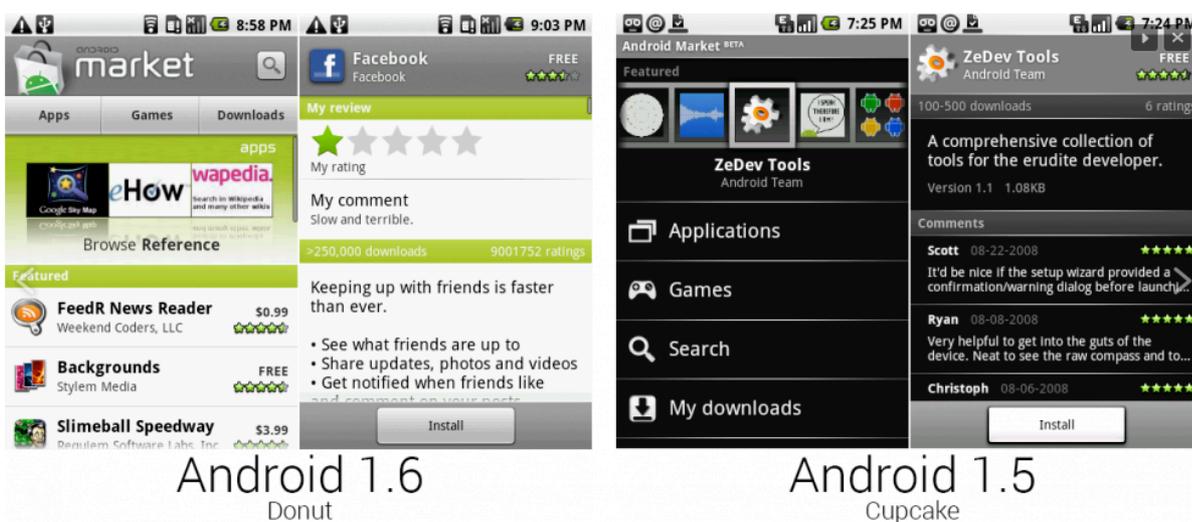


Figura 72 – Comparación Donut

### Android 2.0/2.1 Eclair – Nivel de api 5 y 7

Android 2.0 Eclair fue lanzada en diciembre de 2009, vino acompañada en un principio del smartphone Nexus One aunque en realidad este vino ya con la 2.1.

Esta versión apenas cuenta con usuarios, debido a que la mayoría de fabricantes pasaron de la versión 1.6 a la versión 2.1 (nivel de API 7). Las novedades que introdujo la versión 2.0 son las siguientes:

- Incorpora un API para manejar el *bluetooth 2.1*.
- Permite sincronizar adaptadores para conectarlo a cualquier dispositivo.
- Ofrece un servicio centralizado de manejo de cuentas.
- Mejora la gestión de contactos y ofrece más ajustes en la cámara.
- Incrementa el número de tamaños de ventana y resoluciones soportadas.
- Incorpora una nueva interfaz del navegador y soporte para HTML5.

La versión 2.1 se considera una actualización menor, por lo que conservó el nombre *Eclair*. Su lanzamiento se produjo en enero de 2010. En la Figura 73 podemos ver una comparativa del Nexus One con Eclair con el Iphone 3, el gran competidor del mismo en ese momento.



Figura 73 – Comparación Nexus Iphone

Como se ha explicado, la versión 2.1 fue más bien una actualización de la versión 2.0, más que una nueva versión, en la imagen X se pueden observar unos pequeños cambios de interfaz entre una y otra.

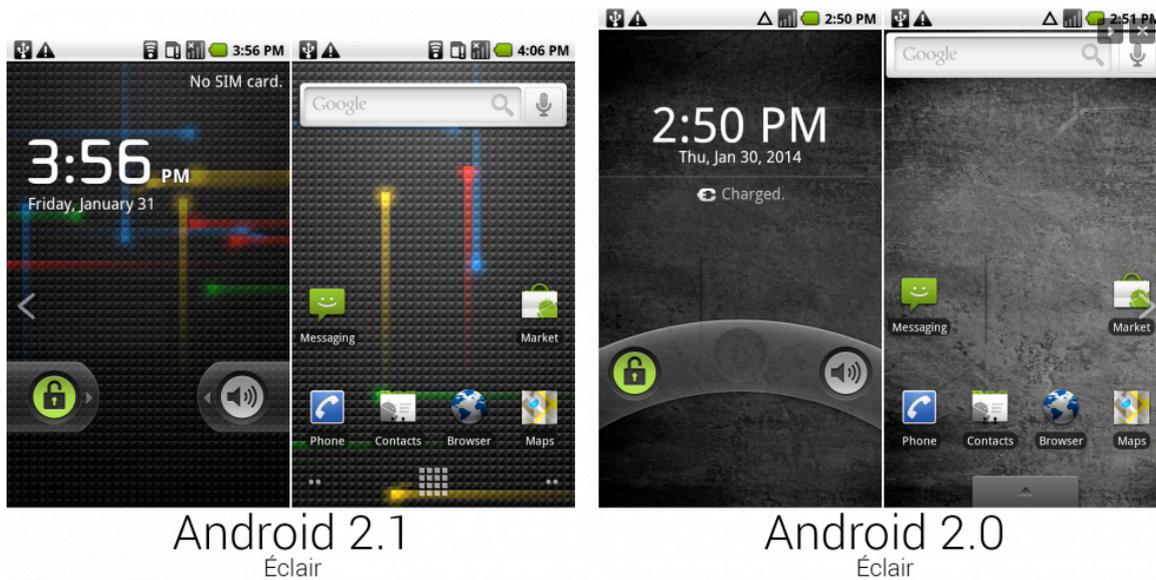


Figura 74- Comparación Eclair

#### Android 2.2 Froyo – NIVEL DE API 8

Esta versión tuvo su lanzamiento en mayo de 2010. Se mejoró muy significativamente el rendimiento. Fue gracias sobre todo a la inclusión del motor V8 javascript en Chrome y la inclusión de JIT. También en Android 2.2 se insertó la barra de búsqueda en el escritorio como algo permanente y característico. Mientras, en el Android Market se añadía la posibilidad de “actualizar todo”.

Android 2.2 Froyo se hizo también famosa por soportar Flash, a diferencia de su principal rival.

La característica más destacable son las optimizaciones en velocidad, memoria y rendimiento. También cabe destacar las siguientes:

- Nuevas mejoras en el navegador web.
- Permite instalar aplicaciones en medios de almacenamiento externo.
- Ofrece actualizaciones automáticas de las aplicaciones cuando aparecen nuevas versiones.
- Permite interacciones con el reconocimiento de voz.
- Mejoras en la conectividad.

En la siguiente imagen se pueden observar algunos cambios de interfaz significativos entre la versión Froyo y la versión anterior.



Figura 75 – Comparación Froyo

#### ANDROID 2.3 GINGERBREAD, NIVEL DE API 9

Esta versión sale a la luz en diciembre de 2010. Las principales mejoras que introdujo esta versión son las siguientes:

- Mejoras en la funcionalidad de “cortar, copiar y pegar”.
- Mejoras en el rendimiento, optimizando el cierre de aplicaciones en desuso y permitiendo así reducir el consumo de batería.
- Soporte nativo para telefonía sobre Internet VoIP/SIP.

En la siguiente imagen se puede ver la interfaz de bienvenida que presentaba esta versión de fábrica



Figura 76 - Diseño Nexus

Android 2.3 Gingerbread fue la primera versión de Android donde Google prestó especial importancia del diseño. El Nexus S fue el primer móvil fruto de la unión entre el buscador y Samsung y el encargado de presentar Gingerbread.

Los menús grisáceos de las versiones anteriores se aligeraron, el panel de notificaciones se modificó y el “copiar y pegar” se resaltó para facilitar su uso. Esto se puede observar en la imagen X. En la cual se puede ver una mayor estilización de las formas respecto a la versión 2.2.



Figura 77 - Comparación Gingerbread

### ANDROID 3.0 HONEYCOMB, NIVEL DE API 11

Android 3.0 Honeycomb sorprendió con su propuesta radical para adaptarse a los tablets, un dispositivo de moda cuando fue lanzada esta versión (febrero de 2011). Los cambios sentaron el precedente de lo que luego tendríamos con ICS. El diseño adquirió los tonos azulados que fueron mantenidos hasta las versiones más actuales.

También fue a partir de esta versión cuando el botón de multitarea empezó a cobrar importancia respecto al de menú. La pantalla de desbloqueo cambió se empezaron a ver las primeras apps Android adaptadas a pantallas grandes.

Esta versión, lanzada en febrero de 2011, se desarrolló para mejorar la experiencia de *Android* en las *tablets*, ya que estaba optimizada para dispositivos con pantallas grandes. Las principales novedades son las siguientes:

- Resolución por defecto WXGA, escritorio 3D con *widjets* rediseñados, nuevos componentes y vistas, mejoras en las notificaciones y otras características para optimizar el uso en las pantallas más grandes.

- Mejoras en la reproducción de animaciones 2D/3D gracias al *renderizador OpenGL* acelerado por hardware.
- Ejecución más rápida de las aplicaciones.
- Compatibilidad con las aplicaciones creadas para versiones anteriores, a pesar de que la interfaz gráfica está optimizada para *tablets*.

En mayo de 2011 se lanzó la versión 3.1 (nivel de API 12), cuya principal mejora es que permite manejar dispositivos conectados por USB. En julio de ese mismo año apareció la versión 3.2 (nivel de API 13), que ofrecía optimizaciones para distintos tipos de *tablets*. En la imagen contigua se puede observar la pantalla inicial de una Tablet con esta versión.



Figura 78 – Interfaz Tablet

#### ANDROID 4.0 ICE CREAM SANDWICH, NIVEL DE API 14

Con la cuarta versión se alcanzó el status de sistema operativo moderno, capaz de empezar a dominar el mercado móvil. El Samsung Galaxy Nexus fue el elegido para representarlo.

Los cambios de diseño probados en Honeycomb llegaron también a las pantallas pequeñas, la pantalla de notificaciones volvió a ocupar todo el espacio, se insertó la pestaña de widgets,

NFC recibió soporte total y también el dialer sufrió cambios. Youtube se actualizó a una versión menos horripilante y vimos en el menú de ajustes la posibilidad de controlar el uso de datos.

Ice Cream Sandwich fue la versión en la que las pautas de diseño de Android se crearon y aún hoy siguen más o menos vigentes. También fue en ICS cuando vimos el cambio del Android Market a Google Play, una evolución que sería el inicio de la estrategia comercial de Google en el sector móvil.

La versión 4.0 de Android, lanzada en octubre de 2011, tuvo como principal objetivo unificar las dos versiones anteriores (2.x para teléfonos y 3.x para tablets), con el fin de buscar la compatibilidad con cualquier tipo de dispositivo. Las principales características son las siguientes:

- La interfaz de usuario aparece totalmente renovada, reemplazando los botones físicos por botones en pantalla como ocurría en las versiones 3.x.
- Mejoras en las aplicaciones de reconocimiento, tanto facial como de voz.
- Acceso a notificaciones y navegación entre aplicaciones abiertas mucho más cómodo, gracias a mejoras en las barras de sistema y de acción.

En diciembre de 2011 se lanzó la versión 4.0.3 (nivel de API 15), en la que aparecen ligeras mejoras en algunas APIs, como el calendario, el revisor ortográfico y las bases de datos, entre otros.

En la Figura 79 podemos ver la pantalla de inicio predeterminada de esta versión:



Figura 79- Interfaz Nexus IV

Por otra parte en la figura 80 se pueden observar cambios significativos en la interfaz respecto a la versión inmediatamente inferior.



Android 4.0  
Ice Cream Sandwich

Android 3.2  
Honeycomb

Figura 80 - Comparación Ice Cream Sandwich

#### Android 4.1 – 4.3 Jelly Bean – NIVEL DE API 16-17

Jelly Bean sigue siendo a Junio de 2014 la versión más extendida y utilizada de Android. Uno de los mayores cambios que se introdujo fue Project Butter con el que se mejoró nuevamente el rendimiento, el sistema de notificaciones se expandió para permitir respuestas rápidas, se mejoró la cámara, se introdujo Hangouts y se añadió Google Now, el sistema inteligente que se anticipa a nuestras búsquedas.

Paralelamente los Google Play Services empezaron a integrarse de manera significativa en el sistema, formando parte de la columna vertebral de Android. Para relanzar JB se lanzaron diversos Nexus, desde la primera tablet de siete pulgadas hasta el Nexus 4, todos ellos con una calidad/precio muy ajustada.

#### Android 4.4 KitKat

Vino acompañada con la presentación del Nexus 5, el cual se puede observar en la figuraX mostrando a su vez la pantalla de inicio predeterminada de esta versión. Se permitió que dispositivos con 512MB de RAM pudieran ejecutar Android sin problemas lo que ha permitido una mejora muy significativa de la gama baja. Actualmente es la versión más utilizada (Cosmo, 2014).



**Figura 81 – Interfaz Nexus V**

La interfaz pasó a tener un tono más grisáceo, las carpetas se modificaron y se lanzó el Google Experience Launcher para los Nexus. También ha sido desde KitKat cuando el comando por voz de “OK, Google” se empieza a utilizar. Otro cambio es la manera de añadir widgets y wallpapers o el dialer con interfaces distintas a las anteriores. Aquí van las imágenes de nuestra versión más actual.

#### Android 5.0 – Lollipop

Fue presentada el 3 de noviembre del 2014, es la última versión conocida. Las características principales y novedosas de la siguiente versión son las siguientes:

- **Material Design:** Un diseño intrépido, colorido, y sensible interfaz de usuario para las experiencias coherentes e intuitivos en todos los dispositivos. Movimiento de respuesta natural, iluminación y sombras realistas y familiares elementos visuales hacen que sea más fácil de navegar su dispositivo. Nuevos colores vivos, tipografía e imágenes de ayuda de borde a borde de enfocar su atención.

- **Notificaciones:** Nuevas formas de controlar cuándo y cómo se reciben mensajes - sólo ser interrumpido cuando se quiere ser. Ver y responder a mensajes directamente desde la pantalla de bloqueo. Incluye la capacidad de ocultar contenido sensible para estas notificaciones. Se puede programar el tiempo durante el cual sólo las notificaciones de prioridad aparecen. También, las llamadas entrantes no interrumpen lo que estés haciendo. Se puede optar por responder a la llamada o simplemente seguir haciendo lo que se esté haciendo. Clasificación más inteligente de notificaciones. Ver todas las notificaciones en un solo lugar tocando la parte superior de la pantalla.
- **Batería:** Una característica de ahorro de batería que se extiende el uso de dispositivos de hasta 90 minutos. El tiempo estimado de batería restante aparece cuando el dispositivo está enchufado. El tiempo restante de batería antes de tener que cargar el dispositivo de nuevo ahora se puede encontrar en la configuración de la batería.





## **ANEXO II.**

# **ESTILO DE CÓDIGO Y NORMAS MOODLE**



## ESTILO DEL CÓDIGO

Cualquier proyecto colaborativo necesita que la consistencia y la estabilidad sean fuertes. Siguiendo el manual de estilo de código desarrollado en la documentación de *Moodle* se ha tratado de cumplir todas las reglas explicadas en él. Todo el código nuevo definitivamente deberá adherirse a estos estándares de la forma más exacta posible (Moodle, 2015e).

### Reglas generales

Las reglas generales que en este manual se especifican son las siguientes:

- Todos los archivos de código deberían utilizar la extensión `.php`.
- Todas las plantillas deberían utilizar la extensión `.html`.
- Todos los archivos de texto deberían utilizar el formato de texto Unix (la mayoría de los editores de texto tienen esto como una opción).
- Todas las etiquetas PHP deben ser 'completas' como `<?php ?>` ... no 'reducidas' como `<? ?>`.
- Todos los avisos de copyright deben ser mantenidos. Puede incluir los suyos propios si resulta necesario.
- Todos los archivos deben incluir el archivo principal `config.php`.
- Cualquier otro `include/require` debería utilizar una ruta absoluta que comience por `$CFG->dirroot` o `$CFG->libdir`, nunca relativos, ya que estos en algunas ocasiones funcionan de forma extraña en PHP.
- Cada archivo debería comprobar que el usuario está autenticado correctamente, utilizando las funciones `require_login()` y `isadmin()`, `isteacher()`, `iscreator()` o `isstudent()`.
- Todos los accesos a la base de datos deberían utilizar las funciones definidas en `lib/datalib.php` cuando sea posible, esto permite la compatibilidad con un gran número de bases de datos. Si se quiere escribir código SQL entonces se deberá comprobar que: funciona en cualquier plataforma; restringido a funciones específicas de su código (normalmente un archivo `lib.php`); y claramente comentado.
- No se deben crear o utilizar variables globales distintas de las estándar `$CFG`, `$SESSION`, `$THEME`, `$SITE`, `$COURSE` y `$USER`.
- Todas las variables deberían ser inicializadas o, al menos, comprobada su existencia utilizando `isset()` o `empty()` antes de ser utilizadas.

- Todas las cadenas deberían ser traducibles. Para ello, se deben crear nuevos textos en los archivos lang/es\_utf8 con palabras reducidas en inglés y su traducción completa al español. Para recuperarlas en el código se utilizan las funciones `get_string()` o `print_string()`.
- Todos los errores deberían ser visualizados utilizando la función `print_error()` para maximizar la traducción y ayudar a los usuarios (automáticamente se enlaza con Moodle Docs).
- Todos los ficheros de ayuda deben ser traducibles. Para ello, se deben crear nuevos textos en el directorio lang/es\_utf8/help y llamarlos utilizando la función `helpbutton()`. Si se necesita actualizar un fichero de ayuda:
- Para un pequeño cambio, donde la traducción antigua del fichero podría tener todavía sentido, está permitido que se haga el cambio, pero se debería notificarlo a [translation@moodle.org](mailto:translation@moodle.org).
- Para un cambio importante se tendría que crear un nuevo fichero añadiéndole en el nombre un número incrementado (p.ej. filename2.html) para que los traductores puedan ver fácilmente que se trata de una nueva versión del archivo. Obviamente el nuevo código y los índices de las páginas de ayuda deben ser modificados para apuntar a las versiones más recientes.
- La información que llega desde el navegador (enviada con los métodos GET o POST) automáticamente tiene las `magic_quotes` aplicadas (sin importar la configuración de PHP) por lo que se pueden insertar con total seguridad en la base de datos. El resto de la información (obtenida desde los archivos, o desde la base de datos) debe ser escapada con la función `addslashes()` antes de insertarla en la base de datos.
- Muy importante: Todos los textos dentro de Moodle, especialmente aquellos que han sido introducidos por los usuarios, deben ser mostrados utilizando la función `format_text()`. Esto asegura que el texto es filtrado y limpiado correctamente.
- Al generar enlaces HTML, se deben hacer siempre relativos a la raíz del sitio Moodle. Esto permite que su código funcione aunque sea llamado por un script que se encuentre en otra carpeta diferente.

## Estilo del código

Aunque pueda resultar un poco complicado modificar el estilo de programación personal de cada uno, es comprensible que, debido a que Moodle evoluciona gracias a una comunidad de desarrolladores, se trate de llegar a un estilo común puesto que puede resultar ciertamente complicado encontrarle sentido al código de Moodle si está compuesto por una mezcla de estilos.

Hay muchos puntos a favor y en contra de cada estilo que la gente utiliza, pero el que se detalla a continuación, es el que se debe emplear.

- El sangrado del texto debe ser siempre de 4 espacios. No se deben utilizar los tabuladores nunca.
- Los nombres de las variables tienen que ser siempre fáciles de leer, procurando que sean palabras en minúsculas con significado en inglés. Si realmente se necesita más de una palabra, se deben poner juntas, pero procurando que sean tan breves como sea posible. Se deben utilizar nombres en plural para las matrices de objetos.
- Las constantes tienen que definirse siempre en mayúsculas, y empezar siempre por el nombre del módulo al que pertenecen. Deberían tener las palabras separadas por guiones bajos.
- Los nombres de las funciones tienen que ser palabras sencillas en minúsculas y en inglés, y empezar con el nombre del módulo al que pertenecen para evitar conflictos entre módulos. Las palabras deberían separarse por guiones bajos. Los parámetros, si es posible, tendrán valores por defecto. Se debe comprobar que no haya espacio entre el nombre de la función y lo siguiente (paréntesis).
- Los bloques de código siempre deben estar encerrados por llaves (incluso si solo constan de una línea).
- Las cadenas tienen que ser definidas utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.
- Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones y variables. Los comentarios en línea deberían utilizar los caracteres `//`, alineados con cuidado por encima de las líneas de código que comenta.
- El espacio en blanco se puede utilizar con bastante libertad para ganar claridad. Generalmente, debería haber un espacio entre llaves y líneas normales y ninguno entre llaves y variables o funciones.
- Cuando se esté realizando una copia de un objeto, se debe utilizar siempre la función `clone()` originalmente sólo disponible en php5 (en caso contrario simplemente tendrá una referencia al primer objeto). Moodle garantiza que este método funcionará también bajo php4. Si lo que se quiere copiar no es un objeto, pero puede contener objetos (p.ej. un array de objetos) utilice la función `fullclone()` en su lugar.

### **Reglas en la gestión de la base de datos**

A continuación se repasan las reglas básicas para la gestión de la base de datos.

- Cada tabla debe tener un campo autonumérico `id` (`INT10`) como clave primaria.

- La tabla principal que contiene instancias de cada módulo debe tener el mismo nombre que el módulo y contener, por lo menos, los siguientes campos:
  - o id: descrito arriba.
  - o course: el identificador del curso al que la instancia pertenece.
  - o name: el nombre completo de la instancia.
- El resto de las tablas asociadas con un módulo que contiene información sobre 'cosas', deberían ser llamadas módulo\_cosas.
- Los nombres de las tablas y de los campos tienen que evitar el uso de palabras reservadas por las bases de datos.
- Los nombres de los campos (columnas) deberían ser sencillos y cortos, siguiendo las mismas reglas que los nombres de las variables.
- Cuando sea posible, las columnas que contengan una referencia al campo id de otra tabla (por ejemplo, módulo) debería ser llamado moduloid.
- Los campos booleanos serán implementados como enteros cortos (por ejemplo, INT4) con los valores 0 o 1, para permitir la futura expansión de los valores si fuera necesario.
- La mayoría de las tablas tienen que tener un campo timemodified (INT10) que será actualizado con la fecha actual (timestamp de UNIX) obtenida con la función time() de PHP.
- Se debe definir siempre un valor por defecto con sentido para cada campo.
- Cada tabla debe comenzar con el prefijo de la base de datos (\$CFG->prefix). En muchos casos esto es gestionado automáticamente. Además, bajo PostgreSQL, el nombre de cada índice debe empezar también con el prefijo.
- Para garantizar la compatibilidad entre bases de datos, se deben seguir las reglas siguientes sobre el uso del comando AS (sólo si se necesita alias en tablas y campos):
  - No utilizar el comando AS para alias de tablas.
  - Utilizar el comando AS para alias de campos (columnas).
- Nunca se deben emplear UNIQUE KEYs (restricciones) para nada. En su lugar se deben utilizar UNIQUE INDEXes. En el futuro, si se decide añadir integridad referencial a Moodle y si se necesitan UNIQUE KEYs, serán utilizadas, pero no por ahora.

- Esas UNIQUE KEYS creadas en el Editor XMLDB sólo deben ser definidas si el campo/campos van a ser el objetivo para alguna FOREIGN KEY (a nivel de Editor). En caso contrario, se deben crear como UNIQUE INDEXes.
- Nunca se deben realizar cambios a la base de datos en ramas estables. Si se hace eso, entonces los sitios actualizando de una versión estable a la siguiente pueden encontrarse con cambios por duplicado, lo cual puede producir errores serios.
- Cuando se haga referencia a una variable entera en consultas SQL, no se debe entrecomillar el valor. Por ejemplo, `get_records_select('question' "category=$catid")` es correcto, mientras que `get_records_select ('question', "category='$catid'")` es incorrecto. Ese uso oculta posibles errores cuando \$catid está sin definir.

### Normas de seguridad

- No basarse en `register_globals`. Cada variable debe ser correctamente inicializada.
- Inicialice todos los arrays y objetos aunque estén vacíos.
- No utilizar la función `optional_variable()`. En su lugar, utilizar la función `optional_param()`.
- No utilizar la función `require_variable()`. En su lugar, utilizar la función `required_param()`.
- Utilizar `data_submitted()` con cuidado. La información debe ser limpiada antes de utilizarla.
- No utilizar `$_GET`, `$_POST` o `$_REQUEST`. En su lugar, utilizar las funciones `required_param()` u `optional_param()`.
- No comprobar las acciones con código como: `if (isset($_GET['algo']))`.
- Cuando sea posible agrupar todas las llamadas a `required_param()`, `optional_param()` y el resto de inicialización de variables en el principio de cada fichero.
- Utilizar el mecanismo `sesskey` para proteger el envío de formularios de ataques.
- Todos los nombres de ficheros deben ser 'limpiados' utilizando la función `clean_filename()`.
- Cualquier información leída desde la base de datos debe tener la función `addslashes()` aplicada antes de volver a enviar la información a la base de datos.
- La información que se almacenará en la base de datos debe venir de peticiones POST.
- No utilizar información obtenida de `$_SERVER`.

- La información enviada a la base de datos debe ser filtrada mediante la función `clean_param()`.
- Asegurarse de que el código SQL es correcto.
- Comprobar toda la información (especialmente la que es enviada a la base de datos) en cada archivo que es utilizada.
- Los bloques de código que se incluyan deben presentar una estructura PHP correcta.
- Para utilizar funciones que invoquen un Shell hay que asegurarse de que se han limpiado los parámetros anteriormente con `escapeshellcmd()` o `escapeshellarg()`.

