Universitat Oberta de Catalunya
www.uoc.edu

Estudis d'Informàtica i Multimèdia
Enginyeria Informàtica

Projecte de final de carrera
Xarxes de computadors

Desenvolupament d'una aplicació Android per a Moodle

Memòria final

Autor: Manel Herrera Borau
Consultor: Jordi Ceballos Villach
Data: Gener 2012

Resum

L'evolució que la tecnologia de les comunicacions sense fils ha experimentat en els darrers anys permet que avui en dia els dispositius mòbils proporcionin una resposta més que acceptable. Ja s'han superat molts dels problemes de mobilitat, d'infraestructura i d'ample de banda que fins ara dificultaven l'ús d'aquests dispositius.

Aquest fet ha fomentat l'expansió de diferents sistemes dels que es pot destacar Android, iPhone OS i Windows Mobile/Windows Phone 7. Els dos primers són els que actualment es disputen el lideratge del mercat dels dispositius i aplicacions mòbils i és Android el que ha experimentat el major creixement els darrers anys.

Paral·lelament, la constant introducció de programari d'*e-learning* destinat a activitats de formació no presencial basades en Internet ha permès arribar a milions d'usuaris en diferents àmbits, com ara universitats, empreses i institucions d'arreu del món. D'aquest conjunt de programari destaca Moodle, *suite* de codi obert que permet crear llocs web de formació en línia de forma senzilla i eficaç.

Aquest Projecte de Final de Carrera, emmarcat en l'àrea de Xarxes de Computadors, conjuga aquests tres conceptes mitjançant el desenvolupament d'una aplicació per a dispositius mòbils Android que connecta a un servidor Moodle. La comunicació entre ells utilitza intensivament les xarxes sense fils i fa crides als serveis que aquest ofereix gràcies als *Web Services*, mètode de comunicacions composat d'un conjunt de protocols i de programari que permet la comunicació entre dos dispositius a través de la xarxa.

Paraules clau: Xarxes de Computadors, xarxes sense fils, dispositius mòbils, Android, Moodle, *e-learning*, *Web Services*

Resumen

La evolución experimentada en los últimos años por la tecnología de las comunicaciones inalámbricas permite que hoy en día los dispositivos móviles proporcionen una respuesta muy aceptable. Ya se han superado muchos de los problemas de movilidad, infraestructura y ancho de banda que hasta ahora dificultaban el uso de estos dispositivos.

Este hecho ha fomentado la expansión de diferentes sistemas, entre los que cabe destacar Android, iPhone OS y Windows Mobile/Windows Phone 7. Los dos primeros son los que actualmente se disputan el liderazgo del mercado de dispositivos y aplicaciones móviles y Android el que ha experimentado el mayor crecimiento en los últimos años.

Paralelamente, la continua introducción de aplicaciones de e-learning destinadas a actividades de formación no presencial basadas en Internet ha permitido llegar a millones de usuarios de distintos ámbitos como universidades, empresas e instituciones de todo el mundo. De este conjunto de aplicaciones destaca Moodle, suite de código abierto que permite crear sitios web de formación en línea de forma sencilla y eficaz.

Este Proyecto Fin de Carrera, enmarcado en el área de Redes de Computadores, conjuga estos tres conceptos mediante el desarrollo de una aplicación para dispositivos móviles Android que conecta con un servidor Moodle. La comunicación entre ellos utiliza intensivamente las redes inalámbricas y realiza llamadas a los servicios que este ofrece gracias a los Web Services, método de comunicaciones compuesto por un conjunto de protocolos y aplicaciones que permite la comunicación entre dos dispositivos a través de la red.

Palabras clave: Redes de Computadores, redes inalámbricas, dispositivos móviles, Android, Moodle, e-learning, Web Services

Abstract

The evolution that wireless communication technology has experienced over the last few years allows that nowadays mobile devices provide a more than acceptable response. Most of the issues related to mobility, infrastructure and bandwidth that used to limit the usage of these devices have now been solved.

As a consequence, different systems such as Android, iPhone OS and Windows Mobile/Windows Phone 7 have been expanded. Nowadays, Android and iPhone are leading the mobile devices and applications market, Android being the one that has experienced the largest growth over the last few years.

In parallel, the continuous introduction of e-learning software focused on Internet-based learning activities allows the access of millions of users from a wide range of locations such as universities, companies and institutions worldwide. Moodle, the open-source suite for building on-line learning websites in a simple and effective manner should be highlighted.

This Minor Thesis, focused on Computer Networks, combines these three elements by developing an application for an Android mobile device that connects to a Moodle server. The communication between both elements intensively uses the wireless networks and calls its services thanks to the Web Services, a method of communication between two devices over the network based on a set of protocols and pieces of software.

Keywords: Computer network, wireless network, mobile devices, Android, Moodle, e-learning, Web Services.

Contingut

Índex de figures i taules

1. Introducció

1.1. Context, motivació i justificació del projecte

El projecte de final de carrera està concebut per realitzar un treball de síntesi dels coneixements adquirits en altres assignatures i per posar-los en pràctica de forma conjunta en un projecte concret, pràctic i vinculat a l'exercici professional de la informàtica.

L'àrea específica d'aquest projecte –Xarxes de Computadors– pretén dur a terme un projecte de desenvolupament d'una aplicació per a dispositius mòbils atès que per l'evolució de la tecnologia de les comunicacions sense fils, actualment aquests dispositius ja proporcionen una resposta acceptable als problemes de mobilitat i d'ample de banda. De les diferents temàtiques proposades en aquesta àrea, s'ha escollit l'enfocada en els dispositius Android. La decisió per triar Android i no un altre ha estat motivada per ser el sistema de dispositius mòbils que ha experimentat un major creixement en els darrers anys i per ser un dels més estesos en el mercat actual, la qual cosa desperta un especial interès per l'aprenentatge d'aquesta tecnologia en expansió.

Pel que respecta al programari concret a desenvolupar, s'ha decidit construir una aplicació per Moodle, *suite* de codi obert que permet crear llocs web de formació en línia de forma senzilla i eficaç. Les raons per desenvolupar una aplicació per Moodle han estat quatre: que els programaris d'*e-learning* destinats a formació no presencial basats en Internet són essencials en les activitats formatives, que Moodle és probablement el líder entre tots els programaris d'*e-learning*, que en els darrers temps han aparegut solucions per a dispositius mòbils que possibiliten l'accés als recursos de formació en mobilitat i, finalment, que Moodle disposa d'una nova API que és ideal per desenvolupar una aplicació que *consumeixi* els serveis que ofereix a través de la xarxa fent ús de *Web Services*.

Aquest darrer punt, l'ús de *Web Services*, és la part més interessant d'aquest projecte atès que la API que Moodle ofereix en la seva versió 2 és recent i en conseqüència no hi ha gaire documentació ni aplicacions que la facin servir. Aquest fet fa que el projecte sigui especialment atractiu en haver de destinar una part important dels recursos a la investigació i l'estudi de l'API.

1.2. Descripció del projecte

Com a resum del que s'ha exposat en el punt anterior, l'objecte d'aquest projecte és l'anàlisi, el disseny i la implementació d'una aplicació Android per a Moodle que utilitzi les xarxes sense fils per establir la connexió entre el dispositiu mòbil Android i Moodle i, més concretament, faci ús de *Web Services* per *consumir* els serveis que ofereix el servidor.

El projecte es sosté sobre quatre pilars:

- El primer és el llenguatge i la tècnica de desenvolupament. L'aplicació es desenvoluparà íntegrament amb el llenguatge de programació Java, que és amb el que es desenvolupen les aplicacions Android. La programació es farà sota el paradigma de l'orientació a objectes i aplicant una arquitectura de tres capes. D'aquesta forma es garanteix que l'aplicació resultant disposarà de nivells òptims d'abstracció, facilitarà la

reutilització, hi haurà baix acoblament entre els seus elements i aquests estaran fortament cohesionats. Addicionalment al llenguatge Java estàndard, es faran servir els complements que permeten la construcció d'aplicacions Android.

- El segon pilar és la tecnologia de xarxa. Un dels objectius del projecte és l'ús de les xarxes sense fils amb la qual cosa l'entorn haurà de ser capaç de funcionar amb tecnologies com per exemple WiFi, 3G... o una combinació d'aquestes. Cal remarcar que l'ús de TCP/IP fa pràcticament transparent per l'usuari la tecnologia que es fa servir ja que els dispositius actuals permeten commutar de forma automàtica d'una xarxa a una altra.

- El tercer pilar, són les crides a *Web Services* que permetin consumir els serveis oferts per l'API de Moodle. Hi ha diferents protocols estàndard que permeten aquest tipus de comunicació orientada a missatges, essent REST el que s'ha seleccionat atenent a les recomanacions de la plataforma Moodle.

- El quart pilar és la plataforma Moodle. Aquesta plataforma és la que fem servir d'escenari per la nostra aplicació i per tant cal destinar-li una especial atenció pel que fa al seu funcionament i, sobretot, a la configuració i complements que requereix per permetre que l'aplicació Android pugui connectar-s'hi fent ús de *Web Services*.

Pel desenvolupament del projecte, i en concret dels quatre pilars descrits, es duran a terme un conjunt d'activitats que es descriuen al llarg d'aquest document, en les quals hi ha recollits tots els aspectes d'aprenentatge, quins aspectes funcionals i tecnològics s'han analitzat, quines decisions s'han pres i quin és el producte resultant.

1.3. Objectius del projecte

Els objectius que s'assoliran amb la realització d'aquest projecte són els següents:

- Posar en pràctica de forma conjunta, en un projecte concret i pràctic, tots els coneixements adquirits durant l'estudi d'Enginyeria Informàtica
- Aprendre el funcionament dels dispositius mòbils, concretament d'Android i de les seves eines i tècniques de desenvolupament
- Adquirir les bases metodològiques necessàries per realitzar l'estudi i investigació d'una API d'una plataforma –en aquest cas de Moodle– i posar-les en pràctica
- Aprofundir en l'ús de xarxes de computadors, especialment en el desenvolupament d'aplicacions que facin ús de les xarxes sense fils com a part essencial del seu funcionament
- Aprendre les tècniques i protocols necessaris per construir aplicacions que requereixin fer crides a *Web Services*
- Aprofundir en la construcció d'aplicacions Java sota el paradigma de l'orientació a objectes amb arquitectura de tres capes
- Adquirir els coneixements bàsics per la instal·lació i configuració d'un servidor Moodle

1.4. Enfocament

Malgrat que l'objectiu principal del projecte –desenvolupar una aplicació Android– apunta a que el seu enfocament és eminentment pràctic, la realitat és que hi ha una bona part d'aquest dedicada a l'estudi i investigació de la plataforma Moodle, de la seva API i dels seus *Web Services*. Addicionalment, per desenvolupar aplicacions Android no és suficient amb tenir coneixements sobre Java, si no que aquests cal ampliar-los amb les peculiaritats genèriques del desenvolupament per a dispositius mòbils i amb les específiques del sistema Android.

Consegüentment, l'enfocament es pot considerar mixt d'acord al següent raonament:

- D'una banda és teòric, atès que és necessària formació i aprenentatge en desenvolupament d'aplicacions mòbils Android
- D'altra banda és d'investigació, per les necessitats d'estudi de l'API de Moodle, sobre la qual no hi ha gaire documentació ni exemples
- Finalment és pràctic, atès que una bona part del projecte consisteix en desenvolupar, la qual cosa correspon al clàssic enfocament de projectes de anàlisi, disseny i programació

1.5. Planificació

1.5.1. Cicle de vida

El desenvolupament del projecte segueix el curs del cicle de vida clàssic el qual s'ha adaptat a la periodificació proporcionada pel consultor al començament del projecte. Aquest consta de quatre fites que corresponen a cadascun dels terminis de lliurament del projecte: primera, segona i tercera PACs més un lliurament final. Al seu torn, cadascuna d'aquestes fases es subdivideix, si s'escau, en vàries fases. La següent figura il·lustra a alt nivell el cicle de vida del projecte, on s'observa la distribució de fites i fases.

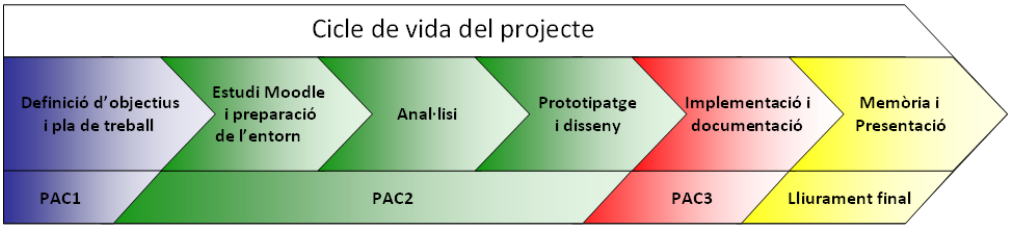


Figura 1 – Cicle de vida del projecte

1.5.2. Descomposició en tasques

Els següents apartats recullen la descomposició de totes les fites i fases en les tasques principals i específiques que s'han realitzat per assolir els objectius del projecte.

1.5.2.1. Pla de treball

En aquesta fita s'ha seleccionat la temàtica del projecte i s'ha elaborat el pla de treball amb la realització de les següents tasques.

ID	Tasca	Descripció
5	Recerca informació sobre PFC	Obtenir informació sobre com i què cal fer per realitzar el PFC, objectius, recursos, etc...
6	Decisió temàtica projecte	Discutir amb el consultor la millor opció d'entre les que es proposen
7	Recerca preliminar d'informació sobre Moodle	Una vegada decidit que el projecte serà sobre Moodle, recopilar informació sobre aquesta plataforma
8	**Pla de treball**	
9	Identificar objectius	Definir l'abast i objecte del projecte
10	Identificar fases, tasques, riscos, recursos...	Realitzar la planificació i temporització tenint en compte els recursos, les fites, els recursos i els riscos
11	Lliurar pla de treball (PAC1)	Revisar i lliurar el document "PAC1 – Pla de treball"

Taula 1 – Descomposició en tasques del Pla de Treball

1.5.2.2. Anàlisi i disseny

En aquesta fita s'ha fet un estudi profund de la plataforma Moodle i Android, s'ha realitzat l'anàlisi de requeriments identificant els requisits funcionals i no funcionals, s'ha realitzat el prototipatge de l'aplicació i finalment s'ha completat el disseny concretant els aspectes tecnològics i arquitectònics.

ID	Tasca	Descripció
15	**Estudi plataforma Moodle**	
16	Recerca en profunditat d'informació sobre Moodle	Obtenir el màxim d'informació possible sobre la plataforma Moodle consultant documentació, fòrums, exemples, programari existent, API, etc...
17	Elaboració document Síntesi Moodle (API, funcionalitats…)	Fer un resum operatiu de Moodle, indicant referències, manuals, i documents interessants
18	**Preparació entorn de desenvolupament i proves**	
19	Descàrrega programari	Descarregar tot el programari necessari pel desenvolupament, sobretot pel que respecta al IDE Eclipse i les eines de desenvolupament per Android
20	Instal·lació programari	Instal·lar i configurar el programari descarrerat. Definir un mètode de còpies de seguretat. Preparar un servidor Moodle per proves.
21	Proves programari	Realitzar les proves preliminars de funcionament de totes les eines per assegurar un entorn de treball estable
22	**Anàlisi**	
23	Requisits	Identificar les funcionalitats que ha d'implementar el programari i altres requisits de caire tècnic/ergonòmic (no funcionals)
24	Especificacions	Detallar les funcionalitats individualment i el seu comportament en conjunt
25	Diagrames necessaris (casos d'ús, col·laboració…)	Identificar els casos d'ús i crear els diagrames
26	Elaboració document Anàlisi	Crear el document parcial de la fase d'anàlisi
27	**Formació Desenvolupament Android**	
28	Formació dia 1	1er dia de formació intensiva en Android
29	Formació dia 2	2on dia de formació intensiva en Android
30	Formació dia 3	3er dia de formació intensiva en Android
31	**Prototipatge**	
32	Confecció prototips d'interfícies gràfiques	Crear uns prototips de les interfícies d'usuari principals per la seva validació amb el client/usuari final
33	Elaboració document Prototipatge	Crear el document parcial de la fase de prototipatge
34	**Disseny**	
35	Exploració	Realitzar un estudi del programari i de la plataforma per determinar la seva arquitectura
36	Diagrames necessaris (classes, seqüència, etc…)	Identificar el disseny estàtic i dinàmic, crear diagrames de classes, entitat-relació i col·laboració i seqüència.
37	Elaboració document Disseny	Crear el document parcial de la fase d'anàlisi
38	Revisió document PAC2	Preparar el document "PAC2 – Anàlisi i disseny" fent servir de base els documents parcials
39	Lliurament PAC2	Lliurar el document "PAC2 – Anàlisi i disseny"

Taula 2 – Descomposició en tasques de l'Anàlisi i Disseny

1.5.2.3. Implementació

En aquesta fita s'ha realitzat el desenvolupament de l'aplicació pròpiament dita d'acord al disseny definit a la fita anterior. Addicionalment s'han realitzat les proves necessàries per assegurar el correcte funcionament.

ID	Tasca	Descripció
41	**Desenvolupament Aplicació**	
42	Desenvolupament	Realitzar el desenvolupament (programació Java) de l'aplicació d'acord al disseny. Crear les interfícies d'usuari d'acord als prototips.
43	Preparació jocs de proves	Definir els escenaris de proves tenint en compte tots els casos d'ús establerts
44	Proves unitàries	Realitzar les proves de cada element de l'aplicació de forma individual identificant i corregint els possibles errors
45	Proves d'integració	Realitzar les proves de tots els elements de l'aplicació de forma conjunta identificant i corregint els possibles errors
46	Elaboració document Aplicació	Crear el document parcial de la fase de desenvolupament
47	**Desplegament i instal·lació**	
48	Desenvolupament	Desenvolupar el programari i/o complements necessaris per la correcta instal·lació de l'aplicació
49	Proves	Provar el correcte funcionament del programari d'instal·lació
50	Elaboració Manual Instal·lació	Crear el document d'instal·lació
51	**Manual d'usuari**	
52	Elaboració Manual d'Usuari	Crear el manual d'usuari
53	Revisió document PAC3	Preparar el document "PAC3 – Implementació" fent servir de base els documents parcials
54	Lliurament PAC3	Lliurar el document "PAC3 – Implementació"

Taula 3 – Descomposició en tasques de la Implementació

1.5.2.4. Memòria i presentació

En aquesta fita s'ha elaborat la memòria del projecte (aquest document) i la presentació virtual que l'acompanya.

ID	Tasca	Descripció
56	**Memòria i presentació**	
57	Elaboració memòria	Elaborar el document "Memòria" fent servir de base els documents de les fites anteriors
58	Elaboració presentació virtual	Preparar la presentació virtual del projecte "Presentació"
59	Revisió documents finals	Revisar el document "Memòria" i la "Presentació"
60	Lliurament final	Lliurar el document "Memòria" i la "Presentació"

Taula 4 – Descomposició en tasques de la Memòria i Presentació

1.5.3. Temporització

Per cada tasca descrita a la descomposició en fites, fases i tasques de l'apartat anterior se li assigna una data prevista d'inici i la seva durada per tal de poder seguir una temporització adequada i poder realitzar una planificació i seguiment correcte del projecte.

ID	Tasca	Durada	Inici	Final
1	**PFC xarxes - Aplicació Android per Moodle**	**128 dies**	**21/09/2011**	**26/01/2012**
2	Inici semestre	0 dies	21/09/2011	21/09/2011
3	Lectura i comprensió del pla docent	1 dia	21/09/2011	21/09/2011
4	**PAC1**	**12 dies**	**22/09/2011**	**03/10/2011**
5	Recerca d'informació sobre PFC	2 dies	22/09/2011	23/09/2011
6	Decisió temàtica del projecte	4 dies	24/09/2011	27/09/2011
7	Recerca preliminar d'informació sobre Moodle	3 dies	28/09/2011	30/09/2011
8	**Pla de treball**	**3 dies**	**01/10/2011**	**03/10/2011**
9	Trobada presencial	1 dia	01/10/2011	01/10/2011
10	Identificar objectius del projecte	1 dia	01/10/2011	01/10/2011
11	Identificar fases, tasques, riscos, recursos…	1 dia	02/10/2011	02/10/2011
12	Revisió document PAC1	1 dia	03/10/2011	03/10/2011
13	Lliurament PAC1	0 dies	03/10/2011	03/10/2011
14	**PAC2**	**28 dies**	**04/10/2011**	**31/10/2011**
15	**Estudi plataforma Moodle**	**7 dies**	**04/10/2011**	**10/10/2011**
16	Recerca en profunditat d'informació sobre Moodle	6 dies	04/10/2011	09/10/2011
17	Elaboració document Síntesi Moodle (API, funcionalitats…)	1 dia	10/10/2011	10/10/2011
18	**Preparació entorn de desenvolupament i proves**	**5 dies**	**11/10/2011**	**15/10/2011**
19	Descàrrega programari	1 dia	11/10/2011	11/10/2011
20	Instal·lació programari	1 dia	12/10/2011	12/10/2011
21	Proves programari	3 dies	13/10/2011	15/10/2011
22	**Anàlisi**	**8 dies**	**11/10/2011**	**18/10/2011**
23	Requisits	1 dia	11/10/2011	11/10/2011
24	Especificacions	4 dies	12/10/2011	15/10/2011
25	Diagrames necessaris (casos d'ús, col·laboració…)	2 dies	16/10/2011	17/10/2011
26	Elaboració document Anàlisi	1 dia	18/10/2011	18/10/2011
27	**Formació Desenvolupament Android**	**5 dies**	**17/10/2011**	**21/10/2011**
28	Formació dia 1	1 dia	17/10/2011	17/10/2011
29	Formació dia 2	1 dia	19/10/2011	19/10/2011
30	Formació dia 3	1 dia	21/10/2011	21/10/2011
31	**Prototipatge**	**4 dies**	**19/10/2011**	**22/10/2011**
32	Confecció prototips d'interfícies gràfiques	3 dies	19/10/2011	21/10/2011
33	Elaboració document Prototipatge	1 dia	22/10/2011	22/10/2011
34	**Disseny**	**8 dies**	**23/10/2011**	**30/10/2011**
35	Exploració	4 dies	23/10/2011	26/10/2011
36	Diagrames necessaris (classes, seqüència, etc…)	3 dies	27/10/2011	29/10/2011
37	Elaboració document Disseny	1 dia	30/10/2011	30/10/2011
38	Revisió document PAC2	1 dia	31/10/2011	31/10/2011
39	Lliurament PAC2	0 dies	31/10/2011	31/10/2011

ID	Tasca	Durada	Inici	Final
40	**PAC3**	**49 dies**	**01/11/2011**	**19/12/2011**
41	**Desenvolupament Aplicació**	**43 dies**	**01/11/2011**	**13/12/2011**
42	Desenvolupament	32 dies	01/11/2011	02/12/2011
43	Preparació jocs de proves	2 dies	03/12/2011	04/12/2011
44	Proves unitàries	4 dies	05/12/2011	08/12/2011
45	Proves d'integració	4 dies	09/12/2011	12/12/2011
46	Elaboració document Aplicació	1 dia	13/12/2011	13/12/2011
47	**Desplegament i instal·lació**	**4 dies**	**14/12/2011**	**17/12/2011**
48	Desenvolupament	2 dies	14/12/2011	15/12/2011
49	Proves	1 dia	16/12/2011	16/12/2011
50	**Elaboració Manual Instal·lació**	**1 dia**	**17/12/2011**	**17/12/2011**
51	Manual d'usuari	1 dia	18/12/2011	18/12/2011
52	Elaboració Manual d'Usuari	1 dia	18/12/2011	18/12/2011
53	Revisió document PAC3	1 dia	19/12/2011	19/12/2011
54	Lliurament PAC3	0 dies	19/12/2011	19/12/2011
55	**Lliurament final**	**21 dies**	**20/12/2011**	**09/01/2012**
56	**Memòria i presentació**	**20 dies**	**20/12/2011**	**08/01/2012**
57	Elaboració memòria	10 dies	20/12/2011	29/12/2011
58	Elaboració presentació virtual	10 dies	30/12/2011	08/01/2012
59	Revisió documents Lliurament Final	1 dia	09/01/2012	09/01/2012
60	Lliurament FINAL	0 dies	09/01/2012	09/01/2012
61	**Debat virtual**	**4 dies**	**22/01/2012**	**26/01/2012**
62	Debat	4 dies	22/01/2012	26/01/2012

Taula 5 – Temporització

1.5.4. Diagrama de Gannt

Id	Task Name	Duración	Comienzo	Fin
1	**PFC xarxes - Aplicació Android per Moodle**	**127,88 días**	**mié 21/09/11**	**jue 26/01/12**
2	Inici semestre	0 días	mié 21/09/11	mié 21/09/11
3	Lectura i comprensió del pla docent	1 día	mié 21/09/11	mié 21/09/11
4	**PAC1**	**12 días**	**jue 22/09/11**	**lun 03/10/11**
5	Recerca d'informació sobre PFC	2 días	jue 22/09/11	vie 23/09/11
6	Decisió temàtica del projecte	4 días	sáb 24/09/11	mar 27/09/11
7	Recerca preliminar d'informació sobre Moodle	3 días	mié 28/09/11	vie 30/09/11
8	**Pla de treball**	**3 días**	**sáb 01/10/11**	**lun 03/10/11**
9	Trobada presencial	1 día	sáb 01/10/11	sáb 01/10/11
10	Identificar objectius del projecte	1 día	sáb 01/10/11	sáb 01/10/11
11	Identificar fases, tasques, riscos, recursos…	1 día	dom 02/10/11	dom 02/10/11
12	Revisió document PAC1	1 día	lun 03/10/11	lun 03/10/11
13	Lliurament PAC1	0 días	lun 03/10/11	lun 03/10/11
14	**PAC2**	**28 días**	**mar 04/10/11**	**lun 31/10/11**
15	**Estudi plataforma Moodle**	**7 días**	**mar 04/10/11**	**lun 10/10/11**
16	Recerca en profunditat d'informació sobre Moodle	6 días	mar 04/10/11	dom 09/10/11
17	Elaboració document Síntesi Moodle (API, funcionalita	1 día	lun 10/10/11	lun 10/10/11
18	**Preparació entorn de desenvolupament i proves**	**5 días**	**mar 11/10/11**	**sáb 15/10/11**
19	Descàrrega programari	1 día	mar 11/10/11	mar 11/10/11
20	Instal·lació programari	1 día	mié 12/10/11	mié 12/10/11
21	Proves programari	3 días	jue 13/10/11	sáb 15/10/11
22	**Anàlisi**	**8 días**	**mar 11/10/11**	**mar 18/10/11**
23	Requisits	1 día	mar 11/10/11	mar 11/10/11
24	Especificacions	4 días	mié 12/10/11	sáb 15/10/11
25	Diagrames necessaris (casos d'ús, col·laboració…)	2 días	dom 16/10/11	lun 17/10/11
26	Elaboració document Anàlisi	1 día	mar 18/10/11	mar 18/10/11
27	**Formació Desenvolupament Android**	**5 días**	**lun 17/10/11**	**vie 21/10/11**
28	Formació dia 1	1 día	lun 17/10/11	lun 17/10/11
29	Formació dia 2	1 día	mié 19/10/11	mié 19/10/11
30	Formació dia 3	1 día	vie 21/10/11	vie 21/10/11
31	**Prototipatge**	**4 días**	**mié 19/10/11**	**sáb 22/10/11**
32	Confecció prototips d'interfícies gràfiques	3 días	mié 19/10/11	vie 21/10/11
33	Elaboració document Prototipatge	1 día	sáb 22/10/11	sáb 22/10/11
34	**Disseny**	**8 días**	**dom 23/10/11**	**dom 30/10/11**
35	Exploració	4 días	dom 23/10/11	mié 26/10/11
36	Diagrames necessaris (classes, seqüència, etc…)	3 días	jue 27/10/11	sáb 29/10/11
37	Elaboració document Disseny	1 día	dom 30/10/11	dom 30/10/11
38	Revisió document PAC2	1 día	lun 31/10/11	lun 31/10/11
39	Lliurament PAC2	0 días	lun 31/10/11	lun 31/10/11
40	**PAC3**	**49 días**	**mar 01/11/11**	**lun 19/12/11**
41	**Desenvolupament Aplicació**	**43 días**	**mar 01/11/11**	**mar 13/12/11**
42	Desenvolupament	32 días	mar 01/11/11	vie 02/12/11
43	Preparació jocs de proves	2 días	sáb 03/12/11	dom 04/12/11
44	Proves unitàries	4 días	lun 05/12/11	jue 08/12/11
45	Proves d'integració	4 días	vie 09/12/11	lun 12/12/11
46	Elaboració document Aplicació	1 día	mar 13/12/11	mar 13/12/11
47	**Desplegament i instal·lació**	**4 días**	**mié 14/12/11**	**sáb 17/12/11**
48	Desenvolupament	2 días	mié 14/12/11	jue 15/12/11
49	Proves	1 día	vie 16/12/11	vie 16/12/11
50	Elaboració Manual Instal·lació	1 día	sáb 17/12/11	sáb 17/12/11
51	**Manual d'usuari**	**1 día**	**dom 18/12/11**	**dom 18/12/11**
52	Elaboració Manual d'Usuari	1 día	dom 18/12/11	dom 18/12/11
53	Revisió document PAC3	1 día	lun 19/12/11	lun 19/12/11
54	Lliurament PAC3	0 días	lun 19/12/11	lun 19/12/11
55	**Lliurament final**	**21 días**	**mar 20/12/11**	**lun 09/01/12**
56	**Memòria i presentació**	**20 días**	**mar 20/12/11**	**dom 08/01/12**
57	Elaboració memòria	10 días	mar 20/12/11	jue 29/12/11
58	Elaboració presentació virtual	10 días	vie 30/12/11	dom 08/01/12
59	Revisió documents Lliurament Final	1 día	lun 09/01/12	lun 09/01/12
60	Lliurament FINAL	0 días	lun 09/01/12	lun 09/01/12
61	**Debat virtual**	**4 días**	**dom 22/01/12**	**jue 26/01/12**
62	Debat	4 días	dom 22/01/12	jue 26/01/12

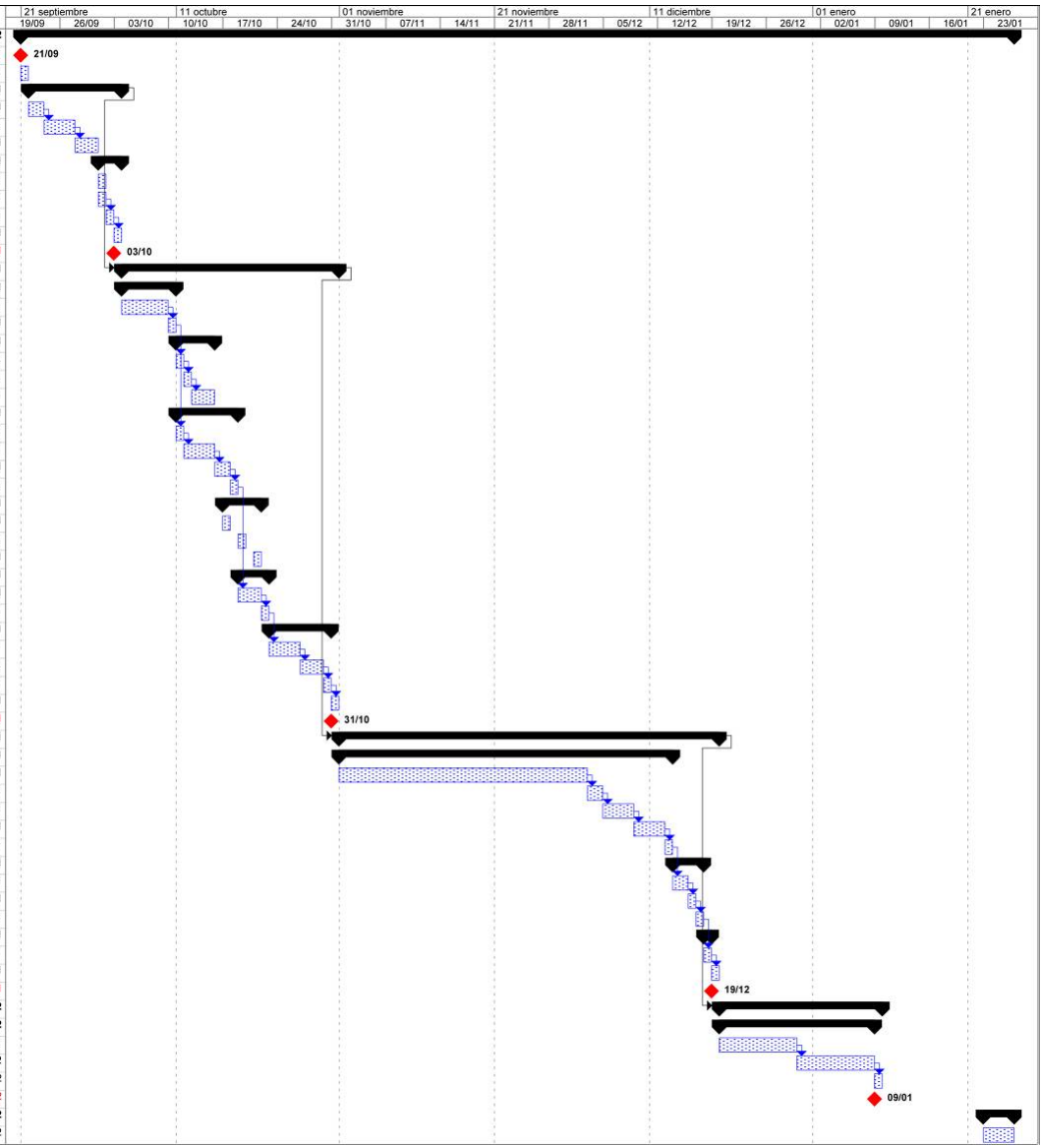


Figura 2 – Diagrama de Gannt

1.6. Producte

Els lliurables resultants del desenvolupament d'aquest projecte són:

- Document "PAC1 – Pla de treball"
- Document "PAC2 – Anàlisi i disseny"
- Document "PAC3 – Implementació"
- Aplicació "Moodle per Android"
- Document "Memòria" (aquest document)
- Document "Manuals"
- Presentació virtual

Com a complement de l'aplicació "Moodle per Android" s'inclouen les següents llibreries externes:

- Llibreries "MoodleWS_gson"[1]
- Llibreries "Google Gson"[2]

Addicionalment, com a subproducte del projecte s'obté:

- Servidor Moodle "PFC Moodle Manel Herrera" necessari per validar i provar l'aplicació amb el *plugin* "OK Tech Web Services"[3] instal·lat

[1] Llibrerires MoodleWS_gson https://github.com/patrickpollet/moodlews_gson
[2] Google-Gson Java library http://code.google.com/p/google-gson/
[3] Plugin "OK Tech Web Services" https://github.com/patrickpollet/moodlews

1.7. Estructura del document

Els següents capítols d'aquesta memòria s'estructuren de la següent forma:

- Anàlisi inicial i prerequisits. Es descriu el punt de partida del projecte, els requeriments inicials i s'identifiquen el riscos potencials.

- Entorn. Es descriu quin és l'entorn de desenvolupament i d'execució i es proporciona una breu introducció als sistemes, plataforma, eines i tecnologies emprades en el projecte

- Anàlisi i especificació. Es descriuen els requeriments funcionals i no funcionals de l'aplicació i es detallen els subsistemes i casos d'ús de l'aplicació

- Disseny. Es detallen els aspectes de disseny de l'arquitectura de l'aplicació, els models entitat-relació, de classes i els diagrames de col·laboració i seqüència.

- Implementació. Es descriu el procés de desenvolupament pròpiament dit, amb les seves peculiaritats, detallant, en tot moment, les decisions preses

- Conclusions i línies de futur. Aquest capítol recull les conclusions, apreciacions i experiències personals observades durant tot el projecte. Addicionalment, proposa les línies de treball que caldria seguir en el futur per ampliar o aprofundir en el projecte.

2. Anàlisi inicial i prerequisits

En aquest capítol es descriu quin es el punt de partida del projecte, quin són els requeriments inicials i es presenta la valoració de riscos realitzada a l'inici del projecte.

2.1. Punt de partida

La definició de l'objectiu general del projecte és "desenvolupar una aplicació Android per Moodle" sense que es proporcioni cap més detall, com per exemple quines funcionalitats cal implementar, quin disseny o estil han de tenir les interfícies o a quin públic ha d'anar destinada l'aplicació.

Les úniques indicacions del consultor (d'ara en endavant el client) de que disposem són que Moodle ha llançat una nova versió de la seva API i que es recomana que el desenvolupador disposi d'un dispositiu Android.

D'aquest dos aspectes, la definició de l'objectiu del projecte i les indicacions del client, malgrat no donen masses detalls, ja se'n pot extreure una imatge de l'escenari de partida i una guia de quines línies generals haurà de seguir el projecte. Són les següents:

- Ha de ser una aplicació per Android, la qual cosa ja estableix per quin tipus de dispositiu es farà servir.

- El fet que el sistema seleccionat sigui Android defineix el llenguatge de programació i les eines que es faran servir pel desenvolupament.

- Serà necessari formar-se en el desenvolupament d'aplicacions mòbils per Android.

- Ha de connectar a un servidor Moodle, fet que d'una banda ens determina a alt nivell quin tipus de funcionalitats podria implementar l'aplicació i de l'altra ens diu el públic al que potencialment anirà destinada l'aplicació, personal docent i alumnes que han triat l'*e-learning* com a mitjà formatiu i que disposen de dispositiu mòbils Android, la qual cosa pressuposa una certa habilitat en l'ús d'aquest tipus de dispositius i entorns.

- Les indicacions del client referents a la nova API que ha llançat Moodle en la seva darrera versió també ens estableixen el mètode de comunicació que caldrà fer servir entre l'aplicació Android i el servidor Moodle.

- D'altra banda, caldrà investigar quins serveis ofereix l'API i quins protocols de comunicació accepta per tal de definir les funcionalitats que cal implementar i els mecanismes de comunicació que cal fer servir.

- Haver de connectar a un servidor Moodle implica que serà imprescindible tenir un servidor disponible per les proves de funcionament, per tant, caldrà investigar la possibilitat de fer servir un servidor públic o bé instal·lar-ne un de propi.

- El client també recomana que pel desenvolupament es disposi d'un dispositiu mòbil Android.

- Tant l'àrea del projecte, Xarxes de Computadors, i més concretament l'ús de comunicacions sense fils en dispositius mòbils, determinen que les comunicacions es realitzaran fent ús d'aquesta tecnologia.

D'aquestes línies es poden extreure les següents conclusions:

- És necessària formació específica en desenvolupament per a dispositius mòbils Android.
- Serà imprescindible preparar un entorn de desenvolupament estable.
- És necessari un coneixement bàsic en el funcionament de servidor Moodle per tal d'utilitzar un servidor públic o instal·lar-ne un de propi, si s'escau.
- És necessari un aprofundiment en les especificacions de l'API de Moodle i els *Web Services* que ofereix per tal de definir les funcionalitats a implementar i l'abast del projecte.
- S'haurà de disposar d'un dispositiu mòbil Android. El dispositiu seleccionat serà un SAMSUNG Galaxy Tab.
- L'arquitectura de la solució es basarà en el consum per part de l'aplicació Android de serveis oferts pel servidor Moodle.
- Caldrà aprofundir en la utilització de protocols de serveis web.
- Els possibles usuaris són el personal docent, els alumnes i els propis administradors del sistema

Com es pot observar, aquest recull de requeriments i conclusions és molt genèric i depèn de l'estudi de la plataforma Moodle, per tant, una part important de l'inici del projecte consistirà en realitzar aquesta investigació preliminar per tal de definir els requeriments funcionals i no funcionals que no han estat especificats.

2.2. Riscos

La següent taula recull els riscos associats al projecte identificats en la fase d'anàlisi inicial. Per cadascun es fa una valoració de quina probabilitat té que es presenti, el seu possible impacte sobre la consecució de l'objectiu del projecte, quines accions es poden prendre per tal de mitigar-lo i quin seria el nou impacte després d'haver dut a terme les accions proposades.

Risc	Conseqüència	Proba-bilitat	Impacte	Acció mitigadora	Nou impacte
Manca de coneixements de l'API de Moodle a l'hora de definir els objectius del projecte	Objectius poc ambiciosos, poc realistes o inassolibles	Alta	Alt	Estudiar l'API de Moodle el més aviat possible per ajustar objectius	Mig
Manca de coneixements de desenvolupament per Android	Dificultat o impossibilitat de desenvolupar l'aplicació	Mitja	Alt	Formar-se en desenvolupament Android. Cerca de manuals, tutorials, fòrums, exemples...	Baix
Dificultat per comprendre el funcionament de Moodle (instal·lació, administració, configuració de servidor...)	Dificultat per tenir un servidor disponible per la connexió de l'aplicació i realització de les proves	Mitja	Alt	Cercar manuals, tutorials, fòrums, exemples...	Mig
Maquinari inadequat , antic o amb recursos insuficients	Dificultat per instal·lar i executar el programari necessari amb un rendiment acceptable	Mitja	Mig	Adquirir nou maquinari	Baix
Pèrdua total o parcial del maquinari o programari emprat en el projecte	Dificultat per complir amb els terminis de lliurament. Haver de reinstal·lar l'entorn.	Baixa	Mig	Crear imatges i còpies de seguretat	Baix
Pèrdua total o parcial del producte en desenvolupament	Impossibilitat de complir amb els terminis de lliurament. Haver d'iniciar de nou el projecte.	Baixa	Alt	Còpies de seguretat	Baix

Taula 6 – Riscos inicials del projecte

2.3. Estudi preliminar de Moodle

Del primer contacte amb la plataforma Moodle versió 2 i dels *Web Services* oferts en la seva API es poden destacar els següents aspectes:

- És una plataforma madura amb una forta implantació i una bona reputació en la comunitat d'*e-learning.*
- Basa tot el seu funcionament en tecnologies de codi obert amb llicència GPL.
- Hi ha servidors de test a disposició del públic per facilitar la realització de proves, amb l'inconvenient que cada hora s'inicialitzen amb la qual cosa és recomanable disposar d'un servidor propi.
- Hi ha paquets auto-instal·lables que inclouen tots els element necessaris per posar en marxa un servidor Moodle de forma senzilla.
- Els *Web Services* de Moodle són una funcionalitat nova apareguda en la versió 2 que no proporcionen documentació pel desenvolupament d'aplicacions en Java o .NET que els consumeixin.

- Permet l'ús de protocols com AMF, SOAP, REST i XML-RPC per realitzar les crides als *Web Services*.

- Disposa de documentació bàsica referent als mètodes que s'ofereixen a la seva API.

Consegüentment, ens trobem, d'una banda, amb una plataforma lliure, estable, ben documentada pel que es refereix a l'aplicació en si, fàcilment instal·lable i que disposa d'una àmplia comunitat de contribuïdors que hi donen suport; i de l'altra, amb uns *Web Services* en estat incipient que no proporcionen una API ben documentada pel desenvolupament en Java.

2.4. Requeriments funcionals

Una vegada realitzat l'estudi preliminar de Moodle ja es disposa de la informació necessària per completar l'anàlisi inicial i definir els aspectes funcionals de l'aplicació, els quals s'ajustaran en la posterior fase d'anàlisi.

2.4.1. Usuaris

Els possibles usuaris previstos són:

- Administradors. Tenen control total. Poden crear, consultar, modificar i eliminar usuaris, cursos i matriculacions.

- Personal docent. Tenen control sobre les dades del seu compte d'usuari i sobre cursos i matriculacions.

- Alumnes. Poden modificar les dades del seu compte d'usuari i les seves matrícules.

Cal remarcar que els privilegis de cada usuari els determina Moodle, per tant, l'aplicació no implementarà cap control sobre els permisos de l'usuari, essent Moodle el que ho gestioni i informi a l'usuari quan intenti fer una acció per la qual no té autorització.

2.4.2. Funcionalitats

Havent analitzat els mètodes que ofereix la seva API es decideix la implementació d'un subconjunt d'aquests mètodes, els quals es fixen com a abast del projecte.

El nombre total de mètodes principals que s'ofereixen a l'API de Moodle és de 23 i es pot consultar a l'Annex I [4]. No obstant això, finalment l'aplicació només n'implementarà 10, proposant-se la resta com a treball futur a la finalització d'aquest projecte.

La següent taula recull els mètodes implementats amb una breu descripció del seu funcionament:

Mètode	Descripció
user:	
create_users()	Permet la creació d'un usuari nou indicant les seves dades bàsiques
delete_users()	Permet l'eliminació d'un usuari existent
update_users()	Permet l'actualització de les dades d'un usuari existent
course:	
get_courses()	Permet obtenir la llista de cursos disponibles a Moodle
create_courses()	Permet crear un curs nou amb les seves dades bàsiques
delete_courses()	Permet eliminar un curs existent
update_courses()	Permet actualitzar un curs existent
enrol:	
get_roles()	Permet obtenir la llista de cursos als que s'ha matriculat un usuari concret
role_assign()	Permet matricular un usuari a un curs
role_unassign()	Permet eliminar la matrícula d'un usuari a un curs

Taula 7 – Mètodes API implementats

Les justificacions per les quals es decideix implementar només una part de les possibles funcionalitats que s'ofereixen són dues:

- Els recursos del projecte no són suficients per poder completar tots els mètodes dins el termini de lliurament del projecte. Caldria augmentar els recursos humans o temporals del projecte o buscar recursos externs; cap d'aquestes possibilitats és factible.

- Hi ha alguns mètodes oferts per l'enviament de missatges i per la càrrega/descàrrega de fitxers que per ser implementats requeririen la instal·lació i configuració de servidors de correu o servidors de fitxers, la qual cosa queda fora de l'abast d'aquest projecte.

No obstant això, el criteri per seleccionar els mètodes que s'ha decidit implementar ha seguit unes pautes clares amb l'objectiu d'assegurar que l'aplicació estigui dotada d'un conjunt complert de funcionalitats que a la vegada sigui rica en la utilització d'elements de disseny.

[4] Annex I – Descripció bàsica de l'API de Moodle

Aquestes pautes són les següents:

- Les funcionalitats implementades han de permetre realitzar accions bàsiques als tres potencials tipus d'usuari: administradors, personal docent i alumnes.

- Ha de permetre la realització d'un conjunt complert d'instruccions de la convenció *CRUD* d'interfícies d'usuari (creació, lectura, modificació i eliminació).

- Per tal de donar riquesa a la interfície ha de permetre la utilització de diferents elements de disseny d'interfícies Android, com per exemple: camps de text, camps d'edició, desplegables, llistes, botons, quadres de diàleg, missatges emergents, menús, etc.

2.4.3. Seguretat

L'aplicació ha de ser accessible només per aquells usuaris que hi estiguin autoritzats, amb la qual cosa, a banda dels mètodes relacionats a l'apartat anterior, cal desenvolupar funcionalitats addicionals per iniciar i finalitzar una sessió. L'inici de sessió es farà amb usuari i contrasenya.

Mètode	Descripció
login()	Permet iniciar sessió a un usuari fent servir usuari i contrasenya
logout()	Permet finalitzar una sessió iniciada

Taula 8 – Mètodes d'autenticació implementats

Cal remarcar que Moodle, de forma nativa, també ofereix la possibilitat d'iniciar sessió mitjançant l'ús de *tokens*. No obstant això, l'ús d'aquest mecanisme d'autenticació està desaconsellat atès que l'identificador *token* queda guardat en el dispositiu Android creant un risc per la seguretat.

2.5. Requeriments no funcionals

2.5.1. Interfície

L'aplicació haurà de proporcionar una interfície d'usuari que satisfaci les necessitats imposades per les funcionalitats a implementar i per les característiques del dispositiu, tenint en compte l'entorn d'ús, atès que els dispositius mòbils, per la seva naturalesa, s'acostumen a fer servir des de qualsevol lloc.

Els requeriments bàsics són:

- Una interfície capaç de realitzar les funcionalitats implementades: inici/finalització de sessió, manteniment d'usuaris, manteniment de cursos i matriculacions.

- Una interfície simple i intuïtiva per tal de minimitzar el procés d'aprenentatge dels usuaris. Per assolir aquest objectiu la interfície es dissenyarà seguint els principis de disseny centrat en l'usuari. Per exemple, mostrarà només els elements necessaris en cada moment eliminant tots aquells que siguin superflus, farà servir les metàfores adients, facilitarà la navegació mitjançant la visualització permanent de la ruta en la que es troba l'usuari, donarà missatges clars, retroalimentació, etc...

- Una interfície que no faci servir combinacions de colors o caràcters estranys i que sigui clarament visible en qualsevol condició d'iluminació.

- Una interfície pensada per dispositius amb pantalla tàctil i amb dimensions de pantalla reduïdes.

- Una interfície multi-idioma

Un dels inconvenients que es preveuen és que el mercat actual ofereix un nombre molt elevat de dispositius Android que tenen diferents característiques en quant a dimensions i resolució de pantalla. Malgrat que les eines de desenvolupament d'Android proporcionen mecanismes per minimitzar la necessitat de dissenyar diferents interfícies segons el dispositiu, l'única forma d'assegurar el correcte funcionament de l'aplicació en qualsevol dispositiu seria provar-la per totes i cadascuna de les possibles combinacions de dimensions, resolucions i orientacions de pantalla. Atès que això requeriria uns recursos dels que en aquest projecte no es disposa, la interfície es dissenyarà únicament pel dispositiu triat, el SAMSUNG Galaxy Tab, deixant com a treball futur l'adaptació a altres dispositius.

2.5.2. Seguretat

Tal com s'ha explicat a l'apartat 2.4.1., els privilegis de cada usuari els determina Moodle i l'aplicació no implementarà cap control sobre els permisos de l'usuari. Quan un usuari intenti fer una acció per la qual no té autorització serà Moodle el que li denegui i l'informi amb el corresponent avís.

La justificació per fer-ho d'aquesta forma és que l'API de Moodle no proporciona cap mètode per obtenir el detall dels privilegis d'un usuari Moodle. Consegüentment, no és possible implementar aquest control a menys que s'implementi una gestió de seguretat paral·lela a l'aplicació Android amb el risc que això comporta, és a dir que es corre el perill que els privilegis a Moodle difereixin dels de l'aplicació Android i el resultat sigui que un usuari tingui permís fer unes accions a Moodle que no pot fer des de l'aplicació Android, generant frustració a l'usuari i desconfiança en l'aplicació.

3. Entorn

3.1. Història

3.1.1. La plataforma Moodle

Moodle (*Module Object-Oriented Dynamic Learning Environment*) és un Sistema de Gestió de Recursos de Codi Obert (*Open Source Course Management System, CMS*), conegut també com Sistema de Gestió de l'Aprenentatge (*Learning Management System, LMS*) o com Entorn d'Aprenenetage Virtual (*Virtual Learning Environment, VLE*).

Va néixer en els anys 90 com una idea de Martin Dougiamas, *webmaster* i administrador de sistemes a la "Curtin University of Technology", però no va ser fins el 20 d'agost de 2002 que Dougiamas va llançar la versió 1.0. El 2003 es va fundar la companyia "moodle.com"[5] per donar suport comercial i altres serveis, com ara hosting i consultoria, i avui en dia ja està estesa a més de 40 països formant la "Moodle Partner Network"[6].

Moodle, programari lliure de codi obert, segueix actiu i en contínua evolució gràcies al projecte "moodle.org"[7] que proporciona un centre d'informació, discussió i col·laboració entre els seus usuaris, administradors, educadors i desenvolupadors.

Actualment s'ha convertit en la solució més popular entre els educadors d'arreu del món com una eina per crear llocs web dinàmics en línea per als seus estudiants.

3.1.2. El sistema Android

Android és un sistema operatiu per a dispositiu mòbils desenvolupat per l'*Open Handset Alliance*[8], consorci de més de 70 fabricants, operadors i desenvolupadors de maquinari i programari liderat per Google.

Va ser creat inicialment per *Android Inc*, companyia que va ser adquirida per Google l'any 2005 i a finals de 2007 es va anunciar el llançament del sistema operatiu Android sota la llicència lliure de codi obert Apache.

L'estructura del sistema operatiu està composada per aplicacions que s'executen en un *framework* Java sobre un nucli Linux de llibreries escrites en C/C++ en una màquina virtual anomenada Dalvik, similar a la coneguda maquina virtual de Java però adaptada a les limitacions imposades pels dispositius mòbils pel que es refereix a ús de la memòria, la CPU i a la gestió de l'energia per tal de preservar la vida de la bateria. Les llibreries del sistema proporcionen, entre d'altres, accés a base de dades SQLite, a gràfics, 3D i a medis.

[5] Companyia moodle.com: http://moodle.com/
[6] Projecte moodle.org: http://moodle.org/
[7] Moodle Partner Network: http://moodle.com/partners/
[8] Consorci Open Handset Alliance http://www.openhandsetalliance.com/index.html

Android ha experimentat un gran creixement en la quota de mercat de dispositius mòbils arribant a superar a Apple, el seu principal competidor en quan a nombre de terminals activats i *apps* descarregades de *l'Android Market*, botiga *on-line* d'aplicacions Android.

Alguns dels principals fabricants de dispositius Android són Samsung, HTC, Sony Ericsson, Motorola o LG. La versió més recent, presentada oficialment l'octubre de 2011 és la 4.0, també coneguda com *Icecream Sandwich*. Com a curiositat, Android posa el nom (en anglès) d'uns postres a cada versió que llença al mercat seguint un ordre alfabètic. Les anteriors han estat *Cupcake, Donut, Eclair, Froyo, Gingerbread* i *Honeycomb*.

3.2. Xarxes sens fils

El funcionament es basarà, per sobre de tot, en l'ús intensiu de les xarxes sense fils les quals possibilitaran la connexió del dispositiu mòbil Android al servidor Moodle.

La següent figura il·lustra tres possibilitats de connexió:

- En primer lloc, el mòbil 1 es connecta a través d'Internet mitjançant un operador de telefonia mòbil 3G de forma que pot arribar a accedir al servidor Moodle.

- En el cas del mòbil 2, fent ús d'una xarxa local sense fils WiFi, el dispositiu és connecta a Internet i d'aquí al servidor Moodle.

- Finalment, el mòbil 3 es connecta directament via WiFi a la mateixa xarxa local on es troba el servidor Moodle.

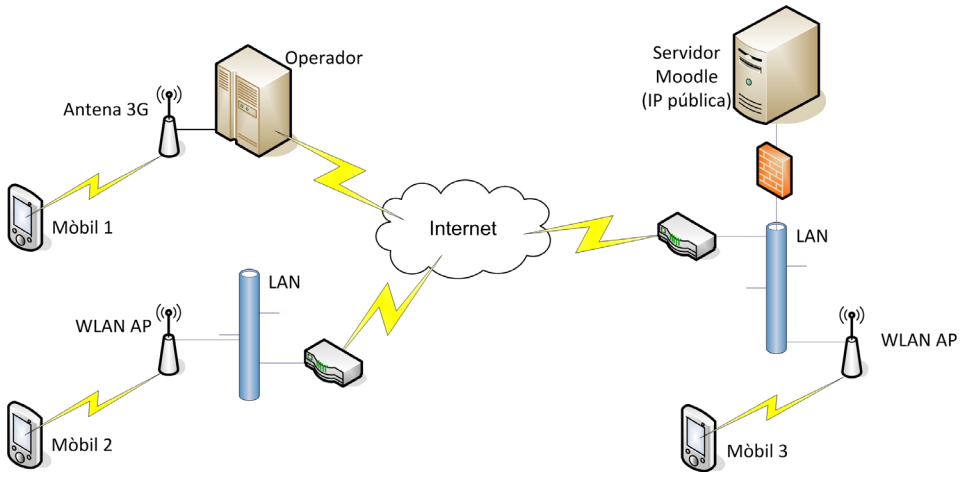


Figura 3 – Exemples de connexió sense fils

NOTA: Cal destacar que la figura anterior il·lustra les possibilitats de connexió de forma senzilla sense entrar en detalls d'encaminament, seguretat perimetral, etc. Molt probablement, en un entorn real el servidor estaria protegit per una DMZ o altres sistemes més segurs que un simple tallafocs.

3.3. Entorn de desenvolupament i execució

3.3.1. Sistema operatiu

El sistema operatiu on es desenvolupa aquest projecte és Microsoft Windows. Es disposa d'un ordinador portàtil amb Windows 7 que es destina a desenvolupament i de dos equips amb Windows XP que executaran una instància del servidor Moodle cadascun.

3.3.2. Dispositiu mòbil

El dispositiu que es farà servir per aquest projecte és un SAMSUNG Galaxy Tab.

Figura 4 – SAMSUNG Galaxy Tab
Font: http://www.samsung.com/es/microsite/galaxytab/

3.3.3. Moodle

3.3.3.1. Instal·lació del servidor Moodle

Inicialment, per tal d'evitar la complicació d'haver d'instal·lar un servidor Moodle, es va explorar la possibilitat de fer servir algun dels servidors que moodle.org posa a disposició dels usuaris. Malauradament aquesta possibilitat es va descartar atès que els servidors de proves es reinicialitzen cada hora per tal d'assegurar un entorn de proves estable, per tant, qualsevol canvi en dades o configuració es perd.

Consegüentment, es decideix realitzar una instal·lació d'un servidor Moodle que finalment resulta ser senzilla gràcies als paquets auto-instal·lables que hi ha disponibles a l'àrea de descàrregues[9] de la web de Moodle. Aquests paquets, creats amb XAMPP, proporcionen una instal·lació guiada del programari del servidor Moodle a més de tots els elements necessaris

[9] Àrea de descàrregues Moodle http://moodle.org/downloads/

per que aquest funcioni: Apache, MySQL, PHP i Perl. Una vegada finalitzada ja es pot començar a fer servir immediatament.

Per una millor gestió de recursos i per dotar aquest entorn de desenvolupament de més flexibilitat, es decideix instal·lar dos servidors en equips separats:

- El primer està ubicat a la xarxa local privada on es realitza el desenvolupament i només és accessible pel desenvolupador.

- El segon està ubicat en una xarxa pública i és accessible des d'Internet pel públic en general.

3.3.3.2. Configuració dels *Web Services*

El següent pas de la preparació de l'entorn Moodle és la configuració dels *Web Services* per tal de poder fer crides als mètodes de l'API. En primer lloc cal determinar quin protocol es farà servir per establir la comunicació i realitzar les crides des de la nostra aplicació Android. Moodle ofereix quatre possibilitats:

- AMF (Flash)

- SOAP (Només PHP)

- REST (Qualsevol llenguatge)

- XML-RPC

Addicionalment, Moodle té el *plugin* "OK Tech Web Services"[10] que proporciona funcionalitats addicionals per la utilització dels protocols SOAP o REST, que són els que Moodle recomana fer servir pel desenvolupament d'aplicacions Java, i més concretament pel desenvolupament d'aplicacions Android.

Relacionats amb aquest *plugin*, hi ha disponibles una sèrie de llibreries de classes Java[11] que faciliten el desplegament d'aplicacions que utilitzin el protocol REST amb format de sortida JSON. El seu funcionament es basa en la serialització/des-serialització de dades fent servir les llibreries "Google Gson"[12], les quals faciliten la conversió de classes Java cap a la seva representació en format JSON i també converteixen cadenes JSON a les seves classes Java equivalents.

Consegüentment, després de valorar les alternatives que Moodle ofereix per fer crides als seus *Web Services* i a la vista que ja existeixen *plugins* i llibreries que ajuden al desenvolupament

[10] Plugin "OK Tech Web Services" http://docs.moodle.org/20/en/Web_Services:OK_Tech_Web_Services
[11] Llibrerires MoodleWS_gson https://github.com/patrickpollet/moodlews_gson (Requereix autenticació Github)
[12] Google-Gson Java library http://code.google.com/p/google-gson/

d'aplicacions Java per Android que facin servir el protocol REST[13] i el format JSON[14], es decideix triar aquest protocol i aquest format atenent a les següents raons:

- Moodle no ofereix nativament una API per Java però disposa del plugin "OK Tech Web Services" que ofereix funcionalitats addicionals per aquesta finalitat.
- L'enviament de dades amb el protocol REST es fa en format JSON.
- Google disposa de les llibreries Java "Google Gson" per la conversió entre classes Java i format JSON i viceversa.
- Google recomana l'ús de "Google Gson" en aplicacions Android.
- Hi ha disponibles llibreries Java per l'ús de "OK Tech Web Services" amb REST i JSON.

Tal com s'il·lustra a la següent figura, l'aplicació Java que s'executa en el dispositiu Android converteix els missatges a format JSON mitjançant les llibreries "MoodleWS_Gson" que al seu torn fan servir les llibreries "Google GSon", i tot seguit els envia. A continuació Moodle els interpreta gràcies al *plugin* "OK Tech Web Services" el qual comunica amb el servidor Moodle fent servir els *Web Services* d'aquest. El procés de resposta del servidor cap al dispositiu Android és el mateix però en sentit invers.

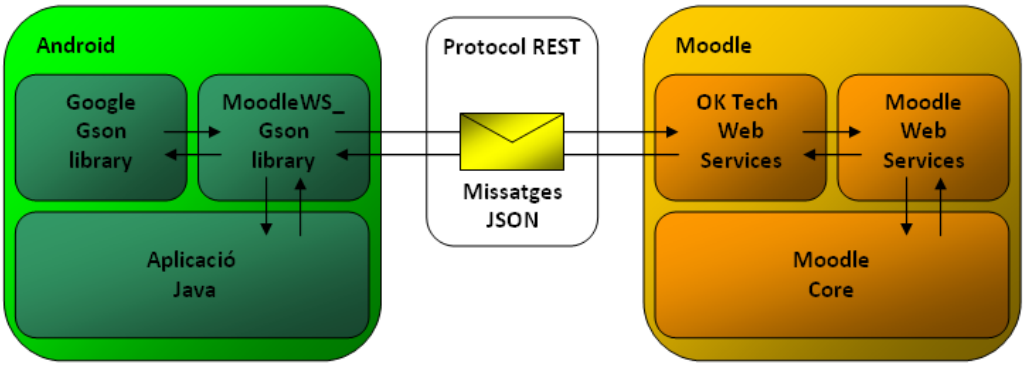


Figura 5 – Funcionament del protocol REST amb JSON

El manual "Instal·lació i configuració de l'entorn Moodle"[15] proporciona una guia per la instal·lació i configuració del servidor Moodle i del seus *Web Services*.

[13] A l'annex II es pot trobar una descripció detallada del protocol REST
[14] A l'annex III es pot trobar una descripció detallada del format JSON
[15] Veure el capítol "Instal·lació i configuració de l'entorn Moodle" del document "Manuals"

3.3.4. Android

3.3.4.1. Entorn de desenvolupament Java

Les aplicacions per Android es desenvolupen en llenguatge Java atès que els dispositius Android disposen d'una màquina virtual anomenada "Dalvik Virtual Machine" –similar a la "Java Virtual Machine" que es pot trobar per altres plataformes– que és l'encarregada d'interpretar el codi Java precompilat.

Per tal de desenvolupar aplicacions en llenguatge Java és necessari disposar, com a mínim, dels següents elements:

- *Java Runtime Environment* (JRE) – Entorn d'execució Java (Java Virtual Machine)
- *Java Development Kit* (JDK) – Eines de desenvolupament Java (Llibreries)
- *Integrated Developement Environment* (IDE) - Programari d'ajuda al desenvolupament

L'entorn i les eines Java es poden trobar al web d'Oracle[16]. Pel que fa als IDE, hi ha diferents opcions, com ara Eclipse, Intellij IDEA o Netbeans. S'ha seleccionat Eclipse[17] per raons d'experiència prèvia i perquè per Eclipse hi ha utilitats disponibles que faciliten el desenvolupament per Android.

3.3.4.2. Eines de desenvolupament per Android

Per desenvolupar aplicacions Java per Android existeix un conjunt d'utilitats i llibreries que donen resposta a les necessitats de desenvolupament d'aquests dispositius. Aquestes eines es poden descarregar[18] i incorporar fàcilment a l'Eclipse amb la instal·lació dels següents elements:

- *Android Software Development Kit* (Android SDK) – Eines de desenvolupament per dispositius Android (Llibreries)
- *Android Development Tools* (ADT) – Utilitats per Eclipse per crear interfícies gràfiques, depurar codi i desplegar aplicacions
- *Android Virtual Device Manager* (AVD Manager) – Emulador de dispositius Android
- API pel dispositiu Samsung Galaxy Tab

[16] Java downloads: http://www.oracle.com/technetwork/java/javase/downloads/index.html
[17] Eclipse downloads: http://www.eclipse.org/downloads/
[18] Android SDK downloads: http://developer.android.com/sdk/index.html

Addicionament, per possibilitar la connexió directa del SAMSUNG Galaxy Tab per USB, caldrà instal·lar el següent element, disponible al lloc web de Samsung[19]:

- *Samsung Kies* pel *driver* del dispositiu Samsung Galaxy Tab (alternativament es pot fer servir el *Google USB driver* per dispositius genèrics)

3.3.5. Resum

La següent figura resumeix l'entorn descrit en els apartats anteriors i indica quin elements hi ha instal·lats en cada equip. A més, es poden apreciar les diferents tecnologies de comunicacions sense fils i com el mateix dispositiu mòbil pot accedir als dos servidors Moodle de diferent forma:

- Mitjançant Wifi al servidor Moodle privat directament per la xarxa local
- Mitjançant WiFi al servidor Moodle públic accedint per WiFi a la xarxa local i després sortint a Internet
- Mitjançant 3G al servidor Moodle públic directament a través d'Internet

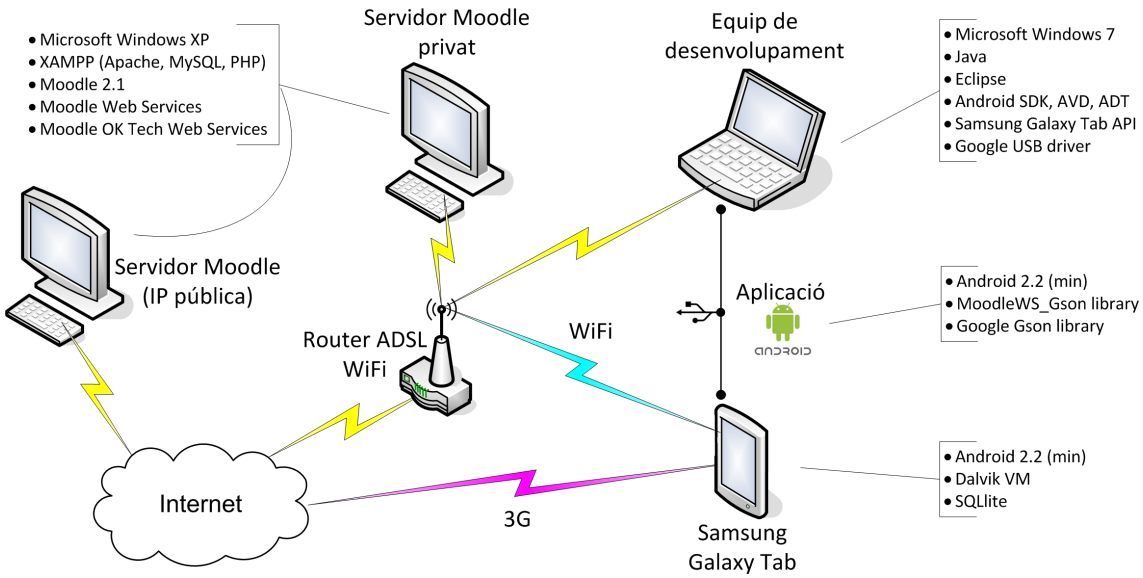


Figura 6 – Entorn de desenvolupament i proves

[19] Samsung Kies: www.samsung.com/es/kies/index.html

3.4. Eines complementàries

A banda de totes les eines i programari descrit als apartats anterior, per la realització d'aquest projecte s'ha fet servir el següent programari addicional:

- Microsoft Word per la redacció de documents
- Microsoft Power Point per l'elaboració de figures i la presentació del projecte
- Microsoft Visio per l'elaboració de diagrames i figures
- Microsoft Project per la creació de la planificació del projecte
- PDF995 per la creació documents PDF
- TechSmith Camtasia Studio per la captura de vídeo (versió d'avaluació)

4. Anàlisi i especificació

4.1. Funcionalitats del sistema

Tal com s'ha descrit a l'apartat 2.4.2. l'aplicació implementarà només un subconjunt de 10 mètodes de tots els que ofereix l'API de Moodle. No obstant això, durant l'anàlisi s'ha trobat que és necessari fer crides a altres mètodes complementaris per poder realitzar una gestió complerta de les funcionalitats seleccionades.

Malgrat això no augmenta ni disminueix el nombre de funcionalitats previstes, tot seguit es detallen aquests mètodes als que també caldrà fer crides:

Mètode	Descripció
user:	
get_my_id()	Permet obtenir l'identificador de l'usuari que ha iniciat sessió
get_user_by_id()	Permet obtenir un usuari pel seu identificador
get_user_by_name()	Permet obtenir un usuari pel seu nom
course:	
get_course()	Permet obtenir un curs pel seu nom
get_categories()	Permet obtenir la relació de categories de cursos

Taula 9 – Mètodes complementaris

Consegüentment, l'aplicació implementarà un total de:

- 10 funcionalitats pel manteniment bàsic d'usuaris, cursos i matriculacions que faran crides a 13 mètodes

- 2 funcionalitats per iniciar i finalitzar sessió que faran crides a 3 mètodes

4.2. Descripció bàsica del procés

El sistema consistirà en una aplicació Android que en iniciar-se permetrà a l'usuari modificar els paràmetres de connexió o iniciar una sessió amb un servidor Moodle d'acord a aquests paràmetres prèviament establerts.

Una vegada establerta la connexió amb el servidor i iniciada la sessió, l'usuari disposarà d'unes opcions per tal de cridar les funcionalitats de manteniment bàsic d'usuaris, cursos i matriculacions que s'han descrit a l'apartat 2.4.2.

En acabar, l'usuari tindrà l'opció de finalitzar la sessió o tancar l'aplicació, cas en que la sessió quedarà finalitzada automàticament.

4.3. Subsistemes

Per tal d'estructurar el funcionament de l'aplicació, aquesta es subdivideix en quatre subsistemes clarament diferenciats:

- Subsistema d'inici i configuració
- Subsistema d'usuaris
- Subsistema de cursos
- Subsistema de matriculació

4.3.1. Subsistema d'inici i configuració

Permet iniciar sessió i establir la configuració de l'aplicació. Les seves funcionalitats són:

1	**Inici i configuració**	
1.1	Inici de sessió	Permet connectar-se a un servidor i obtenir un identificador de sessió
1.2	Configuració de l'aplicació	Permet modificar els paràmetres de connexió
1.3	Finalització de sessió	Permet finalitzar una sessió iniciada

Taula 10 – Funcionalitats del susbsistema d'inici i configuració

4.3.2. Subsistema d'usuari

Permet gestionar els usuaris del sistema Moodle. Les seves funcionalitats són:

2	**Usuari**	
2.1	Crear usuaris	Permet crear un usuari indicant les seves dades bàsiques
2.2	Eliminar usuaris	Permet eliminar un usuari
2.3	Modificar usuaris	Permet modificar les dades d'un usuari

Taula 11 – Funcionalitats del susbsistema d'usuari

4.3.3. Subsistema de cursos

Permet gestionar els cursos del sistema Moodle. Les seves funcionalitats són:

3	**Cursos**	
3.1	Llista de cursos	Obté la relació de cursos disponibles del sistema Moodle
3.2	Crear curs	Permet crear un curs amb les seves dades bàsiques
3.3	Modificar curs	Permet modificar les dades bàsiques d'un curs
3.4	Eliminar curs	Permet eliminar un curs

Taula 12 – Funcionalitats del susbsistema de cursos

4.3.4. Subsistema de matriculació

Permet gestionar les matriculacions. Les seves funcionalitats són:

4	Matriculació	
4.1	Alta	Permet donar-se d'alta a un curs
4.2	Baixa	Permet donar-se de baixa d'un curs
4.3	Llista de cursos matriculats	Obté la relació de cursos als que un alumne està matriculat

Taula 13 – Funcionalitats del susbsistema de matriculació

4.4. Diagrama conceptual

El diagrama conceptual proporciona una visió d'alt nivell de com s'estructura l'aplicació i quines opcions de navegació hi ha entre els diferents menús i pantalles de la interfície.

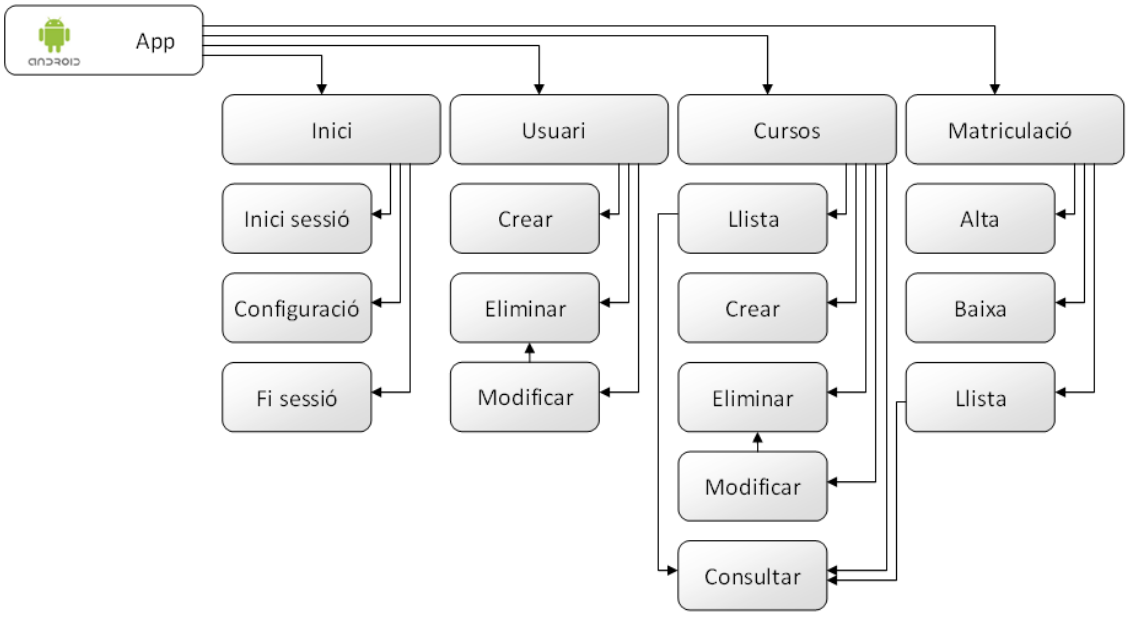


Figura 7 – Diagrama conceptual

4.5. Casos d'ús

4.5.1. Subsistema d'inici i configuració

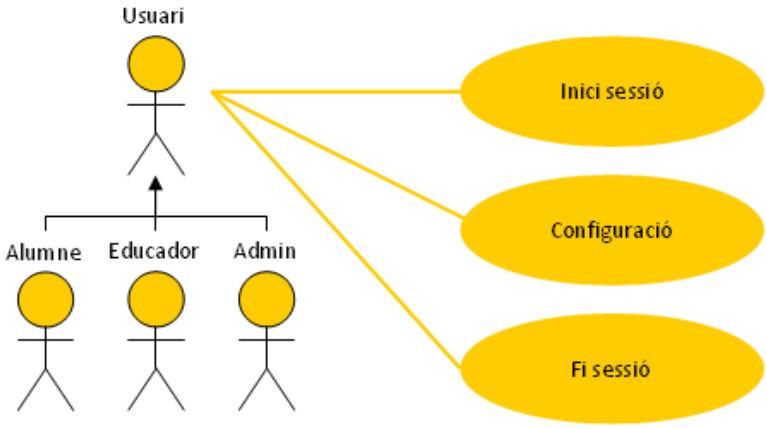


Figura 8 – Cas d'us del subsistema d'inici i configuració

Actor: Usuari

Precondició: Cap

Postcondició: Si s'ha iniciat sessió l'aplicació disposa d'un identificador de sessió vàlid

Descripció: Mostra la pantalla inicial on l'Usuari pot iniciar una sessió i estableixi una connexió amb el servidor Moodle, o bé pot modificar la configuració de l'aplicació

4.5.2. Subsistema d'usuari

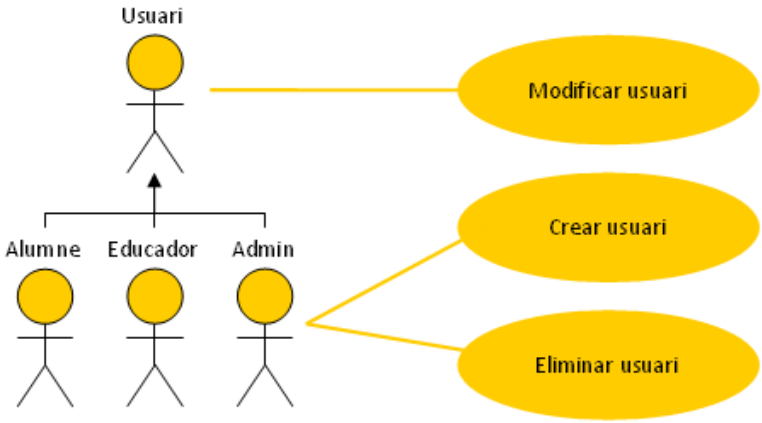


Figura 9 – Cas d'us del subsistema d'usuari

Actor: Alumne, Educador, Administrador

Precondició: Sessió iniciada

Postcondició: S'ha creat, modificat o eliminat un usuari

Descripció: Permet que un Alumne, un Educador o un Administrador modifiqui les seves dades o que un administrador creï un usuari nou o n'elimini un d'existent

4.5.3. Subsistema de cursos

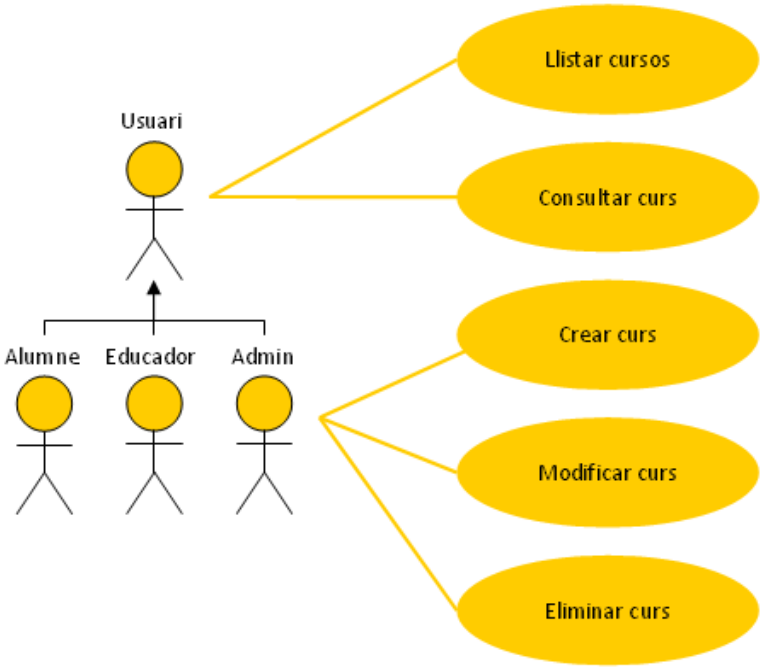


Figura 10 – Cas d'us del subsistema de cursos

Actor: Alumne, Educador, Administrador

Precondició: Sessió iniciada

Postcondició: S'ha creat un nou curs, s'ha modificat, s'ha eliminat o s'ha obtingut la llista de cursos disponible

Descripció: Permet que un Alumne, un Educador o un Administrador llisti la relació cursos disponibles o que un Educador o un Administrador creï un nou curs, l'elimini o el modifiqui

4.5.4. Subsistema de matriculació

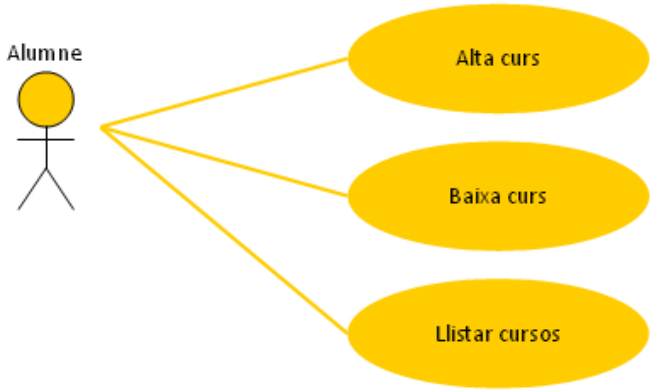


Figura 11 – Cas d'us del subsistema de matriculació

Actor: Alumne

Precondició: Sessió iniciada

Postcondició: L'alumne s'ha donat d'alta o de baixa d'un curs o s'ha obtingut la llista de cursos matriculats

Descripció: Permet que un Alumne es doni d'alta o de baixa d'un curs o obtingui la llista de cursos matriculats

4.6. Prototip

En aquest apartat es presenten els prototips d'algunes de les pantalles de la interfície d'usuari que es van dissenyar inicialment. L'objectiu d'aquests prototips no és un altre que donar al client una aproximació de l'aspecte que tindrà l'aplicació per tal de fer els ajustos necessaris.

Cal remarcar que algunes d'aquestes pantalles van ser modificades durant el desenvolupament amb la qual cosa l'aplicació definitiva presenta algunes diferències envers aquests prototips.

4.6.1. Pantalla Login

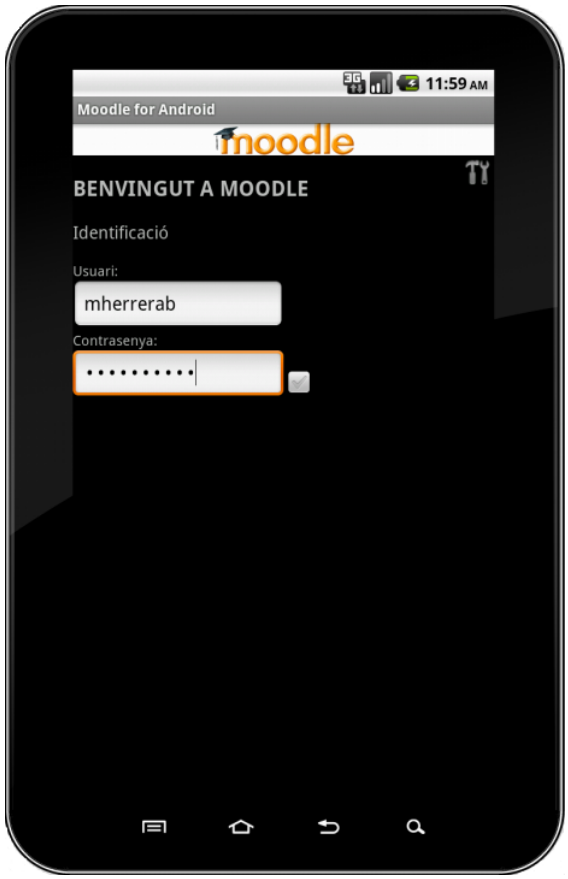


Figura 12 – Prototip pantalla Login

Funcionalitats:

- Autentificació servidor Moodle per usuari i contrasenya o per *token*. A l'exemple es mostra el mode usuari i contrasenya.
- Accés a la pantalla de preferències de l'aplicació

4.6.2. Pantalla Preferències

Figura 13 – Prototip pantalla Preferències

Funcionalitats:

- Establir els paràmetres de configuració de l'aplicació. A la miniatura de la dreta es mostra la mateixa pantalla amb el selector d'idiomes desplegat.

4.6.3. Pantalla Menú

Figura 14 – Prototip pantalla Menú

Funcionalitats:

- Accés a les funcionalitats de l'aplicació: usuaris, cursos i matriculació
- Accés a la pantalla de preferències de l'aplicació

4.6.4. Pantalla Usuari

Figura 15 – Prototip pantalla Modificació Usuari

Funcionalitats:

- Creació i eliminació d'usuaris
- Modificació de les dades de detall bàsiques d'un usuari

4.6.5. Pantalla Llista de cursos

Figura 16 – Prototip pantalla Llista de cursos

Funcionalitats:

- Mostrar la relació de cursos disponibles
- Permet accedir a la consulta de la fitxa de detall d'un curs

5. Disseny

5.1. Arquitectura

5.1.1. Arquitectura Android

Android està basat en un nucli Linux sobre el qual s'executa un *runtime* que es composa de la màquina virtual Dalvik i un conjunt de biblioteques base. De forma paral·lela a aquest *runtime* hi ha una conjunt de biblioteques escrites en C/C++ que proporcionen la major part de funcionalitats, com per exemple, l'accés a base de dades SQLite, gràfics, 3D o accés a medis. Per sobre hi ha el marc d'aplicacions al qual s'hi pot accedir a través de l'Android API per tal de desenvolupar noves aplicacions com la que ens ocupa en aquest projecte. Aquestes, juntament amb les aplicacions natives, es situen a la capa superior que és amb la que interactua l'usuari.

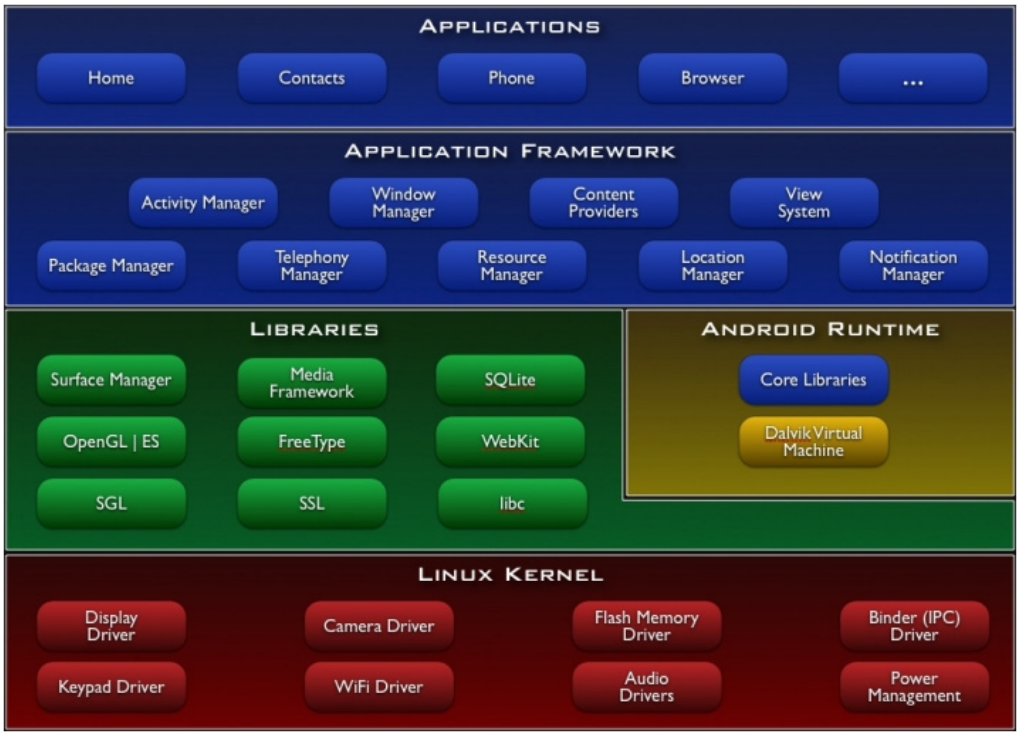


Figura 17 – Arquitectura Android
Font: http://elinux.org/images/c/c2/Android-system-architecture.jpg

5.1.2. Arquitectura dels Moodle *Web Services*

L'estudi de l'arquitectura del nucli de Moodle no forma part de l'objectiu d'aquest projecte atès que, precisament, el fet de fer servir la seva API per construir la nostra aplicació que "consumeix" els seus *Web Services* ens permet abstraure'ns del funcionament intern de Moodle.

Consegüentment, ens centrarem només en l'arquitectura del seus *Web Services*, la qual es basa en tres capes. La superior, *Services Layer*, és la que conté els serveis que ofereix Moodle en els seus *Web Services*; la intermèdia, *Integration Layer*, conté l'API pròpiament dita, és a dir l'especificació i signatura dels serveis oferts, i la inferior i última, *Connect Layer*, representa els diferents protocols que es poden fer servir per establir una connexió amb els *Web Services*.

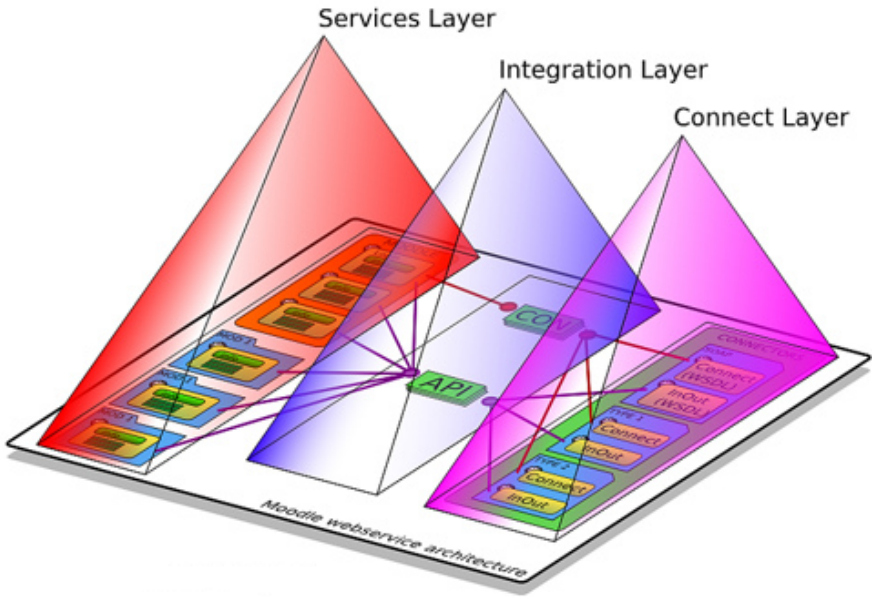


Figura 18 – Arquitectura Moodle *Web Services*
Font: http://farm3.static.flickr.com/2164/2403150142_05d80d3f1c.jpg

5.1.3. Arquitectura lògica de l'aplicació

Per tal d'obtenir un baix nivell d'acoblament i una alta cohesió entre els diferents elements de l'aplicació a desenvolupar, aquesta s'estructurarà en una arquitectura de tres capes. Són les següents:

- Capa de presentació. Es responsabilitzarà de la interacció amb l'usuari, per tant, contindrà només les *Activities*[20] i, si s'escau, algun altre mètode complementari.

[20] Una *Activity* és un component Android que proporciona interfície visual

- Capa de negoci. Serà el nucli de l'aplicació i es responsabilitzarà del funcionament principal de l'aplicació –bàsicament serà l'enllaç entre la interfície d'usuari i la capa d'accés a dades i *Web Services*–, per tant, contindrà *Services*[21] i, si s'escau, algun altre mètode complementari.

- Capa d'accés a dades i *Web Services*. Serà la responsable de gestionar les transaccions de base de dades i fer les crides als mètodes que "consumiran" els *Web Services* de Moodle. Degut a la naturalesa de l'aplicació els accessos a base de dades seran mínims –en concret es faran servir només per guardar/recuperar les preferències de l'aplicació– per la qual cosa la funció principal d'aquesta capa serà la de fer crides i rebre respostes de l'API de Moodle. Els seus components principals seran *Services* i, si s'escau, altres mètodes complementaris.

La següent figura il·lustra l'arquitectura bàsica de l'aplicació.

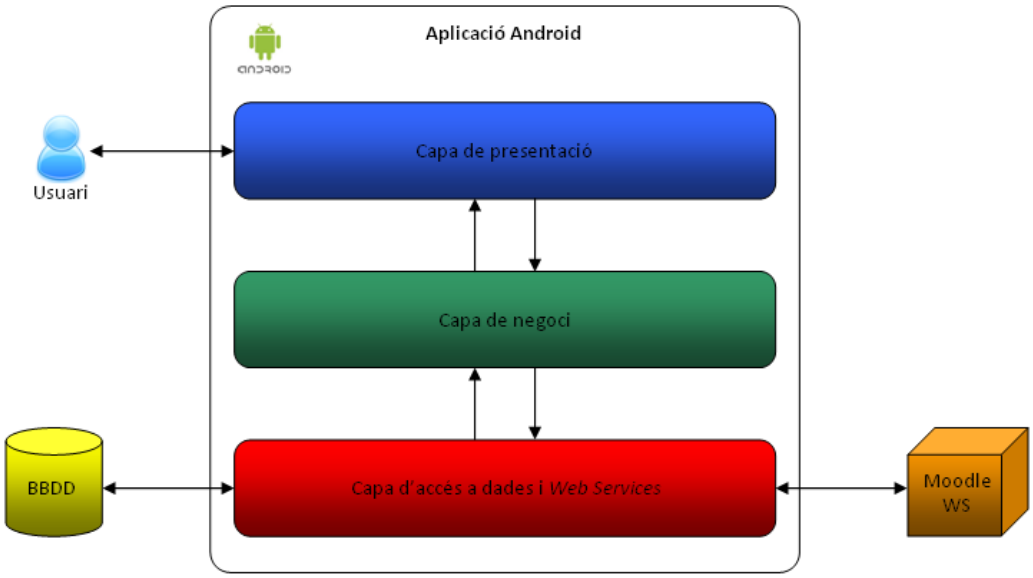


Figura 19 – Arquitectura de l'aplicació

[21] Un *Service* és un component Android que no té interfície gràfica i generalment s'usa per accions de llarga durada

Una vegada presentada l'arquitectura d'Android, la dels Moodle *Web Services* i la de l'aplicació, és fàcil superposar aquesta última sobre les altres per tal d'obtenir l'arquitectura global del projecte.

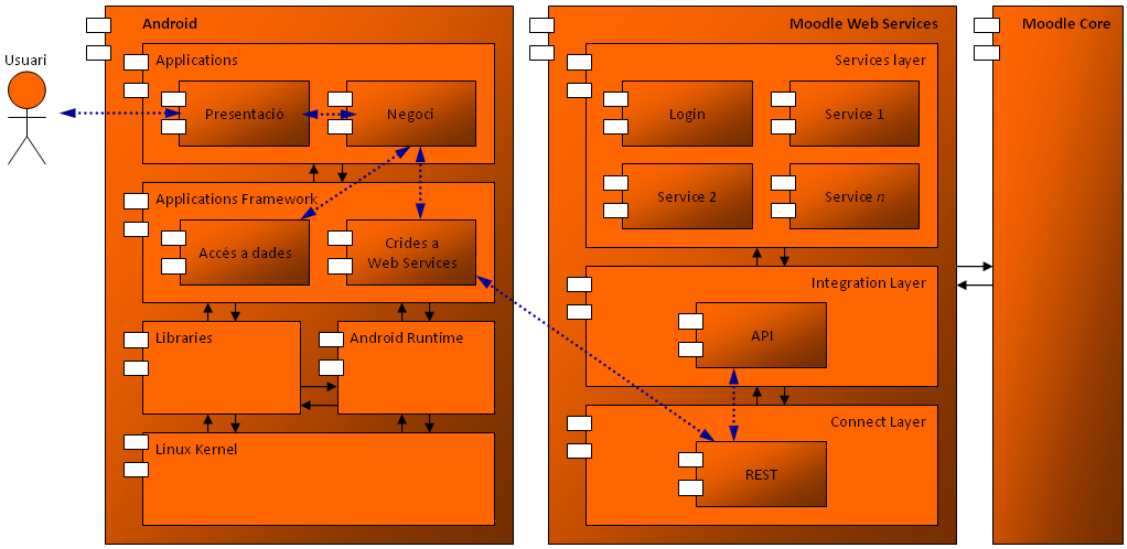


Figura 20 – Arquitectura global

Les línies de punts de la figura anterior indiquen els fluxos entre els diferents components que intervenen en l'aplicació, tenint en compte que el protocol que es farà servir entre Android i Moodle per fer crides als *Web Services* d'aquest és REST.

5.1.4. Arquitectura de xarxa

Segons descrit a l'apartat 3.2, l'arquitectura de xarxa es basa en la comunicació entre l'aplicació i el servidor Moodle mitjançant xarxes sense fils amb diferents possibilitats:

- Directament a través d'Internet mitjançant un operador de telefonia mòbil 3G o per altres mitjans com per exemple WiMAX (802.16) si el dispositiu ho permet.
- Fent ús d'una xarxa local sense fils WiFi (802.11) per connectar-se a Internet.
- Directament via WiFi (802.11) a la mateixa xarxa local on es troba el servidor.

La figura 3 mostrada anteriorment il·lustra aquestes diferents alternatives de connexió.

5.2. Model entitat-relació

Tal com s'ha explicat a l'apartat 5.1.3. els accessos a dades es faran servir només per la gestió de preferències atès que el funcionament principal de l'aplicació es basa en crides a *Web Services*, per tant, no hi haurà una base de dades ni una gestió de persistència que requereixi un anàlisi i disseny profund o l'ús de patrons.

L'aplicació únicament disposarà de les entitats *User*, *Course*, *Category*, *Login*, *Prefs* i *Session* sense persistència, amb la qual cosa el model entitat-relació és molt simple tal com es veu a la següent figura.

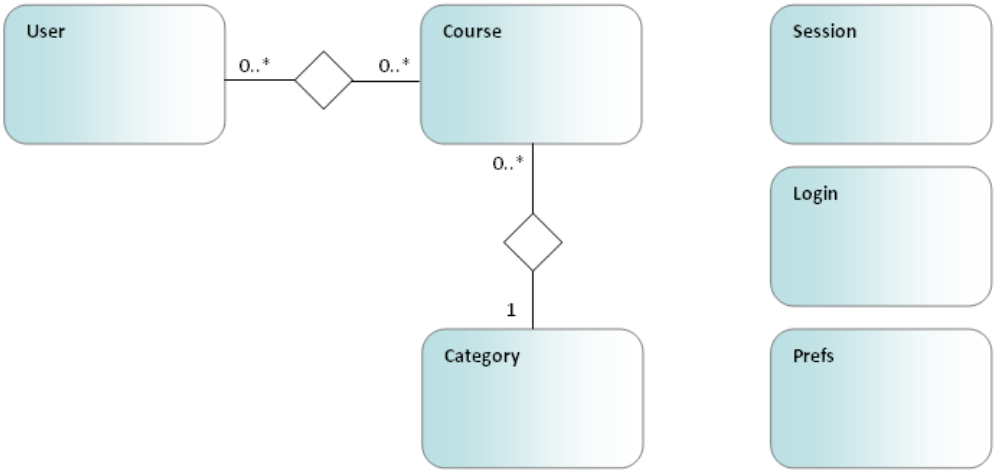


Figura 21 – Model Entitat-Relació

5.2.1. Entitat *User*

Permet instanciar un usuari:

Entitat	User	
Descripció	Modela una instància d'usuari	
Atributs	userName	Codi
	firstName	Nom
	lastName	Cognom
	eMail	Correu electrònic
	idNumber	Identificador únic
	password	Contrasenya
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions		

Taula 14 – Entitat *User*

5.2.2. Entitat *Course*

Permet instanciar un curs:

Entitat	Course	
Descripció	Modela una instància de curs	
Atributs	id	Codi
	categoryId	Identificador únic de la categoria
	Category	Nom de la categoria
	shortName	Nom curt
	fullName	Nom complert
	summary	Resum
	idNumber	Identificador únic
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions	Implementa Parcelable per poder ser enviat com a paràmetre en crides entre Activities	

Taula 15 – Entitat *Course*

5.2.3. Entitat *Category*

Permet instanciar una categoria de curs:

Entitat	Category	
Descripció	Modela una instància de categoria de cursos	
Atributs	id	Identificador únic
	name	Nom
	description	Descripció
	parent	Identificador de categoria pare
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions		

Taula 16 – Entitat *Category*

5.2.4. Entitat *Session*

Permet instanciar les dades de la sessió iniciada:

Entitat	Session	
Descripció	Modela una instància de sessió	
Atributs	url	url de connexió al servidor
	Key	Clau de connexió
	Client	Client de connexió
	userId	Identificador únic d'usuari
	userName	Codi d'usuari
	userDesc	Nom i cognom d'usuari
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions	Implementa Parcelable per poder ser enviat com a paràmetre en crides entre Activities	

Taula 17 – Entitat *Session*

5.2.5. Entitat *Login*

Permet instanciar les dades d'un inici de sessió:

Entitat	Login	
Descripció	Modela una instància de dades d'inici de sessió	
Atributs	user	Codi d'usuari
	Password	Contrasenya
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions		

Taula 18 – Entitat *Login*

5.2.6. Entitat *Prefs*

Permet instanciar les preferències de l'aplicació:

Entitat	Prefs	
Descripció	Modela una instància de dades de preferències de l'aplicació	
Atributs	server	Adreça servidor Moodle
Mètodes	getters de tots els atrbuts	
	setters de tots els atrbuts	
Observacions		

Taula 19 – Entitat *Prefs*

5.3. Diagrama de classes

La següent representació de classes il·lustra a alt nivell la interacció entre les diferents classes de l'aplicació. Es poden observar classes del tipus *Activity*, que són les que formen la interfície d'usuari; les gestores o *Managers* que formen la capa de negoci; la resta d'accés a dades i a Web Services, i finalment les entitats, utilitzades per representar els objectes que instancien aquestes.

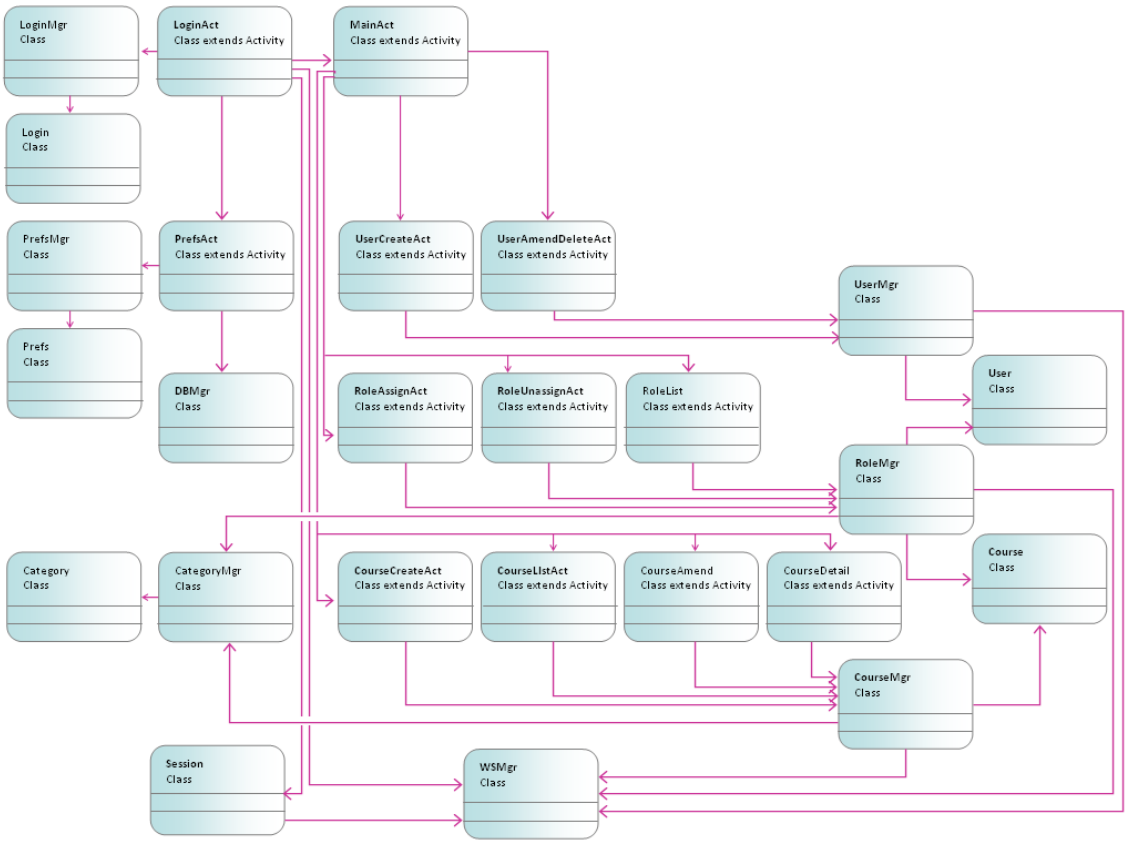


Figura 22 – Diagrama de classes

5.3.1. Classes de la capa de presentació

Les classes de la capa de presentació són les responsables de la interacció amb l'usuari i de presentar les pantalles de la interfície. Són les següents:

Classe	Funcionalitat
CourseAmendAct	Mostra i gestiona la pantalla de modificació i eliminació de cursos
CourseCreateAct	Mostra i gestiona la pantalla de creació de cursos
CourseDetailAct	Mostra i gestiona la pantalla de detall de cursos
CourseListAct	Mostra i gestiona la pantalla de llista de cursos
LoginAct	Mostra i gestiona la pantalla d'inici/finalització de sessió
MainAct	Mostra i gestiona el menú principal
PrefsAct	Mostra i gestiona la pantalla de preferències
RoleCreateAct	Mostra i gestiona la pantalla de creació de matriculacions
RoleDeleteAct	Mostra i gestiona la pantalla d'eliminació de matriculacions
RoleListAct	Mostra i gestiona la pantalla de llista de matriculacions
UserAmendAct	Mostra i gestiona la pantalla de modificació i eliminació d'usuaris
UserCreateAct	Mostra i gestiona la pantalla de creació d'usuaris

Taula 20 – Classes de la capa de presentació

5.3.2. Classes de la capa de negoci

Les classes de la capa de negoci, són les responsables d'executar tota la lògica de l'aplicació i actuen d'enllaç entre la interfície (capa de presentació) i les dades i Web Services (capa d'accés a dades i Web Services) aconseguint així un elevat grau d'abstracció entre l'usuari i el servidor. Són les següents:

Classe	Funcionalitat
CategoryMgr	Gestió de categories de cursos, llegint les categories del servidor i creant l'arbre de categories.
CourseMgr	Gestió de cursos (creació, modificació, eliminació, consulta i llistat). Construcció de resum i validació de dades.
LoginMgr	Gestió d'inici i finalització de sessió
PrefsMgr	Gestió de lectura i escriptura de les preferències (adreça de servidor) a l'àres de preferències de l'aplicació Android. Validació de dades.
RoleMgr	Gestió de matriculacions (creació, eliminació i consulta). Validació de dades.
UserMgr	Gestió d'usuaris (creació, eliminació i consulta). Validació de dades.

Taula 21 – Classes de la capa de negoci

5.3.3. Classes de la capa d'accés a dades i *Web Services*

Les classes de la capa d'accés a dades i *Web Services* són les encarregades d'emmagatzemar les dades corresponents a les preferències de l'aplicació (recordem que no hi ha base de dades ni persistència de cap mena) i de fer les crides als mètodes de l'API a través de *Web Services*:

Classe	Funcionalitat
DBMgr	Lectura i escriptura de les preferències de l'aplicació Android
WSMgr	Crides als mètodes de l'API de Moodle

Taula 22 – Classes de la capa d'accés a dades i *Web Services*

5.3.4. Classes entitat

A l'apartat 5.2 es pot trobar una descripció detallada de les classes entitat.

5.4. Diagrames de col·laboració i seqüència

Com no és possible mostrar una col·lecció de diagrames de col·laboració i seqüència que representi totes les combinacions possibles que ofereix l'aplicació, es mostren dos exemples d'acord al disseny inicial el quals que cobreixen una bona part de les possibilitats.

5.4.1. Exemple 1 – Creació d'un usuari

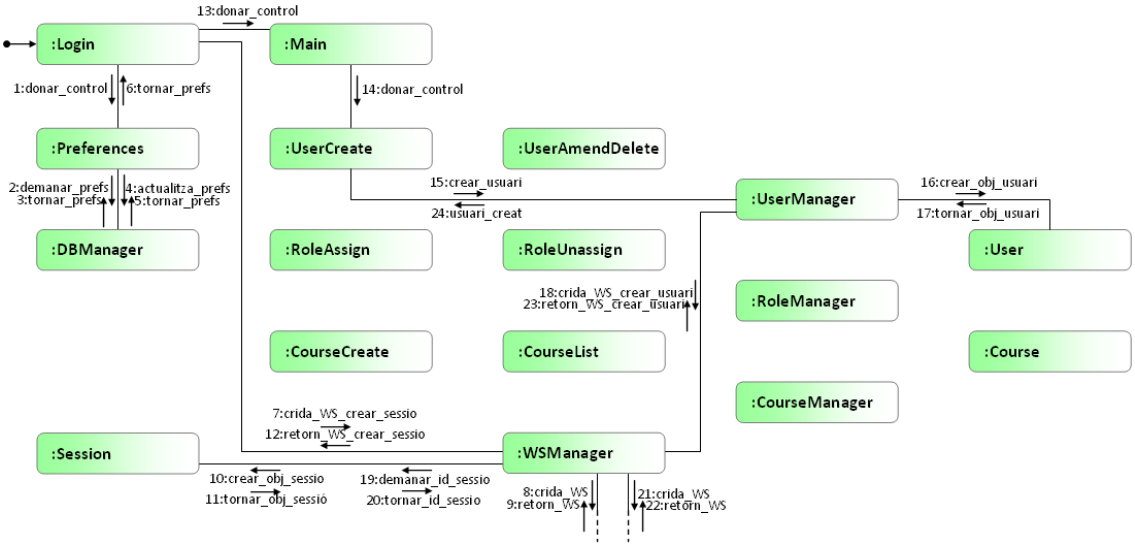


Figura 23 – Diagrama de col·laboració i seqüència (exemple 1)

En la figura anterior es poden apreciar les següent accions:

- 1, 2, 3, 4, 5, 6 : L'usuari inicia l'aplicació, accedeix a la pantalla de configuració de preferències, les modifica i es retornen a la pantalla d'inici de sessió les preferències que s'ha de fer servir a partir d'ara
- 7, 8, 9 : L'usuari introdueix l'usuari i la contrasenya i sol·licita l'inici de sessió, l'aplicació demana als *Web Services* que facin una crida a Moodle per iniciar sessió i aquest torna un identificador de sessió vàlid
- 10, 11 : Amb l'identificador de sessió obtingut es crea una instància de l'objecte sessió
- 12, 13 : Es retorna a la pantalla d'inici de sessió que aquest ha estat satisfactori i es passa el control al menú principal
- 14, 15 : L'usuari selecciona l'opció de crear un nou usuari, omple els camps de detall, confirma la creació de l'usuari i l'aplicació passa el control al gestor d'usuaris
- 16, 17 : L'aplicació crea una instància de l'objecte usuari amb les dades entrades i la torna al gestor d'usuaris
- 18 : El gestor d'usuaris demana als *Web Services* que facin la crida a Moodle per crear l'usuari
- 19, 20 : Els *Web Services* obtenen l'identificador de sessió creat prèviament (necessari per realitzar qualsevol crida)
- 21, 22 : Els *Web Services* fan la crida a Moodle per que creï un compte d'usuari amb les dades de l'objecte usuari que s'ha creat en els passos 16 i 17

- 23 : Els *Web Services* tornen al gestor d'usuaris la resposta de que el nou compte d'usuari s'ha creat satisfactòriament
- 24 : El gestor d'usuaris torna la resposta a la interfície per que informi a l'usuari que el nou compte s'ha creat correctament

5.4.2. Exemple 2 - Llista de cursos

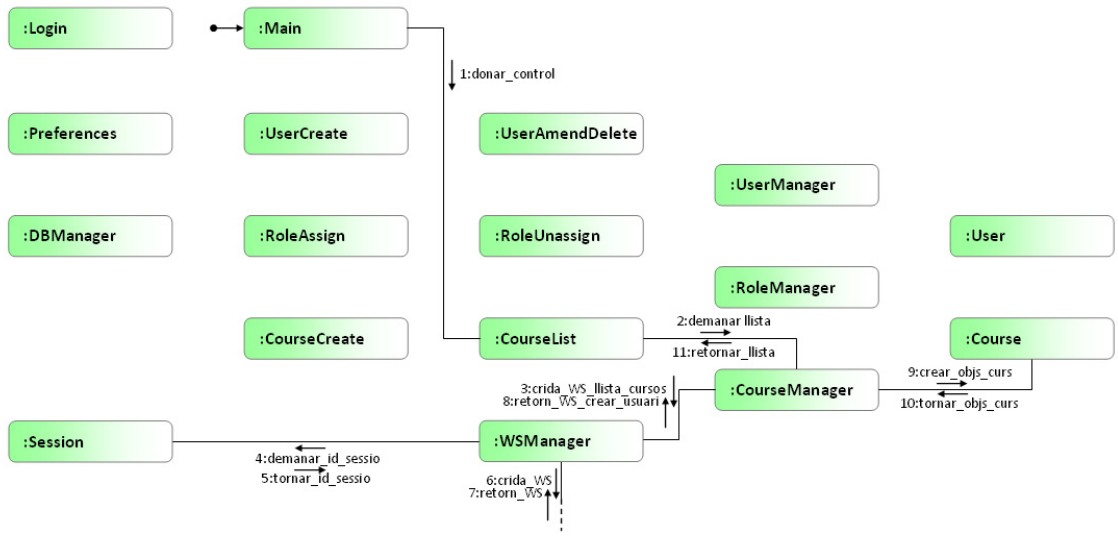


Figura 24 – Diagrama de col·laboració i seqüència (exemple 2)

En la figura anterior es poden apreciar les següent accions:

- Partim del punt en que l'usuari ja ha iniciat sessió i està a la pantalla del menú principal
- 1 : L'usuari selecciona l'opció d'obtenir la llista de cursos, la confirma i passa el control al gestor de cursos
- 2 : El gestor de cursos demana als *Web Services* que facin la crida a Moodle per obtenir la llista de cursos
- 4, 5 : Els *Web Services* obtenen l'identificador de sessió creat prèviament (necessari per realitzar qualsevol crida)
- 6, 7 : Els *Web Services* fan la crida a Moodle per demanar la llista de cursos
- 8 : Els *Web Services* tornen al gestor de cursos la llista de cursos
- 9, 10 : El gestor de cursos crea una instància de l'objecte curs per cada curs de la llista
- 11 : El gestor de cursos retorna la llista de cursos a la interfície per que la presenti a l'usuari

5.5. Altres consideracions de disseny

Per tal de potenciar la usabilitat de l'aplicació i que sigui altament intuïtiva, aquesta es mantindrà sempre en la màxima senzillesa possible i s'aplicaran alguns dels següents principis de disseny:

- No s'inclourà cap element innecessari, com per exemple botons que permetin fer les mateixes funcions que ja incorporen per defecte tots els dispositius Android (tecles de navegació "inici" o "retorn").

- Es faran servir els elements de disseny estàndard Android.

- S'evitarà fer servir elements que, malgrat ser estàndard, són estranys per la majoria d'usuaris Android, com per exemple l'activació de menús contextuals (s'hi accedeix fent una pulsació llarga).

- Pels botons i altres elements gràfics es faran servir les metàfores habituals.

Addicionalment, l'aplicació estarà preparada per suportar diferents idiomes i regions d'acord als processos de disseny de programari *L10n* i *i18n*. Cal remarcar que l'idioma del contingut (texts dels cursos, errors de sistema, etc...) vindrà determinat pel servidor Moodle.

6. Implementació

Aquest capítol descriu el procés d'implementació de l'aplicació d'acord al disseny realitzat, indicant en tots els casos les peculiaritats del sistema i les decisions preses durant el desenvolupament.

Abans d'entrar en el detall de la implementació, es fa una breu descripció dels principals components de les aplicacions i de les característiques de desenvolupament Android.

6.1. Característiques de desenvolupament Android

6.1.1. Components

Les aplicacions Android es basen en components. Aquests poden ser de 4 tipus:

- *Activity*. Són interfícies visuals per interactuar amb l'usuari. Poden executar el seu propi codi i són les responsables de presentar els objectes i recursos que composen les pantalles. Una aplicació pot tenir moltes *Activities* que es cridin entre si.

- *Service*. Són processos que s'executen en segon pla per fer accions de llarga durada. No disposen d'interfície gràfica.

- *Content providers*. Permet que l'aplicació gestioni dades per a ella mateixa o posant-les a disposició d'altres aplicacions, per exemple fent servir la base de dades SQLite.

- *BroadcastReceivers*. Són components que simplement "escolten" a que es produeixin determinats esdeveniments, per exemple bateria baixa, per iniciar alguna acció.

Per tal de poder comunicar tots aquests elements entre si es fan servir els *intents*, els quals també permeten establir enllaços entre diferents aplicacions. Addicionalment als components específics Android, qualsevol aplicació pot fer servir els components Java habituals com per exemple classes Java, llibreries, etc.

6.1.2. Interfície gràfica

Les interfícies gràfiques en Android es defineixen en arxius XML els quals descriuen tots els objectes que composen cada interfície amb els seus atributs. Aquests objectes són o hereten de la classe *View* i s'organitzen en *ViewGroups*, els quals proporcionen diferents *Layouts* o contenidors per tal d'organitzar els elements d'una forma que faciliti l'adaptació del disseny de la interfície a les diferents mides, resolucions i orientacions de pantalla.

Alguns exemples d'objectes *View* poden ser: botons, texts, menús, gràfics. De *ViewGroups* n'hi ha diferents tipus: *FrameLayout*, *LinerLayout*, *RelativeLayout* o *TableLayout*, cadascun amb les seves característiques d'orientació i organització dels elements que contenen.

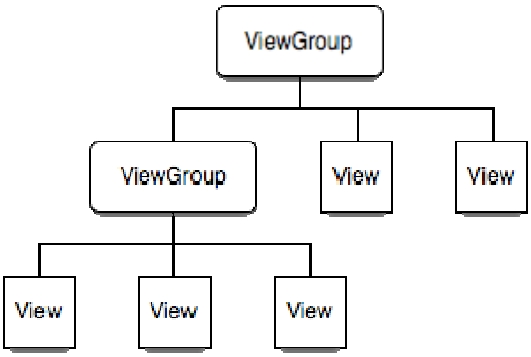


Figura 25 – Herència d'objectes *View*
Font: http://developer.android.com/guide/topics/ui/index.html

6.1.3. El cicle de vida

Una aplicació Android amb interfície d'usuari es basa en una *Activity* –única classe que permet instanciar una interfície– conseqüentment, qualsevol aplicació interactiva s'inicia amb una *Activity* que serà la classe que es cridarà en iniciar-se l'aplicació.

A l'hora de desenvolupar una aplicació Android és essencial conèixer el cicle de vida d'una Activity i quins esdeveniment s'executen quan aquesta es crida, passa a segon pla, es reactiva o es destrueix, per tal de poder incloure el codi necessari en els mètodes adequats. Per exemple, el codi d'inicialització del contingut de la interfície s'haurà de posar al mètode onCreate() i el codi per finalitzar sessió al mètode onDestroy().

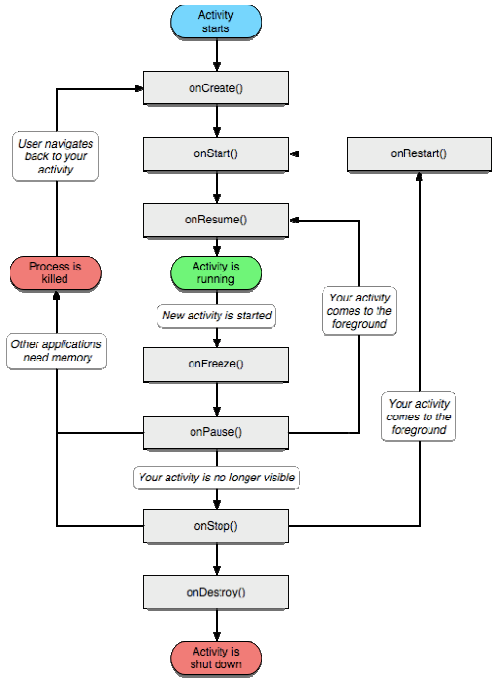


Figura 26 – Cicle de vida Android
Font: http://developer.android.com/reference/android/app/Activity.html

6.1.4. L'arxiu *manifest*

El manifest de l'Android és un fitxer XML que defineix informació sobre l'aplicació. Aquest fitxer es troba en l'arrel del projecte Android i és imprescindible perquè l'aplicació sigui executada pel sistema operatiu.

Serveix per a declarar una sèrie d'informació sobre l'aplicació que el sistema ha de conèixer per endavant, per exemple:

- Permisos necessaris per a poder funcionar, com l'accés a la càmera.
- El nivell de l'API de l'Android necessari per a treballar.
- Les característiques de maquinari i programari necessàries per a treballar, com per exemple tenir o no càmera.
- Els components que formen la nostra aplicació, *Activities, Services*, *Intents*, etc...
- Els permisos específics que requereix l'aplicació, com per exemple accés a Internet.

6.2. Capa de presentació

Tal com s'ha detallat a l'apartat 5.3.1. a la capa de presentació hi ha 12 classes. Totes les classes segueixen la mateixa estructura i tenen les següents característiques i responsabilitats:

- Tenen associada una interfície gràfica.
- Executen codi per interactuar amb la interfície i fer crides a altres classes.
- Es criden entre si per dotar a l'aplicació de la navegació explicada a l'apartat 4.4.
- Es passen, si s'escau, paràmetres mitjançant *parcels*.
- Estan declarades a l'arxiu *manifest*.

6.2.1. Interfície gràfica

Cada interfície gràfica està definida amb un arxiu XML que descriu tots els objectes que la composen amb els seus atributs. Cadascun d'aquests objectes té un identificador únic per possibilitar que l'*Activity* pugui fer-hi referència des del codi Java. Al seu torn, aquests objectes poden fer referència a altres recursos de l'aplicació, com texts o gràfics, que també estan identificats amb el seu identificador únic.

El següent exemple mostra, en primer lloc, la definició d'una etiqueta (*TextView*) i en segon lloc, la d'un camp d'edició tipus text (*EditText*):

```
<TextView
        android:id="@+id/label_pwd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/user"
        android:textSize="20px"
        android:paddingTop="20px"
/>

<EditText
        android:id="@+id/edit_usr"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:inputType="text"
        android:minWidth="300px"
/>
```

Figura 27 – Exemple d'objectes d'una interfície amb XML

Es poden apreciar les característiques d'aquests objectes només veient els seus atributs (*android:layout_height* o *android:textSize*), els seus identificadors (*android:id*) i les referències que tenen a altres objectes (*android:text*).

Finalment, remarcar que un dels objectius del disseny de les interfícies ha estat que aquestes siguin molt senzilles i puguin ser utilitzades amb un entrenament mínim o nul. Per aquest motiu s'han fet servir abastament elements que facilitin el seu ús, no obstant això, hi ha a disposició dels usuaris el "Manual de funcionament"[22] amb una explicació del funcionament de l'aplicació.

6.2.2. L'execució de codi

Les Activities d'aquesta aplicació tenen tres missions:

- Executar el codi per interactuar amb la interfície
- Executar el codi per cridar a altres *Activities* o a gestors de la capa de negoci

El següent exemple mostra la inicialització de les dades de la interfície i la recepció d'un paràmetre passat en un *parcel*.

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        //Inicialitzacions
        label_usrDesc = (TextView) findViewById(R.id.label_usrname);
        button_users = (ImageButton) findViewById(R.id.button_users);
        button_courses = (ImageButton) findViewById(R.id.button_courses);
```

[22] Veure el capítol "Manual de funcionalment" del document "Manuals"

```
        button_enrol = (ImageButton) findViewById(R.id.button_enrol);
        button_users.setOnClickListener(this);
        button_courses.setOnClickListener(this);
        button_enrol.setOnClickListener(this);

        Bundle b = getIntent().getExtras();
        session = (Session) b.getParcelable("session");

        label_usrDesc.setText(session.getUserDesc());
}
```

Figura 28 – Exemple de codi d'una *Activity* de la capa de presentació

A l'exemple anterior es poden veure moltes de les accions que executa el codi de la interfície. Per exemple: fa servir *SetContentView()* per establir la interfície XML amb la que interactua; identifica els elements de la interfície amb el seu identificador únic definit a l'arxiu XML amb *findViewById()*; rep un paràmetre de la classe *Session* amb *getParcelable()*, i inicialitza el valor d'un camp de la interfície amb *setText()*.

El següent exemple mostra la crida a una altra Activity a la que se li passa un paràmetre de la classe *Session* en un *parcel*:

```
myIntent = new Intent(this , UserCreateAct.class);

b.putParcelable("session", session);
myIntent.putExtras(b);

startActivity(myIntent);
```

Figura 29 – Exemple de codi d'una crida entre *Activities*

Es pot observar com les crides entre components Android s'han de fer amb un *intent*.

6.2.3. Els *parcels*

Les crides entre components Android només permeten passar-se paràmetres de tipus natius, com *int*, *string*, *double*, etc. Això representa un inconvenient quan es volen passar classes pròpies.

Per solucionar aquest inconvenient Android proporciona el mecanisme *parcel*, que no és res més que uns mètodes incorporats a les classes que volem "parcelar" que descomposen l'objecte en tipus natius, els empaqueten en un *parcel*, els envien, i a la recepció els desempaqueten per tornar reconstruir l'objecte original.

El següent objecte il·lustra el codi afegir a la classe Session per permetre el seu enviament com a *parcel*:

```
@Override
public int describeContents() {
        return 0;
}

@Override
public void writeToParcel(Parcel out, int flags) {
        out.writeString(getUrl());
```

```
        out.writeString(getKey());
        out.writeInt(getClient());
        out.writeInt(getUserId());
        out.writeString(getUserName());
        out.writeString(getUserDesc());
}

public static final Parcelable.Creator<Session> CREATOR
                = new Parcelable.Creator<Session>() {
        public Session createFromParcel(Parcel in) {
                return new Session(in);
        }
        public Session[] newArray(int size) {
                return new Session[size];
        }
};

private Session(Parcel in) {
        setUrl(in.readString());
        setKey(in.readString());
        setClient(in.readInt());
        setUserId(in.readInt());
        setUserName(in.readString());
        setUserDesc(in.readString());
}
```

Figura 30 – Exemple de *parcel*

6.2.4. Declaració a l'arxiu *manifest*

A l'arxiu *manifest* s'han de declarar tots els components de l'aplicació, els seus permisos especials i altres aspectes estètics com el nom de l'*app* i la icona que es mostrarà.

El següent exemple mostra, en primer lloc, la declaració de l'*Activity* principal d'aquesta aplicació, *LoginAct*, que correspon a la pantalla d'inici de sessió i és la que es crida automàticament en iniciar-se l'aplicació; en segon lloc, la definició del nom i de la icona que mostrarà l'aplicació (es pot observar com estan referenciats a recursos amb el seu identificador únic), i en tercer lloc, el permís que es dóna a l'aplicació per que pugui accedir a Internet.

```
<activity
     android:label="@string/app_name"
     android:name=".LoginAct" >
     <intent-filter >
         <action android:name="android.intent.action.MAIN" />
         <category android:name="android.intent.category.LAUNCHER" />
     </intent-filter>
</activity>


<application
        android:icon="@drawable/moodle_icon"
        android:label="@string/app_name"
>


<uses-permission android:name="android.permission.INTERNET" />
```

Figura 31 – Exemple de *manifest*

6.2.5. Idiomes

Tal com s'ha descrit a l'apartat 5.5., la interfície està preparada per suportar diferents idiomes d'acord als processos de disseny de programari *L10n* i *i18n*.

A Android això s'aconsegueix creant un arxiu *strings* en format XML i ubicant-lo dins del projecte Java sota la carpeta *res/values-cc*, on *cc* correspon al codi ISO de l'idioma, és a dir *ca* pel català, *es* per l'espanyol (castellà), *en* per l'anglès, etc...

Durant l'execució de l'aplicació, el sistema operatiu Android s'encarregarà de seleccionar automàticament l'arxiu de l'idioma definit en el dispositiu.

Els següents fragments dels arxius en català i en anglès mostren la seva estructura on es pot veure l'identificador i el text de cada literal.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Moodle per Android</string>
    <string name="welcome">BENVINGUT A MOODLE</string>
    <string name="identification">Identificació</string>
    <string name="user">Usuari:</string>
    <string name="password">Contrasenya:</string>
    <string name="users">Usuaris</string>
    <string name="courses">Cursos</string>
    <string name="enrol">Matriculació</string>
    <string name="server">Servidor:</string>
    <string name="name">Nom:</string>

</resources>


<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Moodle for Android</string>
    <string name="welcome">WELCOME TO MOODLE</string>
    <string name="identification">Identification</string>
    <string name="user">User:</string>
    <string name="password">Password:</string>
    <string name="users">Users</string>
    <string name="courses">Courses</string>
    <string name="enrol">Enrolments</string>
    <string name="server">Server:</string>
    <string name="name">Name:</string>

</resources>
```

Figura 32 – Exemple de *strings*

Cal recordar que l'idioma del contingut (texts dels cursos, errors de sistema, etc...) vindrà determinat pel servidor Moodle.

6.3. Capa de negoci

Tal com s'ha detallat a l'apartat 5.3.2. a la capa de negoci hi ha 6 classes gestores, una per cada tipus de classe que cal gestionar. La responsabilitat d'aquestes és:

- Rebre les dades de la capa de presentació. Per exemple, en el cas de la creació d'un usuari, el gestor d'usuaris *UserMgr* rebrà un objecte *User* de *l'Activity UserCreateAct* de la capa de presentació.

- Validar que les dades siguin correctes. Seguint amb el mateix exemple, comprovarà que els atributs de *User* compleixin amb les regles de validació i si es produeix algun error retornarà a la capa de presentació el text de l'avís que ha de presentar a l'usuari.

- En cas que no es produeixi cap error, instanciarà el gestor de *WSMgr* de la capa d'accés a dades i *Web Services* passant-li l'objecte *User* per tal que faci la corresponent crida a l'API de Moodle amb l'objectiu de crear l'usuari.

- Si *WSMgr* retorna algun avís, per exemple perquè l'API retorna que l'usuari ja existeix, *UserMgr* el passarà a la capa de presentació per que sigui mostrat a l'usuari. Altrament enviarà confirmació a la capa de presentació de que el nou usuari ha estat creat satisfactòriament per tal que mostri l'avís a l'usuari.

- Addicionalment, en molts casos els gestors de la capa de negoci tenen altres responsabilitats, com per exemple generar identificadors o preparar les dades que han d'enviar a les capes de presentació o d'accés a dades i *Web Services*.

El següent fragment de codi mostra, en primer lloc, la validació de les dades d'un usuari i en segon lloc, la generació del seu identificador únic basat en l'instant de creació:

```
/*
 * Valida dades usuari
 */
private String validateUserData(boolean pwdCheck){
        Resources r = getContext().getResources();

        if (getUser().getuserName().compareTo("") == 0) {
                return r.getString(R.string.err_usr_blank);
        }
        if (pwdCheck && getUser().getpassword().length() < 8) {
                return r.getString(R.string.err_pwd_blank);
        }
        if (getUser().getfirstName().compareTo("") == 0) {
                return r.getString(R.string.err_name_blank);
        }
        if (getUser().getlastName().compareTo("") == 0) {
                return r.getString(R.string.err_surname_blank);
        }
        if (getUser().geteMail().compareTo("") == 0) {
                return r.getString(R.string.err_e_mail_blank);
        }
        else return null;
}

/*
 * Genera id usuari aleatori
 */
private void generateRndUserId() {
        Calendar c = new GregorianCalendar();

        StringBuffer sb = new StringBuffer();
```

```
        sb.append(getUser().getfirstName().substring(0, 1));
        sb.append(getUser().getlastName().substring(0, 1));
        sb.append(Integer.toString(c.get(Calendar.YEAR)));
        sb.append(Integer.toString(c.get(Calendar.MONTH)));
        sb.append(Integer.toString(c.get(Calendar.DATE)));
        sb.append(Integer.toString(c.get(Calendar.HOUR_OF_DAY)));
        sb.append(Integer.toString(c.get(Calendar.MINUTE)));
        sb.append(Integer.toString(c.get(Calendar.SECOND)));
        sb.append(Integer.toString(c.get(Calendar.MILLISECOND)));

        getUser().setIdNumber(sb.toString());
}
```

Figura 33 – Exemple de codi d'un gestor de la capa de negoci

6.4. Capa d'accés a dades i *Web Services*

Tal com s'ha detallat a l'apartat 5.3.3. a la capa de negoci hi ha 2 classes gestores, una per accés a dades i l'altre per fer crides a *Web Services*. Atès que aquesta capa té especial rellevància en l'arquitectura de l'aplicació, seguidament es dediquen uns apartats a detallar-la.

6.4.1. Accés a dades

Tal com s'ha descrit anteriorment, aquesta aplicació no implementa cap base de dades ni cap persistència atès que totes les dades resideixen en el servidor Moodle. Conseqüentment, el gestor de base de dades *DBMgr* –que és sempre cridat des del gestor *PrefsMgr* de la capa de negoci– és molt simple i té l'única responsabilitat de llegir i escriure les preferències de l'aplicació Android, que en aquest cas consisteixen només en el l'adreça del servidor Moodle.

Per accedir a les preferències es fa servir la funcionalitat que incorpora Android. Aquesta característica permet crear espais de preferències en l'àmbit de cada aplicació per emmagatzemar tipus de dades natius. Aquests espais es poden llegir i escriure en qualsevol moment des de l'aplicació i persisteixen mentre no es desinstal·li l'*App* del dispositiu, la qual cosa els fa ideals per emmagatzemar preferències i altres dades simples.

El següent fragment de codi il·lustra la classe *DBMgr* i els mètodes per llegir i escriure el valor *server* de l'espai de preferències. Com es pot veure es tracta d'un gestor molt senzill.

```
public class DBMgr {

        private SharedPreferences prefs;
        private static final String DEFAULT_SRV = "http://62.82.98.156/";

        public DBMgr(SharedPreferences prefs) {
                this.prefs = prefs;
        }

        /*
         * Llegeix preferències
         */
        public Prefs readPrefs() {
                String server = prefs.getString("server", DEFAULT_SRV);
                return new Prefs(server);
        }
```

```
        /*
         * Escriu preferències
         */
        public void writePrefs(Prefs settings) {
                SharedPreferences.Editor editor = prefs.edit();
                editor.putString("server", settings.getServer());
                editor.commit();
        }
}
```

Figura 34 – Exemple de codi del gestor de dades de la capa d'accés a dades

6.4.2. Accés a *Web Services*

L'accés als *Web Services* per fer crides a l'API de Moodle és un dels pilars d'aquesta aplicació, per tant, el gestor *WSMgr* és una peça clau de la seva arquitectura.

El gestor *WSMgr* té una única responsabilitat: *traduir* totes les peticions dels gestors de la capa de negoci en crides a l'API de Moodle i retorna'ls-hi els resultat.

Tot seguit es detalla el funcionament, en primer lloc de l'inici de sessió, necessari per poder realitzar qualsevol crida, i en segon lloc d'una crida des d'un gestor de la capa de negoci.

6.4.2.1. Inici de sessió

Els següents passos descriuen el procés que segueix una petició d'inici de sessió:

- Quan l'usuari inicia sessió, *WSMgr* fa una petició de connexió al servidor Moodle fent servir 3 paràmetres:
    - L'adreça del servidor indicada a les preferències de l'aplicació.
    - El nom del servei del servidor Moodle que és "wspp/service_pp2.php" tal com està codificat en el gestor *WSMgr* el qual permet accedir als *Web Services* del *plugin* "OK Tech Web Services" instal·lat al servidor Moodle.
    - El nom de l'espai de noms, que és "json" tal com està codificat en el gestor *WSMgr* el qual possibilita l'intercanvi de missatges en format JSON.
- Una vegada establerta la connexió, *WSMgr* crida al mètode *login()* de l'API fent servir el compte d'usuari i contrasenya que ha indicat l'usuari i que ha arribat des de la capa de presentació, passant per la de negoci.
- Si és satisfactori retorna un objecte *LoginReturn* amb les dades de la sessió, que són la clau de sessió i el client i que seran necessàries per qualsevol petició posterior durant aquesta sessió. Altrament, retorna un error que serà enviat a la capa de presentació, passant per la de negoci, per ser mostrat a l'usuari.

- Les dades de la sessió, la clau de sessió i el client, es guarden a un objecte *Session* juntament amb el codi d'usuari, l'identificador i el nom, i es retornen a la capa de presentació, passant per la de negoci, per ser reutilitzades en crides posteriors.

- Finalment, l'objecte WSMgr es destrueix i la connexió amb el servidor finalitza deixant la sessió iniciada. Això és important atès que mantenir la connexió activa consumiria recursos innecessàriament i aquest és una aspecte al que cal parar especial atenció en el disseny i desenvolupament d'aplicacions mòbils.

El següent fragment de codi mostra les instruccions d'inici de sessió:

```
/*
 * Inici sessio
 */
private void login(String usr, String pwd, String token) {

        moodle = new Mdl_restserverBindingStub(getUrl() + MOODLE_SERVICE,
              MOODLE_NAMESPACE, false);
        lr = moodle.login(usr, pwd);

        if (lr != null) {
                String key = lr.getSessionkey();
                int client = lr.getClient();
                int userId = moodle.get_my_id(client, key);

                UserRecord[] res = moodle.get_user_byid(client, key, userId);
                String userDesc = res[0].getFirstname() + " " +
                        res[0].getLastname();

                setSession(url, key, client, userId, usr, userDesc);
        } else {
                session = null;
        }
}
```

Figura 35 – Codi d'inici de sessió del gestor de *Web Services*

6.4.2.2. Crida des d'un gestor de la capa de negoci

Els següents passos descriuen el procés que segueix una crida des d'un gestor de la capa de negoci per realitzar una acció sobre el servidor Moodle:

- Una vegada la sessió està iniciada, quan un gestor de la capa de negoci necessita realitzar una acció sobre el servidor Moodle crea una instància de *WSMgr* passant-li l'objecte *Session* que, com hem vist a l'apartat anterior, conté les dades de la sessió activa.

- *WSMgr* crea una connexió amb el servidor tal com s'ha descrit a l'apartat anterior fent servir els mateixos paràmetres: nom del servidor, nom del servei i nom de l'espai de noms.

- Si es produeix algun error, com per exemple que ja no es pugui accedir al servidor o que la sessió ja no sigui vàlida (s'ha finalitzat o ha caducat), es mostrarà el

corresponent missatge d'error a la capa de presentació, com sempre, passant per la de negoci.

- Altrament, amb les dades de la connexió es farà la crida al mètode de l'API passant-li les dades necessàries, que sempre són la clau de sessió i el client més les dades específiques de l'acció concreta que es vulgui realitzar. Per exemple, si el que es vol és llegir un usuari per nom, caldrà passar un *string* amb el nom d'usuari.

El següent fragment de codi mostra les instruccions d'una crida a un mètode de l'API, en aquest cas llegir un usuari per nom:

```
/*
 * Llegeix usuari per nom
 */
public UserRecord[] readUserByUserName(String usr) {
        moodle = new Mdl_restserverBindingStub(session.getUrl() + MOODLE_SERVICE,
              MOODLE_NAMESPACE, false);
        return moodle.get_user_byusername(session.getClient(), session.getKey(),
              usr);
}
```

Figura 36 – Exemple de codi d'una crida de la capa de *Web Services*

6.4.3. Ús de les llibreries "MoodleWS_gson" i "Google Gson"

Recordant el que es descriu amb detall a l'apartat 3.3.3.2., per l'ús de *Web Services* amb l'API de Moodle l'aplicació fa servir el protocol REST i el format JSON atenent a les següents raons:

- Moodle no ofereix nativament una API per Java però disposa del plugin "OK Tech Web Services" que ofereix funcionalitats addicionals per aquesta finalitat.
- L'enviament de dades amb el protocol REST es fa en format JSON.
- Google disposa de les llibreries Java "Google Gson" per la conversió entre classes Java i format JSON i viceversa.
- Google recomana l'ús de "Google Gson" en aplicacions Android.
- Hi ha disponibles llibreries Java per l'ús de "OK Tech Web Services" amb REST i JSON.

Amb l'objectiu de poder fer servir d'una forma senzilla el *plugin* "OK Tech Web Services" amb REST i JSON s'han d'importar les llibreries "MoodleWS_gson" i "Google Gson" seguint les instruccions descrites en el "Manual de configuració de l'entorn de desenvolupament"[23].

[23] Veure capítol "Manual de configuració de l'entorn de desenvolupament" del document "Manuals"

6.4.3.1. Llibreria "MoodleWS_gson"

Aquesta llibreria de lliure distribució proporciona un conjunt de classes i mètodes Java per fer crides als serveis que proporciona "OK Tech Web Services" utilitzant el protocol REST i enviament de missatge en format JSON.

Aquesta llibreria té com a format de sortida JSON, que és el que suporta el protocol REST que es fa servir per la comunicació entre l'aplicació i el servidor, consegüentment, és necessari un procés de conversió entre JSON i objectes Java. Per aquesta conversió, el més convenient és la llibreria "Google Gson" que es descriu a continuació.

6.4.3.2. Llibreria "Google Gson"

"Google Gson" és una llibreria de classes Java que té per objecte facilitar utilitats per convertir objectes Java en una representació JSON. També es pot fer servir en sentit contrari, és a dir per convertir una cadena JSON en el seu objecte Java equivalent.

Per tal d'il·lustrar el procés és convenient revisar la figura 5 de l'apartat 3.3.3.2. El funcionament és el següent:

- Quan des de l'aplicació Android *WSMgr* fa una crida a l'API de Moodle, el que realment fa es cridar a un mètode de la llibreria "MoodleWS_gson". Cal fer menció a que en determinats casos *WSMgr* fa servir classes auxiliars, com *UserDatum* o *CourseDatum*, que també pertanyen a la llibreria "MoodleWS_gson".

- El mètode cridat converteix a forma JSON la petició fent servir les classes Java que proporciona la llibreria "Moodle Gson".

- Envia el missatge JSON als *Web Services* d'"OK Tech Web Services" fent servir el protocol REST.

- El servidor Moodle processa la petició i genera un resultat que pot ser un conjunt de dades o un error.

- Els *Web Services* d"OK Tech Web Services" retornen un missatge JSON a l'aplicació Android fent servir el protocol REST.

- "MoodleWS_gson" converteix el missatge JSON en un objecte Java gràcies a les classes Java que proporciona la llibreria "Moodle Gson".

- "MoodleWS_gson" retorna el resultat a l'aplicació Android, la qual els processa.

6.5. Organització del projecte Java

Per cadascuna de les capes de l'arquitectura de l'aplicació s'ha creat un paquet que conté les classes que li corresponen. Addicionalment, s'han creat dos paquets més per utilitats de la capa de presentació i per les classes entitat.

L'estructura és la següent:

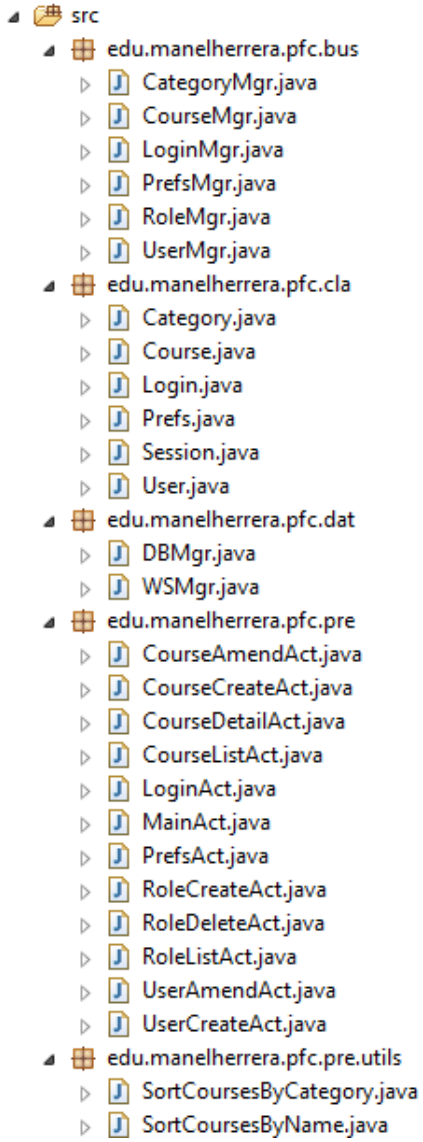


Figura 37 – Estructura de paquets i classes

6.6. Canvis sobre el disseny inicial

Durant la fase d'implementació ha sorgit la necessitat de fer alguns canvis respecte al que s'havia previst en el disseny. Aquests canvis són menors i no afecten als requeriments funcionals ni a l'arquitectura del sistema. No obstant, alguns d'ells alteren lleugerament alguns aspectes pràctics de l'aplicació, com per exemple la selecció d'idioma o els mètodes d'autenticació. Són els següents:

#	Motivació	Canvi - Decisió presa	Justificació
1	S'aconsella que sigui Android qui gestioni els diferents idiomes de les aplicacions i no les pròpies aplicacions	Eliminar l'opció de seleccionar l'idioma a les preferències de l'aplicació i deixar que sigui l'idioma seleccionat al dispositiu el que determini l'idioma de l'aplicació	Seguir les recomanacions d'Android per evitar que l'aplicació estigui en contraposició amb la gestió d'idioma que incorpora el sistema operatiu incorpora de forma nativa
2	Els *Services* d'Android són exclusivament per serveis que s'executen en segon pla mentre l'usuari interacciona amb una *Activity* o mentre s'executa una altra aplicació	No fer servir classes de tipus *Service* i implementar tota l'aplicació amb *Activities* i classes complementàries (gestors)	Atès que l'aplicació desenvolupada és totalment interactiva no es faran servir *Services* eliminant la complexitat del *binding* entre *Activities* i *Services*
3	"OK Tech Web Services" no permet iniciar una sessió *Web Services* fent servir un *token* preestablert	Descartar la possibilitat d'iniciar sessió amb un *token* i deixar només la d'usuari/contrasenya	No és possible iniciar una sessió *Web Services* fent servir un *token* preestablert

Taula 23 – Canvis sobre el disseny en la fase d'implementació

7. Conclusions i línies de futur

7.1. Conclusions

Amb la realització d'aquest projecte ha estat possible implementar amb èxit una aplicació Android per Moodle fent servir comunicacions a través de xarxes sense fils i complint amb els requeriments i els objectius definits al començament del projecte.

En el seu inici, el principal repte –i a la vegada risc– era el desconeixement general tant de l'entorn de desenvolupament (dispositius mòbils Android), com del marc d'execució (plataforma Moodle), com de les tecnologies de comunicació de xarxa (protocols i formats de missatges), com de la utilització d'una API com la de Moodle.

Per afrontar aquest repte i capgirar el desconeixement general fins convertir-lo en una oportunitat d'aprenentatge era necessari aplicar una bona metodologia de gestió de projectes, la qual ha estat clau per planificar correctament totes les activitats i facilitar-ne el seguiment.

Durant el projecte s'ha aprofundit en els següents aspectes:

- L'us de Moodle i l'estudi de la seva API.
- El desenvolupament mòbil, concretament en Android.
- L'ús de xarxes per la comunicació amb protocols com REST en format com JSON.

L'experiència ha estat del tot enriquidora i em proporciona una visió global de com gestionar un projecte des de zero partint d'una definició escassa i amb un coneixement quasi nul del marc de desenvolupament del projecte.

Havent finalitzat el projecte, seria necessari repassar els quatre pilars presentats al principi i veure les conclusions, lliçons apreses i dificultats trobades:

- Començant per la tecnologia Android, estic molt satisfet d'haver-me pogut iniciar en aquest món, encara més tenint en compte que Android és el sistema pel que es vaticinen més èxits en un futur immediat. He trobat que és una plataforma potent i senzilla a la vegada, la qual cosa és un mèrit si es té en compte la complexitat i les limitacions imposades pels dispositius mòbils. A més, disposa d'una enorme quantitat de documentació, per això el seu aprenentatge m'ha resultat fàcil i divertit.

  Malgrat tot, he tingut algunes dificultats que no he pogut superar, com per exemple la creació de *threads* per tal crear processos paral·lels per millorar la implementació dels accessos al servidor. Un altre inconvenient amb que m'he trobat ha estat l'elevat nombre de dispositius i tipus de pantalles, resolucions, mides i orientacions que hi ha al mercat, la qual cosa ha impossibilitat provar l'aplicació per totes les combinacions possibles i m'ha obligat a centrar-me en el dispositiu seleccionat, el SAMSUNG Galaxy Tab.

- Pel que fa a Moodle com a plataforma, m'ha sobtat la seva magnitud, el seu elevat grau d'implantació i la gran comunitat que l'envolta. En canvi, no puc dir el mateix de

la seva nova API. La manca de documentació específica ha estat una de les principals dificultats d'aquest projecte. Tasques aparentment trivials com la configuració dels seus *Web Services* es converteixen en complexes degut a la documentació gairebé inexistent. No obstant això, hi ha alguns col·laboradors de la comunitat Moodle que amb les seves contribucions alleugen aquestes mancances.

- En quant a la utilitat de la seva API, el que més m'ha sobtat és el nombre reduït de funcionalitats que ofereix. Actualment només ofereix serveis pel manteniment d'alguns arxius mestres (usuaris, cursos, matriculacions, grups) i poca cosa més, sense permetre que un alumne pugui veure la planificació d'activitats del semestre, accedir als materials o consultar l'avaluació. En la meva opinió, l'actual API pot ser útil pels administradors del sistema i alguns docents o bé per institucions que tinguin altres aplicacions de gestió acadèmica i vulguin crear interfícies amb Moodle per tal de simplificar tasques administratives i evitar duplicitats, però no per l'alumnat que té necessitats totalment diferents.

- Finalment, pel que fa a les xarxes, per pràcticament tota la tecnologia, protocols, llenguatges i formats que es fan servir en aquest projecte ja existeixen solucions, amb la qual cosa només fa falta aplicar-les i no cal fer desenvolupaments específics. Per exemple:

    - Tots els dispositius mòbils permeten connexió a Internet i són capaços de commutar entre WiFi i 3G sense intervenció de l'usuari i sense necessitat de desenvolupar cap solució que ho gestioni.

    - En tots els dispositius mòbils TCP/IP funciona gairebé sense necessitat de configuració.

    - Per la conversió de Java a JSON existeix la llibreria "Google Gson" que proporciona els mètodes necessaris i que és suficient amb incorporar-la al projecte per fer-la servir.

    - Per l'accés al *Web Services* de Moodle existeix el pluggin "OK Tech Web Services" i la llibreria "MoodleWS_gson" que facilita la programació de crides a l'API de Moodle.

  Tot això que per una banda és un avantatge ja que és suficient aplicar les solucions existents per resoldre un problema, és un inconvenient des del punt de vista de l'aprenentatge. No obstant això, he pogut comprovar la funcionalitat i l'aplicabilitat de la reutilització de components a l'hora que he après el funcionament bàsic d'aquestes tecnologies.

7.2. Línies de futur

A la vista de les dificultats i limitacions trobades durant el projecte, les línies de futur que sens dubte seguiria són les següents:

- Completar la implementació de les funcionalitats implementades, per exemple possibilitant la creació o modificació d'usuaris, cursos i matrícules amb totes les seves dades (ara només hi ha les dades bàsiques).

- Completar la implementació de la resta de funcionalitats que ofereix Moodle i que ara no s'han implementat.

- Implementar les millores necessàries per que l'aplicació pugui funcionar correctament en qualsevol dispositiu Android i no només en el seleccionat

- Participar en la comunitat de Moodle per ampliar les funcionalitats de l'API i millorar la seva documentació.

- Millorar alguns aspectes no funcionals de l'aplicació Android, com per exemple la creació de *threads* per poder realitzar activitats en paral·lel, la possibilitat de tenir una llista de servidors preferits o ampliar el nombre d'idiomes (ara només hi ha català, castellà i anglès).

8. Glossari

.NET Projecte de Microsoft para crear una nova plataforma de desenvolupament de programari que permeti un ràpid desplegament d'aplicacions.

3G Tercera generació de transmissió de veu i dades a través de telefonia mòbil amb UMTS.

802.11 Estàndard que defineix l'especificació i les normes de funcionament de les xarxes WLAN.

802.16 Estàndard que defineix l'especificació i les normes de funcionament de les xarxes WiMAX.

Apache Servidor web de codi obert per diferents plataformes.

ADSL *Asymmetric Digital Subscriber Line*. Tecnologia d'accés a Internet de banda ampla.

ADT *Android Development Tools*. *Plugin* per Eclipse que proporciona eines pel desenvolupament d'aplicacions Android.

AMF *Action Message Format*. Format per serialitzar objectes en diferents formats, per exemple XML.

Android Sistema operatiu de codi obert dissenyat inicialment per a dispositius mòbils que actualment s'ha estès a altres tipus de dispositius d'ús domèstic i professional.

API *Application Programming Interface*. Conjunt de funcions i procediments que un sistema ofereix per ser utilitzats per altres programaris amb un cert nivell d'abstracció.

app De l'anglès, abreviatura d'*application* per fer referència a una aplicació de programari i especialment a una aplicació per a un dispositiu mòbil

AVD *Android Virtual Device*. Emulador de dispositius Android emprat en el desenvolupament d'aplicacions Android.

binding De l'anglès, enllaç. En programació és la referència a un altre objecte o element amb el que es fa una crida o un intercanvi.

C/C++ Llenguatge de programació orientat a objectes usat habitualment en la indústria del programari pel desenvolupament de llibreries a més del d'aplicacions comercials.

CRUD De l'anglès, acrònim de *Create*, *Read*, *Update* i *Delete* que es fa servir per referir-se a operacions bàsiques de base de dades o d'interfícies d'usuari

Dalvik Veure DVM.

driver De l'anglès, controlador que permet a un sistema operatiu o un programari interactuar amb un dispositiu perifèric proporcionant una interfície que permeti a aquests abstraure's del maquinari.

DVM *Dalvik Virtual Machine*. Màquina virtual que es fa servir en els dispositius mòbils Android.

DMZ De l'angles, acrònim de *DeMilitarized Zone* que es fa servir per referir-se a una xarxa perimetral que generalment s'ubica entre la xarxa local i Internet

Eclipse IDE de codi obert multiplataforma típicament emprat per desenvolupament d'aplicacions Java.

e-learning De l'anglès, sistema d'educació electrònica o a distància a través d'Internet.

framework De l'anglès, marc de treball. En l'entorn de desenvolupament de programari és una estructura de suport per organitzar i desenvolupar aplicacions.

IDE *Integrated Development Environment*. Programari que integra diferents facilitats pel desenvolupament d'aplicacions de programari.

Intellij IDEA IDE pel desenvolupament d'aplicacions Java.

GPL *General Public License.* Llicència Pública General orientada a protegir la lliure distribució, modificació i ús del programari lliure.

i18n Abreviació de *internationalization*, de l'anglès, internacionalització.

IEEE *Institute of Electrical and Electronics Engineers.* Institució mundial dedicada a la estandardització, principalment en els àmbits de l'enginyeria elèctrica, electrònica, de telecomunicacions i informàtica.

ISP *Internet Service Provider.* Empresa que ofereix al seus clients connexió a Internet i altres serveis com correu electrònic, registre de dominis web, etc.

Java Llenguatge de programació orientada a objectes.

JDK *Java Development Kit*. Programari per ser instal·lat en un ordinador que proveeix d'eines de desenvolupament en llenguatge Java.

JRE *Java Runtime Environment.* Conjunt d'utilitats i llibreries per possibilitar l'execució de programes Java, també conegut com Màquina Virtual de Java.

JVM *Java Virtual Machine*. Màquina virtual multiplataforma que es fa servir per interpretar programari escrit en llenguatge Java.

L10n Abreviació de *localization*, de l'anglès, localització.

LAN *Local Area Network*. Xarxa d'àrea local.

Linux Sistema operatiu de codi obert sota llicència GPL.

layout De l'anglès, esquema de distribució. En disseny d'interfícies es fa servir per referir-se a la distribució dels seus elements.

Moodle *Module Object-Oriented Dynamic Learning Environment*. Plataforma d'*e-learning* de codi obert per l'organització d'entorns educatius.

MySQL Sistema de gestió de base de dades relacional.

Netbeans IDE pel desenvolupament d'aplicacions Java.

Oracle Companyia de programari propietària de Java, entre d'altres.

Perl Llenguatge de programació basat en blocs amb característiques de molts altres llenguatges.

PHP Llenguatge de programació interpretat usat habitualment en pàgines web dinàmiques.

plugin De l'anglès, complement, extensió o afegit que en ser instal·lat proporciona funcionalitat addicional a un programari.

REST *Representational State Transfer.* Tècnica d'arquitectura de programari que en sistemes distribuïts permet que aplicacions client-servidor intercanviïn dades.

runtime De l'anglès, temps d'execució, interval de temps que un programari necessita per executar-se.

SDK *Software Development Kit*. Programari per ser instal·lat en un ordinador que proveeix d'eines de desenvolupament d'un determinat llenguatge.

SOAP *Simple Object Access Protocol*. Protocol per l'intercanvi de dades XML per facilitar la comunicació entre dos processos.

SQLite Sistema de gestió de bases de dades contingut en una llibreria de dimensions reduïdes per tal de poder ser executada en dispositius amb recursos limitats com ara dispositius mòbils.

suite De l'anglès, recopilació d'utilitats o aplicacions de programari que resolen un problema concret.

token De l'anglès, fitxa, que en termes informàtics correspon una seqüència de caràcters que té un determinat significat, per exemple una instrucció o un identificador.

thread De l'anglès, subprocés, generalment emprat en programació o sistemes operatius per referir-se a diferents processos o fils d'execució actius.

UMTS *Universal Mobile Telecommunications System.* Tecnologia emprada pels mòbils de tercera generació que representa un increment considerable de velocitat i serveis respecte als seus predecessors.

USB *Universal Serial Bus.* Tecnologia que permet connectar perifèrics a un ordinador a través d'un port universal.

VLE *Virtual Learning Environment.* Entorn d'Aprenenetage Virtual. Plataforma d'*e-learning*.

WiFi Sistema de connexió sense fils per dispositius electrònics.

WiMAX *Worldwide Interoperability for Microwave Access*. Norma per la transmissió de dades que combina ones de ràdio i microones generalment utilitzada en xarxes d'àrea metropolitana. Respon a la norma IEEE 802.16.

WLAN *Wireless Local Area Network*. LAN que incorpora connexió sense fils.

XAMPP Servidor independent de la plataforma que consisteix en un servidor web Apache, una base de dades MySQL i intèrprets de PHP i Perl.

XML *Extensible Markup Language*. Metallenguatge d'etiquetes per l'intercanvi de dades i definició d'estructures de dades en diferents aplicacions com ara bases de dades o interfícies d'usuari.

XML-RPC *Extensible Markup Language – Remote Procedure Call*. Protocol d'intercanvi de missatges que utilitza crides a procediment remots en format XML.

9. Bibliografia

ANDROID. *Google-Android.* [En línia]. http://www.android.com/.
[Data consulta: 05/Octubre/2011]

CLARISÓ, ROBERT. (2011). *Orientacions Generals per al Treball Final.* Barcelona: Universitat Oberta de Catalunya, Estudis d'Informàtica i Multimèdia.

ECLIPSE. *The Eclipse Foundation.* [En línia]. http://www.eclipse.org/.
[Data consulta: 05/Octubre/2011]

ELINUX. *Embedded Linux Wiki.* [En línia]. http://elinux.org/Main_Page .
[Data consulta: 25/Octubre/2011]

FLICKER. *Moodle Web Services Architecure images.* [En línia].
http://farm3.static.flickr.com/2164/2403150142_05d80d3f1c.jpg .
[Data consulta: 25/Octubre/2011]

GITHUB. *Github social Coding.* [En línia]. https://github.com/ (requereix autenticació).
[Data consulta: 05/Octubre/2011]

GOOGLE-GSON JAVA LIBRARY. *Google.* [En línia]. http://code.google.com/p/google-gson/.
[Data consulta: 05/Octubre/2011]

JAVA. *Oracle-Java.* [En línia]. http://java.com/es/.
[Data consulta: 05/Octubre/2011]

JSON. *Introducción a JSON.* [En línia]. http://www.json.org/json-es.html .
[Data consulta: 31/Octubre/2011]

LEARN REST: A TUTORIAL. *A fast-training course for REST.* [En línia].
http://rest.elkstein.org/2008/02/what-is-rest.html
[Data consulta: 31/Octubre/2011]

MOODLE. *Moodle.* [En línia]. http://moodle.org/.
[Data consulta: 05/Octubre/2011]

PEREZ NAVARRO, ANTONI I ALTRES. (2008). *Treball final de carrera.* Barcelona: Universitat Oberta de Catalunya, Estudis d'Informàtica i Multimèdia.

SAMSUNG. *Samsung.* [En línia]. www.samsung.com.
[Data consulta: 25/Octubre/2011]

PRIETO BLÁZQUEZ, JOSEP I ALTRES. (2011). *Tecnologia i desenvolupament en dispositius mòbils.* Barcelona: Universitat Oberta de Catalunya, Estudis d'Informàtica i Multimèdia.

WIKIPEDIA. *Wikipedia - La enciclopedia libre.* [En línia]. www.wikipedia.es.
[Data consulta: 25/Octubre/2011]

Annex I – Descripció bàsica de l'API de Moodle

moodle

Web services

From MoodleDocs

Moodle 2.0

Contents

Introduction

This page describes the Web Services implementation that was added to Moodle 2.0 and is being extended with every subsequent version.

We support multiple web service protocols (REST, SOAP, XML-RPC and AMF). If that's not enough, support for new protocols can be added easily.

How it works

This simple example will give you an idea of how our web services infrastructure works. See External services security for more details.

1. The client sends a username and password to the web service protocol server script.
2. The protocol server returns a session token for that user account (how this is sent depends on the protocol).
3. The client calls a particular web service function including the session token.
4. The protocol server uses the token to check that the web service session is still active.
5. The protocol server call the matching external function, located in a externallib.php file inside the relevant module.
6. The external function checks that the current user has_capability to do this operation.
7. The external function calls the matching Moodle core function (in lib.php usually).
8. The core function can return a result to the external function.
9. The external function will return a result to the protocol server.
10. The protocol server returns the result to the client.

Web description format

All functions that can be called via web services are declared in a db/service.php file and can be defined in an externallib.php file somewhere. The classes we use to define these functions are explained in the External Service description document (http://docs.moodle.org/en/Development:Web_services_description) .

Web services functions

Every Moodle site can provide you with a page listing all function descriptions, with useful advice for any supported protocol.

At the moment the only way to get the full Moodle web service documentation is:

1. login as admin
2. enable ws documentation into Admin block > Web service > Manage Protocol
3. create a service and add all Moodle functions to it
4. give create_token capability to a none admin user
5. login as this user
6. access your Security Keys page and click on the documentation.

An issue has been created to make it a lot easier: MDL-28650 (http://tracker.moodle.org/browse/MDL-28650)

However, here is a list of the functions available in Moodle 2.1:

user:

- create_users()
- delete_users()
- update_users()
- get_users_by_id() - returns a list of full user objects specified by user ids (only useful for site admin accounts)
- get_course_participants_by_id() - returns a list of full user objects in specific courses (that you can see)
- get_users_by_courseid() - returns some subset of full user objects from a course (perhaps by group or capability)

course:

- get_courses()
- create_courses()

group:

- create_groups()
- get_groups()
- get_course_groups()
- delete_groups()
- get_groupmembers()
- add_groupmembers()
- delete_groupmembers()

enrol:

- get_enrolled_users() - get some minimal information about the users enrolled in a course (DEPRECATED)
- get_users_courses() - get list of course ids that a user is enrolled in (if you are allowed to see that)
- role_assign()
- role_unassign()

enrol/manual:

- manual_enrol_users()

webservice:

- get_siteinfo_parameters()

message:

- send_instantmessages()

files:

- get_files()
- upload()

Authentication

Please have a look to the External services security page

See also

- The official discussion forum is here (http://moodle.org/mod/forum/view.php?id=6971)
- The tracker issues are here: MDL-12886 (http://tracker.moodle.org/browse/MDL-12886) and MDL-17135 (http://tracker.moodle.org/browse/MDL-17135)
- External services security
- External services description
- Creating a web service and a web service function
- Creating a web service client
- Web_Services

Retrieved from "http://docs.moodle.org/dev/Web_services"
Categories: Moodle 2.0 | Web Services

Font: http://docs.moodle.org/dev/Web_services_API

Annex II – Protocol REST

EL protocol REST (*Representational State Transfer*), és una tècnica d'arquitectura de programari per sistemes distribuïts que va ser publicada l'any 2000 per Roy Fielding, un dels principals autors de l'especificació del protocol HTTP.

Es basa en un protocol sense estat, per aplicacions client-servidor que necessitin realitzar un intercanvi de dades sense les complicacions que imposen altres protocols més complexos, com per exemple SOAP. Gairebé sempre fa servir XML i HTTP.

Està pensat per interfícies simples i que necessitin fer peticions de lectura, escriptura, actualització o eliminació, amb la qual cosa cobreix les quatre operacions típiques també conegudes com CRUD (*Create/Read/Update/Delete*).

Les característiques principals de REST són:

- És un protocol sense estat amb la qual cosa ni el client ni el servidor tenen necessitat de mantenir cap estat de comunicacions. Com a contrapartida, cada missatge ha de contenir tota la informació necessària per tal que l'emissor/receptor pugui processar la petició/resposta correctament.

- Comprèn el conjunt d'instruccions bàsiques PUT, GET, UPDATE i DELETE

- El direccionament de recursos es fa amb sintaxi universal atès que només accepta un URI com a identificador

- És independent de la plataforma

- És independent del llenguatge

- Està basat en l'estàndard HTTP

El principal inconvenient derivat de la seva simplicitat és la manca de seguretat. No obstant això, hi ha mecanismes per minimitzar aquest risc:

- Fer servir la tècnica de *tokens* en comptes d'enviar l'usuari/contrasenya en clar

- Fer servir HTTPS com a protocol d'aplicació amb xifrats basats amb SSL/TLS

Com a conclusió, REST és un protocol simple, sense estat, independent del llenguatge, independent de la plataforma i basat en HTTP, la qual cosa el fa compatible amb la pràctica totalitat de sistemes i el converteix en la solució ideal per l'intercanvi de missatges en aplicacions client-servidor senzilles.

Annex III – Format JSON

El format JSON (*JavaScript Object Notation*) és un format lleuger d'intercanvi de dades. Es basa en un subconjunt de la notació d'objectes del llenguatge de programació JavaScript.

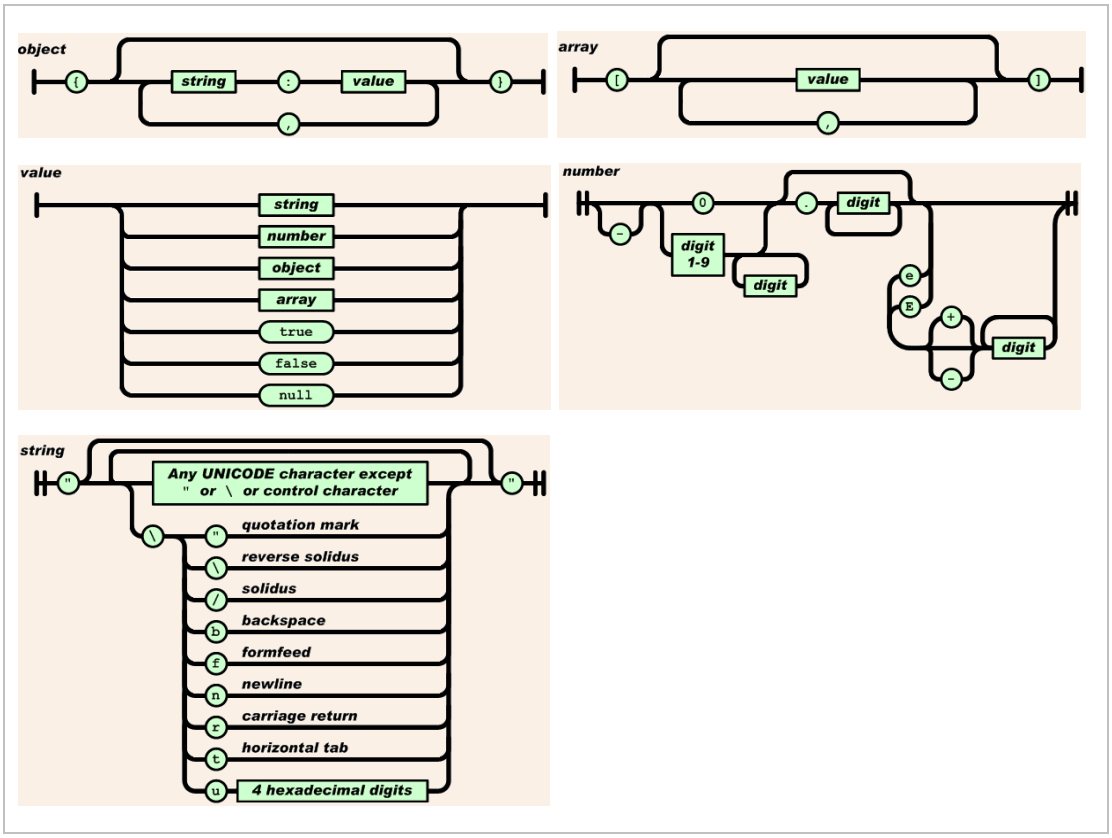
Degut a la seva simplicitat, la seva acceptació ha crescut entre els desenvolupadors, sobretot per la senzillesa a l'hora d'escriure un analitzador sintàctic (*parser*) i perquè hi ha llenguatges, com JavaScript, que ja incorporen funcions pel seu anàlisi. Addicionalment, JSON fa servir sintaxi i convencions que fan servir la majoria de llenguatges orientats a objectes. Totes aquestes raons fan que JSON sigui un llenguatge ideal per a l'intercanvi de dades.

JSON es composa de dues estructures:

- Una col·lecció de parells nom/valor com pot ser un objecte, registre, estructura, diccionari, taula *hash*, llista de claus o *arrays* associatius

- Una llista ordenada de valors implementada com *arrays*, vectors, llistes o seqüències

La següent figura il·lustra les estructures amb que es presenten les representacions de dades JSON:



Font: http://www.json.org/json-es.html