Entorns de programació mòbils

PID_00245984

Julián David Morillo Pozo
Javier Salvador Calvo

Temps mínim de dedicació recomanat: 4 hores

Universitat Oberta de Catalunya

Índex

Introducció

El desenvolupament d'aplicacions mòbils és el procés pel qual es desenvolupa un programari per a dispositius mòbils com telèfons intel·ligents (*smartphones*), ordinadors corporals (*wearables*) o tauletes tàctils (*tablets*). La forma de distribució d'aquestes aplicacions pot variar: poden venir preinstal·lades als telèfons o bé les poden baixar els usuaris des de repositoris d'aplicacions (*app stores*) i altres plataformes de distribució de programari (*software*).

En aquest mòdul veurem els diferents entorns de programació per a aplicacions mòbils existents fent primer una revisió històrica de l'evolució que han tingut. Després farem una revisió dels diferents llenguatges de programació que es poden utilitzar dins d'aquests entorns. Finalment estudiarem a fons alguns dels entorns més populars.

Objectius

Amb l'estudi d'aquest mòdul es pretén que l'estudiant assoleixi els objectius següents:

1. Conèixer i comprendre el concepte d'*entorn de programació* dins de l'àmbit del desenvolupament d'aplicacions per a dispositius mòbils.

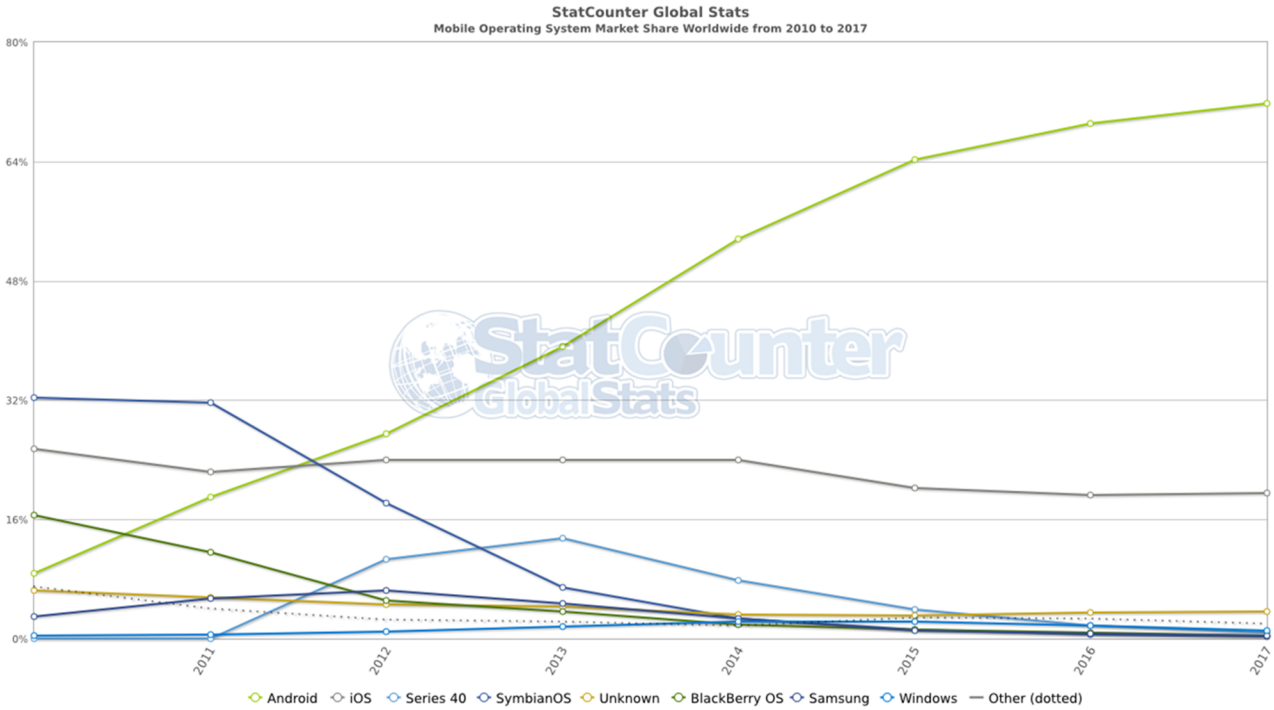
2. Identificar els diversos entorns de programació existents, coneixent-ne les arquitectures, les característiques dels sistemes operatius usats i les tècniques de programació requerides en cadascun.

3. Comprendre el fet que els entorns no funcionen aïlladament, sinó que coexisteixen.

4. Estar en condicions de triar l'entorn de programació idoni, d'acord amb els requisits de l'aplicació mòbil que es pretén desenvolupar.

1. Història i evolució dels entorns de programació mòbils

La indústria dels dispositius i les aplicacions mòbils és un entorn en canvi constant. El 2009 el mercat mòbil estava fragmentat en moltes plataformes diferents: Symbian, iOS, Android, OS BlackBerry, Windows Mobile. A poc a poc, el mercat s'ha anat concentrant en dues plataformes: Android (Google) i iOS (Apple).



Data Source: http://gs.statcounter.com. Published under a Creative Commons Attribution-Share Alike 3.0 Unported License.

Alguns aspectes clau que van afavorir aquesta concentració van ser:

- El 2013 Microsoft va comprar la divisió de mòbils de Nokia amb l'objectiu de potenciar el seu sistema operatiu Windows Phone 8.1. Aquesta compra va suposar la desaparició gairebé total de Symbian. Malgrat aquesta compra Microsoft mai va aconseguir fer-se amb una quota significativa de mercat.

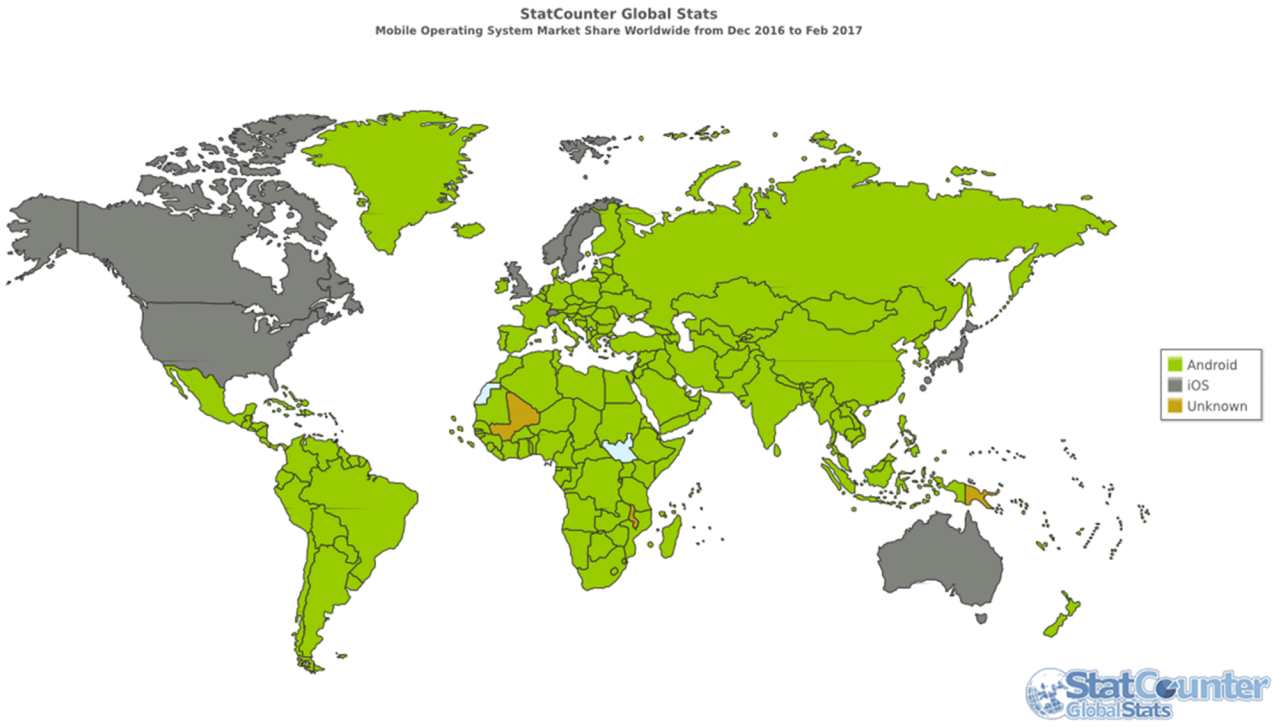
- Abans del 2013 BlackBerry ocupava el tercer lloc com a plataforma mòbil. Estava fortament implantada en entorns professionals. Va ser precisament el 2013 quan es va disparar la crisi dins de la companyia. Android i iOS li guanyaven terreny ràpidament en entorns professionals i en aquell moment Windows Mobile es veia com una gran amenaça que va arribar a treure-li el tercer lloc dins del rànquing de les plataformes.

Van aparèixer també altres sistemes alternatius minoritaris com els següents:

- Firefox OS és un sistema operatiu mòbil basat en HTML5 amb nucli Linux, de codi obert per a diverses plataformes, desenvolupat per Mozilla Corporation. El 2016 Mozilla va decidir posar fi al seu desenvolupament.

- Tizen és un sistema operatiu mòbil basat en la plataforma Linux de Samsung (Samsung Linux Platform - SLP). Tizen es va acabar utilitzant sobretot en dispositius *wearables* (Samsung Gear S) i en televisions intel·ligents (Smart TV de Samsung).

L'existència de molts dispositius Android de gamma baixa ha fet que la seva expansió sigui imparable. Destaca el domini d'Apple a Amèrica del Nord i diversos estats d'Europa, com el Regne Unit.



Data Source: http://gs.statcounter.com. Published under a Creative Commons Attribution-Share Alike 3.0 Unported License.

A principis del 2017 es va arribar als 2.800.000 milions d'aplicacions a Google Play i 2.200.000 a l'Apple Store. L'App Store d'Apple va duplicar els ingressos de Google Play durant els tres primers mesos de 2016, tot i que els usuaris descarreguen el doble d'aplicacions en el segon. D'aquestes dades es pot concloure que els usuaris d'iOS poden estar més predisposats a pagar per una aplicació, però hi ha un major nombre d'usuaris amb dispositius Android.

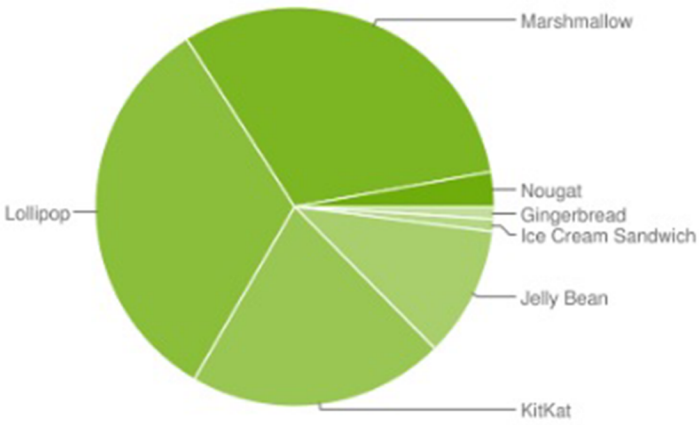
Una de les diferències més importants entre Android i iOS és la gran fragmentació del mercat Android. Aquesta fragmentació es dona a nivell de mida del dispositiu, de potència de maquinari i de versió de sistema operatiu. A

l'ecosistema Android podem trobar dispositius amb maquinari de baixes prestacions i que en moltes ocasions no es poden actualitzar a una versió superior del sistema operatiu.

Les diferències de maquinari inclouen entre d'altres la potència del processador, la presència o no de sensors com la brúixola (fonamental per a les aplicacions de realitat augmentada i per a afinar el guiat dels sistemes de posicionament), la versió de la llibreria gràfica Open GL ES suportada per la targeta gràfica, etc.

Un efecte d'aquesta fragmentació del sistema operatiu és l'augment de la complexitat en el moment de programar les aplicacions. Hem de fer servir llibreries diferents depenent de la versió de sistema on s'estigui executant la nostra aplicació.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.0%
4.1.x	Jelly Bean	16	3.7%
4.2.x		17	5.4%
4.3		18	1.5%
4.4	KitKat	19	20.8%
5.0	Lollipop	21	9.4%
5.1		22	23.1%
6.0	Marshmallow	23	31.3%
7.0	Nougat	24	2.4%
7.1		25	0.4%



Font: https://developer.android.com/about/dashboards/index.html.. Dades: març 2017

Aquest problema no és tan important a iOS, ja que els usuaris d'iOS solen actualitzar-se ràpidament a les darreres versions del sistema operatiu. Com Apple controla el maquinari i el sistema operatiu, resulta més senzill mantenir l'ecosistema controlat.

Atès l'elevat nombre d'aplicacions existents, un dels reptes als quals s'enfronten contínuament els desenvolupadors d'aplicacions mòbils és el de la promoció: com aconseguir que els usuaris descobreixin la seva aplicació. Les botigues d'aplicacions en tenen milers i les agrupen per categories, destacant-ne les més populars o més ben valorades. Això fa que una nova aplicació tingui una forta barrera d'entrada. Es podria dir que és com anar a una botiga

de discos amb dos-cents mil CD: només es mira el Top 10. Així doncs, en el desenvolupament cal preveure el cost de màrqueting i difusió, i pensar estratègies (per exemple, contingut viral) que donin a conèixer l'aplicació entre el seu públic objectiu.

Per altra banda, els usuaris poden ser reticents a comprar una aplicació sense haver-la provat. Per tal d'enfrontar-se a aquest problema, les estratègies que segueixen els desenvolupadors es basen a proporcionar versions gratuïtes de les aplicacions, que inclouen alguna estratègia de generació d'ingressos (monetització). Els models de negoci que es deriven d'aquestes aplicacions gratuïtes són diversos:

- Introduir publicitat dins de l'aplicació.
- Proporcionar compres dins de l'aplicació que ens permeten accedir a noves funcionalitats o suprimir limitacions (per exemple, eliminar la publicitat).
- Produir una versió prèmium de pagament.

Per tant, la conclusió que es pot treure de tot això és que cal estar molt atent per a veure com evoluciona el desenvolupament d'aplicacions, els beneficis, la distribució, la venda al detall, la portabilitat i la fragmentació, com a factors importants entre molts d'altres.

2. Aplicacions web, aplicacions natives i aplicacions híbrides

Abans d'entrar a fons en els entorns de programació d'aplicacions per a dispositius mòbils, establirem de manera resumida l'àmbit en el qual ens mourem. A continuació es defineixen termes clau i es comparen els avantatges i inconvenients dels tres paradigmes de desenvolupament més comuns.

Per començar, definirem què s'entén per *aplicació web*, *aplicació nativa* i *aplicació híbrida* i considerarem els avantatges i inconvenients de cadascuna.

Una **aplicació web** és bàsicament un lloc web específicament optimitzat per a un dispositiu mòbil. Les característiques que defineixen una aplicació web són que la interfície d'usuari es construeix amb tecnologies web estàndard, que és disponible en un URL[1] (públic, privat o protegit per una contrasenya) i que està optimitzada per als dispositius mòbils. Una aplicació web no està instal·lada al dispositiu mòbil.

(1)localitzador uniforme de recursos o *uniform resource locator*

En el cas de l'aplicació web, el lloc pot ser qualsevol, des d'una web anunci d'un petit negoci estàndard a un calculador d'hipoteques o un controlador de calories diari –el contingut és irrellevant.

Les **aplicacions natives**, per contra, estan instal·lades al dispositiu mòbil, tenen accés al maquinari o *hardware* (altaveus, acceleròmetre, càmera, etc.) i estan escrites en algun llenguatge de programació compilat com Objective-C.

Les **aplicacions híbrides** intenten combinar el millor dels dos models anteriors. Per a facilitar la creació i el manteniment de les aplicacions es basen en l'ús d'un llenguatge de programació comú per a totes les plataformes, donant accés a la vegada al maquinari del dispositiu. Algunes d'aquestes capacitats del maquinari són: rebre notificacions, accedir al sistema de fitxers local, accedir al valor dels sensors, etc.

Les més conegudes són les basades en l'ús de tecnologia web (HTML, JavaScript, CSS), que s'executen en un component que permet visualitzar web, natiu de cada plataforma. El problema d'aquest tipus d'aplicacions és el rendiment quan es treballa amb grans volums de dades o dispositius de gamma mitjana. PhoneGap és un dels entorns més coneguts per a generar aquest tipus d'aplicacions híbrides.

Un altre tipus són les basades en l'ús d'un intèrpret que executa un llenguatge comú per a totes les plataformes. El fabricant és el responsable d'optimitzar l'intèrpret per a cada plataforma. Com a exemples d'aquest model tenim Unity per a desenvolupament de videojocs 2D i 3D i Adobe Flash Builder per a desenvolupament de videojocs 2D i aplicacions amb un alt contingut de multimèdia.

Diferents aplicacions tenen diferents requisits. Algunes aplicacions s'adapten millor a les tecnologies web que d'altres. Conèixer els avantatges i inconvenients de cada paradigma ajudarà a decidir quin camí és apropiat per a cada situació.

El principal avantatge del desenvolupament d'aplicacions natives és que es pot accedir a totes les característiques de maquinari del dispositiu.

Els inconvenients del desenvolupament d'aplicacions natives són els següents:

- L'aplicació només funcionarà a la plataforma escollida.
- Cal desenvolupar-la usant el llenguatge de programació establert per a la plataforma.
- És més complicat distribuir pegats o actualitzacions que solucionin errors.
- El cicle de desenvolupament és més lent.

Swift / Objective-C

Per a implementar una aplicació nativa per a l'iPhone o iPad cal programar en Swift o Objective-C.

Quant al desenvolupament d'aplicacions web, aquests són els avantatges:

- Els desenvolupadors web poden usar les seves pròpies eines.
- Es poden usar els coneixements i les habilitats que ja es tinguin quant a disseny i desenvolupament web.
- L'aplicació funcionarà en qualsevol dispositiu que tingui un navegador web.
- Es poden solucionar errors en temps real.
- El cicle de desenvolupament és més ràpid.

Els inconvenients del desenvolupament d'aplicacions web són els següents:

- No es pot accedir a totes les característiques del dispositiu mòbil.

- Pot ser difícil aconseguir efectes sofisticats en la interfície d'usuari.

Quina aproximació és la millor en cada cas és un debat interessant. La naturalesa dels dispositius mòbils que cada vegada més estan sempre connectats fa que es difumini la línia entre aplicacions web i aplicacions natives. En moltes ocasions on el rendiment i l'accés a les capacitats del maquinari del dispositiu és limitat, la solució híbrida s'ha convertit en l'opció més habitual.

3. Enumeració dels diferents entorns

Microprogramari

El sistema operatiu d'un dispositiu es coneix en anglès com a *firmware*.

Igual que el sistema operatiu d'un ordinador, un sistema operatiu mòbil és la plataforma de programari que determina les funcions i les característiques disponibles al dispositiu, com control de teclats, seguretat sense fils, sincronització amb aplicacions, correu electrònic o missatges de text. El sistema operatiu mòbil determina també quines aplicacions de terceres parts es poden instal·lar al dispositiu. Per tant, cada sistema operatiu defineix uns entorns sobre els quals podem crear aplicacions. En aquest apartat farem un repàs dels més importants.

3.1. Entorns per a dispositius de diferents venedors

En aquest subapartat s'estudiaran les plataformes de programari que poden funcionar en diferents plataformes de maquinari de diferents fabricants. En concret, s'explica la història i les característiques principals de les següents:

- Java ME.
- Symbian.
- Android.
- Windows Mobile.
- Qt *framework*.
- BREW.
- Palm OS.

3.1.1. Java ME

El 1999, Sun va desenvolupar una versió de Java especialment dissenyada per a dispositius mòbils, Java 2 Micro Edition, basada en una màquina virtual anomenada *KVM*. Aquesta primera versió només contenia una única màquina virtual i una única API (inicialment dissenyades per al Palm OS), fet que va posar de manifest la insuficiència d'aquesta solució per a la gran varietat de dispositius diferents que hi havia. D'aquesta manera, el 2000 va néixer la primera versió d'una configuració, és a dir, el *connected limited device configuration* (J2ME CLDC 1.0). Una configuració ofereix l'API bàsica per a programar dispositius, encara que no aporta totes les classes necessàries per a desenvolupar una aplicació completa. Per tant, la primera configuració no tenia les eines necessàries per a permetre als desenvolupadors escriure programes per al dispositiu Palm. El juliol del 2000 va néixer la primera implementació d'un perfil, concretament anomenada *mobile information device profile* (MIDP), que no estava destinada a PDA sinó a telèfons mòbils i a paginadors. A partir d'aquest pri-

mer perfil, J2ME va ser considerablement acceptat per la comunitat de desenvolupadors de dispositius mòbils, i s'ha expandit a una gran velocitat fins als nostres dies.

Java ME[2] (anteriorment coneguda com a *J2ME*[3]) és, per tant, una edició de Java orientada a dispositius petits. És una versió retallada delJava SE amb certes extensions enfocades a les necessitats particulars d'aquest tipus de dispositius. Aquesta tecnologia consisteix en una màquina virtual i un conjunt d'API[4] adequades per a aquests dispositius.

(2)Java Micro Edition

(3)Java 2 Platform, Micro Edition

(4)interfície de programa d'aplicació o *application program interface*

Aquesta plataforma normalment produeix aplicacions portables, encara que algunes vegades hi ha biblioteques específiques de cada dispositiu (normalment usades per a jocs), que les fan no portables. Malgrat això, Java ME s'ha convertit en una bona opció per a crear aplicacions per a telèfons mòbils, a causa que es pot emular en un PC durant la fase de desenvolupament i després carregar-les fàcilment al mòbil. Encara que el procés no sigui directe, com que s'utilitzen tecnologies Java per al desenvolupament, resulta bastant econòmic portar-les a altres dispositius.

S'usa moltes vegades per a proporcionar aplicacions simples a telèfons mòbils de gamma baixa. Per tant, les aplicacions (incloent-hi les dades d'aquestes aplicacions) no poden ocupar massa memòria si s'han d'executar en la majoria d'aquests telèfons. A més, han d'estar criptogràficament signades per a poder usar API com la d'accés al sistema de fitxers. Això és relativament car i rarament es fa, fins i tot per a aplicacions comercials. Java ME s'executa sobre una màquina virtual que permet un accés raonable, però no complet, a les funcionalitats del dispositiu sobre el qual s'executa l'aplicació. El procés JSR[5] serveix per a incrementar gradualment la funcionalitat disponible per a Java ME, mentre proporciona als operadors i als fabricants la capacitat de prevenir o limitar l'accés al programari disponible.

(5)sol·licituds d'especificació per a Java o *Java specification requests*

A la fi de 2012, Oracle es va embarcar en un ambiciós projecte per a lliurar una important actualització de la plataforma Java ME, la versió Java ME 8. Amb aquesta versió Java ME 8 es va convertir en un referent per al desenvolupament de IoT (*Internet of Things*). Ideal per a usar-la en petits dispositius connectats a internet.

Malgrat això, Android va anar guanyant terreny a la plataforma J2ME fins a deixar-la pràcticament fora del mercat mòbil.

3.1.2. Symbian

Symbian té una història que comença temps enrere, el 1981. En la cronologia següent es pot veure l'evolució del sistema operatiu Symbian:

- **1981**. Psion llança el seu primer producte, el Flight Simulator.
- **1984**. Psion Organiser veu la llum.
- **1990**. SIBO SO (16 bits).
- **1997**. EPOC SO (32 bits).
- **1998**. El nom de Symbian apareix per primera vegada.
- **1999**. EPOC versió 5.
- **2000**. Symbian 6.0.
- **2001**. Symbian 6.1.
- **2003**. Symbian 7.0.
- **2004**. Symbian 8.0.
- **2005**. Symbian 9.0.
- **2008**. Nokia compra Symbian Ltd., l'empresa que hi ha darrere del Symbian OS.
- **2009**. Creació de la Symbian Foundation.
- **2010**. Es publica el codi font del Symbian amb llicència EPL[6].
- **2011**. Nokia du a terme una important aliança amb Microsoft i deixa de costat el sistema operatiu Symbian, que seria reemplaçat pel Windows Phone 7.

(6)llicència pública Eclipse o *Eclipse public license*

Symbian és un sistema operatiu fruit de l'aliança de diverses empreses de telefonia mòbil, entre les quals hi ha Psion, Nokia, Ericsson i Motorola, amb la intenció de desenvolupar i estandarditzar un sistema operatiu que permetés els telèfons mòbils de diferents fabricants intercanviar informació.

El Symbian OS va ser durant uns anys el sistema operatiu estàndard per als telèfons intel·ligents de l'època, ja que més del 85% dels fabricants d'aquests dispositius tenien llicències per a usar-lo. El Symbian OS estava dissenyat per als requisits específics dels telèfons mòbils 2.5G i 3G.

Dissenyada des de l'inici per a dispositius mòbils, la plataforma Symbian és un sistema operatiu de temps real, multitasca, específicament pensada per a funcionar bé en sistemes amb recursos limitats, maximitzant l'eficiència i la durada de la bateria, mentre que minimitza l'ús de memòria. La Symbian Foundation manté el codi per a la plataforma de programari lliure basada en el Symbian OS i aportacions de programari de Nokia, NTT DOCOMO i Sony Mobile, incloent-hi les interfícies d'usuari S60 i MOAP(S). La plataforma és totalment de codi obert, i la majoria es proporciona amb la llicència pública de l'Eclipse.

Popularitat del Symbian OS

S'han venut més de 300 milions d'unitats basades en el Symbian OS i durant anys ha gaudit de més del 50% de quota de mercat.

El sistema operatiu Symbian va incorporar el suport a pantalles tàctils gràcies a UIQ[7]. UIQ és una interfície gràfica d'usuari basada en l'ús d'un llapis que es pot trobar en telèfons 2.5G i 3G de les marques següents: Motorola, Sony Mobile, BenQ i ARIMA. Els telèfons UIQ utilitzen pantalles tàctils amb una resolució de 208-240 × 320 píxels i profunditat de color de 12, 16, 18 o 24 bits, depenent de la versió d'UIQ o del terminal. Les últimes versions d'UIQ van ser les 3.x.

(7) interfície d'usuari Quartz o *user interface Quartz*

3.1.3. Android

L'Android és una plataforma, basada en el Linux, de l'Open Handset Alliance, entre els 34 membres de la qual s'inclouen Google, HTC, Motorola, Qualcomm i T-Mobile. Per tant, 34 de les principals companyies de programari, maquinari i telecomunicacions donen suport a aquesta plataforma. El nucli Linux s'usa com a HAL[8] i a l'entorn d'execució d'aplicacions (ART) per a realitzar tasques com la generació de subprocessos i l'administració de memòria de baix nivell. La programació d'aplicacions es fa bàsicament en Java. És necessari l'SDK[9] específic de l'Android per a desenvolupar.

(8) capa d'abstracció de maquinari o *hardware abstraction layer*

(9) equip de desenvolupament de programari o *software development kit*

La majoria de dispositius Android disposen d'una combinació de codi obert i codi propietari. El nucli d'Android és de codi obert, però una gran quantitat de llibreries i aplicacions són de codi propietari de Google. És propietari, per exemple, tot el codi per accedir a serveis de Google com les notificacions, geolocalització, etc. Algunes aplicacions propietàries de Google són Gmail, Google Chrome, Google Maps i tot l'accés a la botiga d'aplicacions Google Play.

Algunes empreses com Cyanogen Inc van intentar proporcionar versions d'Android sense cap tipus de codi propietari de Google. Aquestes iniciatives no van tenir molt èxit, i Cyanogen Inc va tancar al 31 de desembre del 2016.

Les diferents versions del sistema han anat solucionant problemes de seguretat, rendiment, durada de la bateria, interfície gràfica, etc.

Com ja s'ha comentat anteriorment el gran problema d'Android és la fragmentació. Hi ha multitud de dispositius amb diferents capacitats de maquinari i diferents versions del sistema, en molts casos no es poden actualitzar a les versions noves.

Android és sens dubte la plataforma que compta amb major nombre de dispositius, usuaris, aplicacions i desenvolupadors. Es va començar utilitzant com a entorn de programació Eclipse, però al desembre de 2014 Google va llançar la primera versió d'Android Studio. L'Android Studio és l'entorn de desenvolupament integrat oficial per a la plataforma Android.

Si volem escriure un codi que necessiti alta eficiència podem fer servir Android NDK, un conjunt d'eines que permet implementar parts de les aplicacions usant llenguatges de codi natiu com C i C ++.

3.1.4. Windows Mobile

Des de l'any 2002 la plataforma Windows Mobile va ser disponible en una gran varietat de dispositius de diferents operadors sense fils. Es podia trobar el programari Windows Mobile en productes Dell, HP, Motorola, Palm i i-mate. Els dispositius amb el Windows Mobile eren disponibles per a xarxes GSM o CDMA.

El Windows Mobile és una variant del Windows CE per a telèfons mòbils. Originalment el Windows CE es va desenvolupar per a ordinadors de butxaca i PDA amb pantalles tàctils que funcionaven amb un llapis, i es va adaptar posteriorment per a usar-lo en telèfons intel·ligents equipats amb un teclat. Per tant, els telèfons es van convertir en la base més gran de dispositius instal·lats amb el CE, encara que la quota de mercat ha caigut dràsticament des de l'aparició de l'Android i l'iPhone. El Windows Mobile suporta un subconjunt de la interfície de programació de Win32 i una GUI[10] simplificada amb una finestra a la pantalla alhora.

[10] interfície gràfica d'usuari o *graphical user interface*

El 2010 la gran quantitat de dispositius industrials que contenien Windows Mobile va fer que Microsoft dividís el desenvolupament de les versions mòbils entre la continuació de Windows Mobile, coneguda com a Windows Embedded Handheld, i el desenvolupament d'un sistema operatiu mòbil per al gran públic, conegut com a Windows Phone.

Cap versió de Windows Phone va aconseguir fer-se amb una quota significativa de mercat, tot i que Microsoft va comprar Nokia i va equipar els nous dispositius del fabricant finlandès amb aquestes versions.

Després de Windows Phone 8.1, Microsoft va introduir Windows 10 Mobile. Era el seu sistema operatiu mòbil més innovador. Comptava amb característiques com *continuum*. Aquesta característica permetia a les aplicacions mòbils

convertir-se en versió escriptori quan es connectava a un monitor i un teclat. La transformació es realitzava en temps real sense haver de tornar a arrencar l'aplicació.

L'entorn de programació és Visual Studio. Es crea un nou tipus d'aplicacions denominat Universal (UWP), que en permet l'execució amb independència del tipus de dispositiu. El llenguatge de programació és C#, el *framework* .NET i es fa servir XAML (*eXtensible Application Markup Language*) per a definir la interfície gràfica que al final es materialitzarà en WPF (*Windows Presentation Foundation*) o Silverlight.

3.1.5. Qt *framework*

El Qt usa C++ estàndard però fa un ús extensiu d'un preprocessador especial, anomenat *MOC*[11], per a enriquir el llenguatge. També es pot usar el Qt en altres llenguatges de programació usant enllaços entre llenguatges. Funciona sobre les principals plataformes i té un suport internacional extens. Entre les característiques no relacionades amb la GUI s'inclouen accés a bases de dades SQL, tractament d'XML, gestió de fils, suport de xarxa i una API multiplataforma unificada per a la gestió de fitxers.

(11) *meta object compiler*

3.1.6. BREW

El BREW s'usa per a aplicacions en dispositius CDMA (però també suporta models GPRS/GSM). En general s'utilitza en dispositius de gamma baixa. Les aplicacions es distribueixen via una plataforma de contingut BREW i ha tingut poca penetració a Europa. El BREW pot proporcionar control complet del dispositiu i accedir a tota la funcionalitat que té. No obstant això, el potencial que proporciona el codi natiu amb accés directe a les API del dispositiu ha causat que el procés de desenvolupament en BREW hagi hagut de ser adaptat en gran manera per a tots els venedors de programari reconeguts. Mentre que l'SDK del BREW és disponible de manera lliure, executar programari en maquinari real d'un dispositiu mòbil (en contraposició amb l'emulador proporcionat) requereix una signatura digital que es pot generar amb eines publicades per diversos participants, essencialment proveïdors de contingut per a mòbils i Qualcomm. Fins i tot llavors, el programari només funcionarà en dispositius habilitats per a test. Per a baixar-lo en telèfons normals, el programari ha de ser comprovat i provat i ha de rebre l'aprovació de Qualcomm via el seu programa de verificació True BREW.

3.1.7. Palm OS

Des de l'aparició del primer Palm Pilot el 1996, la plataforma Palm US ha proporcionat als seus dispositius mòbils eines de negoci essencials, i també la capacitat d'accedir a Internet o a una base de dades central corporativa via una connexió sense fils.

El Palm OS va tenir una gran acceptació empresarial en l'important mercat dels Estats Units, basada en els PDA Palm.

El Palm webOS és el sistema operatiu mòbil de propietat evolució del Palm. Funciona sobre un nucli Linux que suporta multitasca. Es va llançar amb Palm Pre i Pixi, i ara és propietat de Hewlett Packard.

3.1.8. Flash Lite

S'usa en dispositius que suporten el reproductor Flash Lite.

3.1.9. Micronavegador

Els entorns basats en el concepte de *micronavegador* o *microbrowser* proporcionen una funcionalitat limitada via una interfície web.

3.2. Desenvolupament multiplataforma

En aquest subapartat descriurem entorns de treball que permeten desenvolupar aplicacions que funcionin tant en l'iPhone com en l'Android.

3.2.1. Titanium Mobile

El Titanium és un entorn de codi obert que permet desenvolupament multiplataforma. Es pot desenvolupar una aplicació que funcioni en dispositius mòbils (iOS, Android, RIM) o plataformes d'escriptoris (OS X, Windows).

El codi font de l'aplicació s'escriu tot en JavaScript, CSS i HTML5. Això és positiu en el sentit que no es necessita aprendre llenguatges complexos com Objective-C o C++.

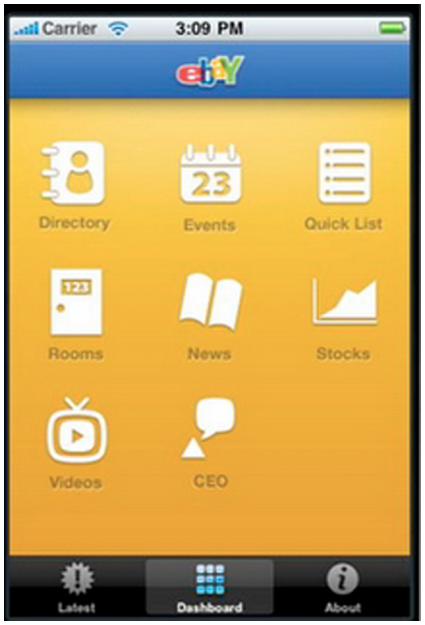
El Titanium és extensible: es pot estendre l'entorn afegint mòduls propis en Objective-C o Java per al cas de l'Android.

Nota

Hi ha una gran quantitat de documentació per al Titanium.

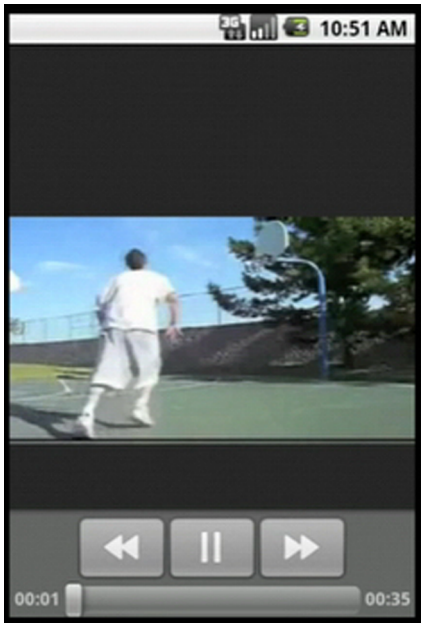
Amb el Titanium un desenvolupador es pot beneficiar de l'ús del següent:

1) Interfícies natives



Interfície nativa

2) Aplicacions multimèdia



Aplicació multimèdia

3) Entorn mòbil i d'escriptori

4) Llenguatge JavaScript

Tot això és possible gràcies al fet que el Titanium té un pont que tradueix el codi JavaScript al codi equivalent Objective-C o Java en temps d'execució.

Titanium és la part de codi obert de The Appcelerator Platform. Aquesta plataforma ofereix serveis de pagament complets que inclouen: proves, accés a dades, control de rendiment, mètriques d'ús de les funcions de l'aplicació, etc.

3.2.2. PhoneGap

El PhoneGap permet desenvolupar aplicacions per a l'Android usant tecnologies web com HTML, CSS i JavaScript, i pot convertir aquestes aplicacions web en aplicacions natives d'Android. De fet, el PhoneGap suporta múltiples plataformes com Android, iPhone, Palm, Windows Mobile i Symbian, de manera que es pot usar el mateix codi font per a crear aplicacions per a múltiples plataformes. Malgrat que es venguin com a «eines de tecnologia web», el que ofereixen el PhoneGap o altres entorns de treball similars com el Titanium és accés al maquinari de la màquina: es poden fer aplicacions en HTML i JavaScript que usin la càmera, la brúixola o l'acceleròmetre. El PhoneGap és, a més, lliure amb llicència MIT[12].

(12)Institut Tecnològic de Massachusetts (Massachusetts Institute of Technology)

jQuery i jQTouch

El PhoneGap permet desenvolupar aplicacions amb biblioteques de JavaScript com jQuery/jQTouch.

Per tant, el PhoneGap és una solució de codi obert dissenyada per a donar accés JavaScript als desenvolupadors web a característiques populars dels dispositius mòbils com la càmera, el GPS, l'acceleròmetre o bases de dades SQLite locals sense la necessitat d'haver d'escriure aplicacions completes. La idea de fons és fer fàcil el desenvolupament d'aplicacions mòbils.

Per a aconseguir això, l'entorn PhoneGap actua com un pont entre les aplicacions web i els dispositius mòbils. Permet als desenvolupadors envoltar aplicacions web dins d'una aplicació nativa, i així fer el desenvolupament més fàcil per a aquells que no estan familiaritzats amb Objective-C i Cocoa.

El 2011 Adobe va comprar Nitobi, l'empresa propietària de PhoneGap. A partir d'aquest moment i per a evitar problemes legals, la part de codi obert de PhoneGap es va anomenar Cordova.

Adobe ha convertit PhoneGap en un sistema de generació i publicació d'aplicacions en les diferents botigues basat en el seu núvol. Això fa que no s'hagin de tenir instal·lats els entorns de programació de les diferents plataformes: només ens hem de preocupar del codi JavaScript i HTML. Ens proporciona una forma senzilla de provar aquest codi en els dispositius mòbils fent servir una aplicació PhoneGap que es pot descarregar de les diferents botigues (Google Play i Apple Store). Tot i que es manté un model de subscripció gratuït, té moltes limitacions: una sola aplicació, la mida de la qual ha de ser molt petita, etc.

Si volem utilitzar Cordova directament necessitem tenir instal·lat l'entorn natiu (Xcode o Android Studio), baixar-nos el seu codi i compilar-ho en un projecte natiu. Després, en una carpeta del nostre projecte ja podem afegir la part HTML i JavaScript.

3.2.3. Ionic *framework*

El 2013 va aparèixer el *framework* Ionic. Ionic és un sistema molt similar a PhoneGap, que també permet la creació d'aplicacions híbrides basades en tecnologies web.

La diferència entre PhoneGap i Ionic és que mentre a PhoneGap s'usa qualsevol tecnologia web (JQuery, Bootstrap, etc.), a Ionic hi ha una relació més profunda entre la part nativa i el *framework* JavaScript Angular JS. Fent servir aquest *framework*, Ionic assegura un rendiment més gran en el codi basat en JavaScript.

3.2.4. Xamarin

Xamarin és un entorn multiplataforma que permet la generació d'aplicacions natives fent servir la implementació de codi lliure de .NET (Mono) i el llenguatge de programació C#. El 2016 Microsoft va comprar la companyia. Aquesta compra va augmentar encara més la seva integració amb l'entorn de desenvolupament de Microsoft (Visual Studio).

Xamarin genera interfícies basades en components nadius, amb el que el seu rendiment és molt més gran que les aplicacions basades en la encapsulació d'HTML i JavaScript. També permet que C# faci servir de manera senzilla codi que ja tinguem desenvolupat en Java o Objective-C.

3.2.5. Unity

Unity és un motor de videojocs multiplataforma que ha tingut una gran acceptació de forma molt ràpida. En l'apartat mòbil suporta versions per a Android i iOS utilitzant en ambdós casos OpenGL ES. Des de la versió 5.6 suporta les APIs de baix nivell Metal a iOS i Vulcan a Android. Com a llenguatges de programació fa servir C# i UnityScript, un llenguatge molt similar a JavaScript. Ofereix un repositori de components molt ben gestionat que s'anomena Unity Asset Store. Genera aplicacions amb rendiment natiu. El problema d'aquest tipus de sistemes és que és el proveïdor, en aquest cas Unity, el que ha de proporcionar l'accés a les noves capacitats que introdueix el sistema operatiu. La seva versió coneguda com Personal és gratuïta però només es pot utilitzar si es tenen ingressos inferiors a cent mil dòlars.

3.2.6. Adobe Flash Builder

Adobe Flash Builder permet generar aplicacions multiplataforma (iOS i Android) basades en l'Adobe Flash Player. Com a entorn de programació fa servir Eclipse i com a llenguatge Action Script (similar a JavaScript). El rendiment

de les aplicacions depèn de les optimitzacions que realitza Adobe de l'Adobe Flash Player. S'han detectat problemes de rendiment i de seguretat en iOS i Android. És per aquest motiu que Unity s'ha acabat imposant.

3.3. Entorns per a dispositius d'un venedor únic

Les plataformes de programari següents només funcionen en plataformes de maquinari d'un fabricant específic:

1) BlackBerry. BlackBerry té suport per a correu electrònic, telèfon mòbil, missatges de text, tramesa de faxos, navegació per Internet i altres serveis d'informació sense fils, i també una interfície tàctil. Alguns dispositius BlackBerry disposen de sèrie d'un teclat QWERTY optimitzat per a utilitzar-lo teclejant amb els polzes. Quan van aparèixer, els dispositius BlackBerry van agafar aviat una posició dominant al mercat nord-americà dels telèfons intel·ligents.

2) iOS d'Apple. L'SDK per a l'iPhone, l'iPad i l'iPod usa Swift i Objective-C, que està basat en el llenguatge de programació C. Swift és un llenguatge pensat per al desenvolupament d'aplicacions presentat per Apple el 2014. En el seu moment l'SDK només estava disponible en Mac OS 10.5+ i era l'única manera d'escriure una aplicació per a l'iPhone o l'iPad. A més, Apple ha de verificar totes les aplicacions abans d'allotjar-les en l'App Store, l'únic canal de distribució per a les aplicacions per a iOS.

4. Llenguatges de programació

Com s'ha vist prèviament, hi ha tres classes principals d'aplicacions per a dispositius mòbils: aplicacions natives, aplicacions web i aplicacions híbrides. Un quart cas de paradigma és el marcat per Java. Per a aquest cas, molts dels nous mòbils suporten alguna versió d'MIDP[13] i el desenvolupament en aquest entorn és bastant senzill. Per a la instal·lació, no obstant això, és una mica més complicat. En general s'instal·len aplicacions per mitjà d'enllaços a Internet, però alguns operadors o fabricants posen límits en les aplicacions que es poden instal·lar al mòbil.

(13) Mobile Information Device Profile

Per a aplicacions natives, depèn del sistema operatiu del mòbil. Per a molts, desenvolupar aplicacions natives pot costar diners (per a eines i SDK) i també hi ha problemes en la manera com es distribueixen les aplicacions. La instal·lació i la depuració d'errors són també diferents depenent del sistema operatiu.

Per tant, el llenguatge de programació que s'usi serà probablement dictat pel dispositiu i la plataforma per a la qual es vol desenvolupar una aplicació, a més de l'aplicació que es vol crear.

A continuació s'enumeren els diferents llenguatges amb què es poden desenvolupar aplicacions natives per a diferents plataformes:

- Si es vol fer una aplicació per iOS, s'usarà **Swift** o **Objective-C**.
- Si es vol fer una aplicació Android, s'usarà **Java o C i C++** si necessitem accés al maquinari.
- Si es vol fer una aplicació per a BlackBerry, s'usarà **Java Micro Edition**.
- Si es vol fer una aplicació per al Symbian OS, es pot usar **C++**, **Java** o **.NET Compact Framework**.
- Si es vol fer una aplicació per al Windows Mobile, les opcions són **Visual C++**, **Visual C#**, **XAML**, **Visual Basic**, **JScript** i **ASP.NET**.

Per tant, la plataforma o el dispositiu dictaran quins llenguatges de programació es tenen com a opcions. Si es vol desenvolupar per a una plataforma que permet tant C++ com Java, llavors el tipus d'aplicació que es planeja desenvolupar podria dictar quin llenguatge és la millor opció.

A continuació es mostra una compilació dels llenguatges de programació més populars per a dispositius mòbils.

4.1. Llenguatges de programació per al Windows Mobile

Es pot escollir entre diverses opcions de llenguatges de programació a l'hora de desenvolupar aplicacions per a dispositius amb el Windows Mobile. En aquest subapartat veurem aquestes opcions i farem una breu descripció de cadascuna.

4.1.1. Visual C++

Es coneix C++ com un llenguatge de desenvolupament «natiu», a causa que interactua directament amb el maquinari d'un dispositiu Windows Mobile, sense la intervenció de cap altra capa (al contrari que Visual C#, per exemple). Programar usant C++ pot ser un desafiament, ja que no és un llenguatge trivial d'aprendre. Alguns errors en un programa C++ poden bloquejar potencialment tot el dispositiu.

Errors en un programa C++

Accedir a memòria que ha estat alliberada o oblidar-se d'alliberar memòria en un programa C++ pot bloquejar un dispositiu Windows Mobile.

Els avantatges d'usar Visual C++ són la velocitat d'execució, la mida de l'aplicació i la flexibilitat. Les aplicacions escrites en C++ s'executen molt ràpid i consumeixen els recursos mínims.

Jocs d'acció

Els jocs d'acció en temps real són bons exemples de programes que es beneficien de la velocitat d'execució de C++.

Lectura recomanada

Una bona manera d'aprendre a fer servir les eines Microsoft és investigar el Visual Studio Community. És la versió gratuïta de l'entorn de desenvolupament de Microsoft. Es poden desenvolupar solucions per a tot tipus de dispositius.

Les aplicacions Visual C++ poden interaccionar amb el dispositiu Windows Mobile utilitzant les crides del SDK de Windows 10.

Si volem fer servir capacitats específiques d'una plataforma a la nostra aplicació hem d'afegir el seu SDK. Al canal Channel9 de la MSDN hi ha molta informació per a aprendre a fer aplicacions universals.

Si es té experiència desenvolupant per al Windows usant Visual C++, la transició al Windows Mobile no és especialment complicada. Caldria aprendre a instal·lar i usar les eines específiques, i després aprendre a utilitzar les característiques específiques del dispositiu, cosa que permetrà explotar les capacitats dels dispositius.

Per a començar una aplicació Visual C++, cal arrencar el Visual Studio i seleccionar *File* > *New* > *Project* > *Installed* > *Templates* > *Visual C++* > *Windows* > *Universal*.

Si som nous tant programant com amb el Windows Mobile, seria una bona idea començar amb Visual C# i llavors fer la transició a Visual C++.

4.1.2. Visual C# i Visual Basic

Visual C# i Visual Basic .Net són llenguatges de desenvolupament més senzills que Visual C++. No solament són relativament fàcils d'aprendre, sinó que a més tenen suport per al .NET.

.NET

El .NET és una biblioteca de classes que fan gran quantitat de tasques usades freqüentment en programació, per simplificar el desenvolupament d'aplicacions.

Les eines de desenvolupament per a C# i Visual Basic .NET inclouen un dissenyador complet d'interfície d'usuari WYSIWYG[14]. Es poden arrossegar i col·locar botons i altres controls directament a la finestra de l'aplicació, i llavors fer doble clic per accedir al codi que hi ha per sota. Aquest sistema fa que crear interfícies d'usuari per a les aplicacions sigui extremament ràpid i fàcil.

(14)«What you see is what you get».

La interfície gràfica creada amb el dissenyador es materialitza en un llenguatge conegut com a XAML (eXtensible Application Markup Language). Quan aquest llenguatge s'interpreta genera classes pertanyents a les Windows Presentation Foundation (WPF) i Silverlight.

Com a part de la biblioteca .NET, hi ha disponibles classes extra que cobreixen des d'estructures de dades fins a intercepció de missatges de text. Per a fer ús de les característiques específiques del Windows Mobile, es proporcionen un conjunt de classes extra. Aquestes classes proporcionen accés a les característiques del dispositiu, com per exemple la llista de contactes o la càmera.

Si es té experiència desenvolupant aplicacions per a Windows usant Visual C#, la transició hauria de ser relativament senzilla.

Visual C# és una bona manera d'aprendre programació. Per a aprendre tot el necessari tant de Visual C# com de Visual Basic, es pot acudir a l'MSDN de Microsoft.

Per a començar una aplicació Visual C# o Visual Basic .NET, cal arrencar el Visual Studio i seleccionar *File* > *New* > *Project* > *Installed* > *Templates* > *Visual C#* > *Windows* > *Universal*.

4.1.3. JScript

El navegador web inclòs amb els dispositius Windows Mobile –Internet Explorer Mobile– suporta JScript. JScript és un superconjunt del llenguatge conegut com a *JavaScript*. Els programes JScript són fitxers de text pla que executa el navegador web. Poden estar incrustats en una pàgina HTML o emmagatzemats en fitxers separats.

Una aplicació JScript s'executa dins del navegador web i usa la finestra del navegador web per a l'entrada/sortida d'informació. És possible fer ús de tècniques de programació AJAX[15] per a proporcionar un grau d'interacció amb l'usuari i comunicar-se amb un servidor remot. A causa de la naturalesa de JScript, les aplicacions no poden accedir a dades locals que no siguin simplement galetes, la qual cosa introdueix algunes limitacions.

[15] JavaScript asíncron i XML o *asynchronous JavaScript and XML*

Eina de desenvolupament

No es necessita cap eina de desenvolupament especial: un editor de text és suficient per a crear una aplicació JScript. El programa es pot desar de manera local o s'hi pot accedir des d'un servidor web.

4.1.4. ASP.NET

Mentre que JScript és una solució del costat del client per a programar aplicacions d'Internet, ASP.NET és una solució del costat del servidor. Amb ASP.NET es poden escriure aplicacions en C# o Visual Basic .NET que siguin en un servidor web i facin tasques complexes, incloent-hi crear controls d'interfície d'usuari i accedir a bases de dades. ASP.NET aïlla les característiques del dispositiu de l'aplicació, i fa senzill executar una aplicació en diversos tipus de dispositius diferents. Com a alternativa al ASP.NET tradicional basat en formularis Microsoft permet crear aplicacions web ASP.NET usant el paradigma MVC. Aquest paradigma genera aplicacions web més eficients i permet independitzar de forma més clara la presentació de les dades.

5. Exemples d'entorns

En aquest apartat es descriuran de manera més detallada dos entorns: iPhone/iOS i Android.

5.1. iPhone/iOS

L'iOS comprèn el sistema operatiu i les tecnologies que s'usen per a executar aplicacions de manera natiua en dispositius com l'iPad, l'iPhone i l'iPod Touch. Encara que comparteix una herència comuna i moltes tecnologies de base amb el Mac OS X, l'iOS es va dissenyar per a satisfer les necessitats d'un entorn mòbil, en què les necessitats dels usuaris són lleugerament diferents. Si s'han desenvolupat prèviament aplicacions per a Mac OS X, es trobaran moltes tecnologies familiars, però també tecnologies que només són disponibles en l'iOS, com el suport per a interfície tàctil o acceleròmetre.

L'SDK de l'iOS conté el codi, la informació i les eines necessàries per a desenvolupar, verificar, executar, depurar errors i adaptar aplicacions per a l'iOS. Les eines Xcode proporcionen l'entorn bàsic per a editar, compilar i depurar errors en el codi. Xcode també proporciona el punt de llançament per a verificar les aplicacions en un dispositiu iOS i en un simulador iOS –una plataforma que imita l'entorn bàsic iOS però s'executa en un ordinador Macintosh local.

Aquest subapartat proporciona una descripció d'alt nivell de les característiques bàsiques que es poden trobar en l'iOS per a orientar sobre aquesta plataforma.

5.1.1. Visió general del sistema iOS

L'iOS (conegut com a *iPhone OS* abans del 2010) és el sistema operatiu per a dispositius mòbil d'Apple. Originalment desenvolupat per a l'iPhone, s'ha anat estenent per a donar suport a altres dispositius Apple com l'iPod Touch, l'iPad i Apple TV. Apple no dóna llicències per a la instal·lació de iOS en maquinari de terceres parts.

El 2015 el tràfic provinent des de dispositius iOS als Estats Units superava el generat des de dispositius Android. A principis del 2017 es va arribar als 2.200.000 milions d'aplicacions a l'Apple Store. En l'últim trimestre del 2016 es van arribar als 140 mil milions d'aplicacions descarregades.

La interfície d'usuari de l'iOS es basa en el concepte de *manipulació directa*, usant gestos multicontacte. Els elements de control de la interfície consisteixen en lliscadors, interruptors i botons. La resposta a les peticions de l'usuari és immediata i proporciona una interfície fluida. La interacció amb l'iOS inclou gestos com *tocar fort*, *tocar de manera més feble*, *subjectar* i *deixar anar*, els quals tenen definicions específiques en el context del sistema operatiu iOS i la interfície multicontacte d'aquest sistema. Des de les versions iPhone 6s i iPhone 6s Plus Apple incorpora la tecnologia 3D Touch. Aquesta tecnologia permet obtenir amb precisió el nivell de pressió sobre la pantalla. Ha generat nous esdeveniments i ha dotat la interfície de noves funcions. També ha millorat la resposta hàptica de tal manera que els usuaris senten una vibració a la zona de la pressió.

Algunes aplicacions usen els acceleròmetres interns per a respondre quan se sacseja el dispositiu (un resultat comú és l'ordre *desfer*) o fer girar en tres dimensions (un resultat comú és canviar d'orientació vertical a horitzontal i viceversa).

L'iOS està derivat del Mac OS X, amb el qual comparteix la fundació Darwin, i és, per tant, un sistema operatiu semblant a l'Unix per naturalesa.

En l'iOS hi ha quatre capes d'abstracció:

- La capa Core OS.
- La capa Core Services.
- La capa Media.
- La capa Cocoa Touch.

Espai de l'iOS

El sistema operatiu iOS ha fet servir al llarg de la seva història entre 500 MB de les seves primeres versions, 4,5 GB de la versió 9 i 1.3 GB de la versió 10.

5.1.2. Història del sistema iOS

El sistema operatiu va aparèixer amb l'iPhone en la Macworld Conference & Expo el gener deL 2007, i es va llançar el juny d'aquell any. Al principi, els missatges de màrqueting d'Apple no especificaven un nom diferent per al sistema operatiu, i deien simplement que «l'iPhone executa l'OS X». Inicialment, no suportava aplicacions de terceres parts. Steve Jobs argumentava que els desenvolupadors podien programar aplicacions que «es comportarien com a aplicacions natives en l'iPhone». L'octubre del 2007, Apple va anunciar que s'estava desenvolupant un SDK natiu i que planejaven posar-lo «en mans dels desenvolupadors al febrer». El març del 2008 Apple va llançar la primera versió beta juntament amb un nou nom per al sistema operatiu: iPhone OS.

Les grans vendes dels dispositius mòbils d'Apple van encendre l'interès en l'SDK. El setembre previ, Apple havia llançat l'iPod Touch, que tenia la majoria de les capacitats de l'iPhone no relacionades amb la telefonia. Apple, a

més, va vendre més d'un milió d'iPhones durant les vacances del 2007. El gener del 2010, Apple va anunciar l'iPad, dispositiu amb una pantalla més gran que l'iPhone i l'iPod Touch, i dissenyat per a navegar per Internet, continguts multimèdia i lectura de llibres electrònics.

El juny del 2010, Apple va rebatejar l'iPhone OS com a *iOS*. El nom *IOS*[16] havia estat usat per Cisco durant una dècada per al seu IOS, usat en encaminadors Cisco. Per a evitar plets potencials, Apple va pagar la llicència per a usar la marca IOS de Cisco.

(16)sistema operatiu d'interconnexió de xarxes o *internetwork operating system*

5.1.3. Història de les versions del sistema iOS

- iOS 4-4.3.5 (2010): Apple va anunciar que l'iPhone OS s'havia rebatejat com a *iOS*. L'iOS 4 va ser la primera versió del sistema operatiu, que era una actualització gratuïta. Com a novetat afegia la capacitat de multitasca, però amb molt de control sobre les capacitats de les aplicacions en segon pla perquè no consumeixin molts recursos (CPU, bateria, etc.). Incloïa una solució per a l'indicador de la intensitat del senyal rebut. Utilitzava carpetes per a organitzar les aplicacions. Va afegir l'aplicació iBooks per a la consulta de documents en local i el Game Center, un conjunt de llibreries que els programadors de videojocs podien fer servir per a tenir un control centralitzat d'usuaris, puntuació, i facilitava la creació de jocs multijugador. Va afegir suport per a AirPlay i AirPrint. Va millorar el navegador Safari amb el motor de JavaScript Nitro.

- iOS 5-5.1.1 (2011): S'afegeix el centre de notificacions i accés a les notificacions des de la pantalla de bloqueig. Suport per a gestors de multitasca en iPad. S'afegeix un assistent intel·ligent anomenat Siri, al qual es van incorporant nous idiomes.

- iOS 6-6.1.5 (2012): Apareix el mode «no molestar». S'incorpora suport per al sistema de videoconferència FaceTime en xarxes 3G i 4G. Apple va presentar la seva pròpia aplicació de mapes, substituint Google Maps. S'afegeixen nous controls per a protegir la privacitat. S'incorporen noves funcionalitats a Siri.

- iOS 7-7.1 (2013): Es modifica la interfície d'usuari. S'incorporen millores en el servei de localització d'iPhone, que inclou el bloqueig del dispositiu. Apareix per primera vegada el reconeixement d'empremta dactilar a l'iPhone 5S. Inclou AirDrop, el servei per a intercanviar fitxers entre dispositius iOS.

- iOS 8- 8.4.1 (2014): Es millora el control de la interfície al centre de notificacions. S'afegeix l'opció de teclats de terceres companyies. Van aparèixer les llibreries HealthKit per a control d'exercici físic i constants vitals, i Homekit per a domòtica (gestió d'il·luminació, aire condicionat, alarmes, calefacció, etc.). S'incorpora suport per a Apple Pay, el sistema de paga-

ment basat en NFC d'Apple. Apple va presentar Apple Watch, el seu primer *smartwatch*.

- iOS 9-9.3 (2015): Es dona molt més protagonisme a Siri. S'afegeixen modes de baix consum per a allargar la vida de la bateria i el mode nocturn. S'afegeix el suport per a iPhone 6S sèries i iPad Pro.

- iOS 10 (2016): S'afegeix molt més control hàptic en molts elements d'interfície. Es proporciona accés a terceres companyies a l'SDK de Siri. Aquest accés està limitat a empreses d'alguns dominis d'activitat com missatgeria, recerca de fotos, etc. Es milloren aspectes relacionats amb la privacitat i Apple Pay. Els desenvolupadors poden accedir al sistema de reconeixement de veu. Incorpora canvis a 3D Touch, nous components per a la pantalla de bloqueig i el centre de notificacions.

5.1.4. Característiques del sistema iOS

Sempre que s'encén el dispositiu o es pressiona el botó d'inici, es presenta la pantalla principal amb icones d'aplicacions i un receptacle en la part inferior on els usuaris poden col·locar les aplicacions usades amb més freqüència. La pantalla té una barra d'estat al llarg de la part superior per a mostrar dades com l'hora, el nivell de bateria o la potència de senyal. La resta de la pantalla es dedica a l'aplicació actual.

Amb l'iOS 4 va arribar la introducció d'un sistema de carpetes simple. Es pot arrossegar qualsevol aplicació i deixar-la anar a sobre d'una altra per crear una carpeta. A partir d'aquell moment, es poden afegir més aplicacions a la carpeta usant el mateix procés. Se selecciona un títol per a la carpeta de manera automàtica segons el tipus d'aplicacions que hi ha a dins, però l'usuari en pot editar també el nom.

La pantalla d'inici de l'iOS conté aplicacions per defecte. Algunes d'aquestes aplicacions no són visibles per defecte i l'usuari hi pot accedir mitjançant l'aplicació *Settings* o un altre mètode.

Totes les utilitats com notes de veu, calculadora o brúixola són en una carpeta anomenada *Utilidades*.

Compartició de dades

En el sistema iOS es pot seleccionar un número de telèfon d'un correu electrònic i desar-lo com un contacte o marcar-lo per fer una trucada

L'iPad i l'iPod Touch mantenen les mateixes aplicacions que són presents per defecte en l'iPhone, amb l'excepció de les aplicacions relacionades directament amb connectivitat amb xarxes.

A l'iPad la majoria de les aplicacions estan completament refetes per a beneficiar-se de la pantalla més gran. La configuració per defecte del receptacle inferior inclou *Safari*, *Mail*, *Fotos* i *Música*.

Multitasca

Abans de l'iOS 4, la multitasca era limitada a una selecció de les aplicacions que Apple incloïa en els dispositius. A Apple li preocupava que executar múltiples aplicacions de terceres parts de manera simultània descarregués la bateria massa ràpid. A partir de l'iOS 4, en dispositius iOS de tercera generació en endavant, hi ha suport per a multitasca per mitjà de 7 API:

- Àudio en segon pla.
- Veu sobre IP.
- Localització en segon pla.
- Notificacions *push*.
- Notificacions locals.
- Acabament de tasques.
- Canvi ràpid d'aplicació.

Pressionar dues vegades el botó d'inici activa l'intercanviador d'aplicació. Aquest mecanisme permet canviar d'aplicació i finalitzar les mateixes.

Game Center

El Game Center és una «xarxa social de joc» multijugador en línia llançada per Apple. Permet als usuaris «convidar amics a jugar a un joc, començar un joc multijugador, controlar els assoliments i comparar les puntuacions més altes en un tauler de líders».

El Game Center es va anunciar durant una presentació de l'iOS 4 l'abril del 2010. Es va llançar una versió prèvia per als desenvolupadors registrats d'Apple a l'agost. Finalment es va llançar el setembre del 2010 amb l'iOS 4.1 en l'iPhone, l'iPhone 3GS i l'iPod Touch de la segona a la quarta generació. El Game Center va fer el debut públic en l'iPad amb l'iOS 4.2.1. A la versió iOS 10, el Game Center va desaparèixer com a aplicació independent. Apple va eliminar moltes funcions socials perquè preferia que els usuaris poguessin compartir els seus èxits amb usuaris d'altres sistemes usant plataformes com Facebook.

5.1.5. Desenvolupament d'aplicacions per a l'iOS

Les aplicacions han d'estar escrites i compilades específicament per a l'iOS i l'arquitectura ARM. El navegador web Safari suporta aplicacions web com altres navegadors. Hi ha disponibles aplicacions natives autoritzades de terceres parts, per a dispositius amb l'iOS 2.0 o posterior, per mitjà de l'App Store de Apple.

SDK

L'octubre del 2007, en una carta oberta, Steve Jobs va anunciar que el febrer del 2008 es posaria a disposició de desenvolupadors externs a Apple un SDK. L'SDK es va llançar el març del 2008 i permet als desenvolupadors fer aplica-

cions per a l'iPhone i l'iPod Touch, i també verificar-les en un «simulador iPhone». No obstant això, carregar una aplicació als dispositius és només possible després de pagar l'iPhone *developer program*. Des del llançament de l'Xcode 3.1, l'Xcode és l'entorn de desenvolupament per a l'iOS SDK. Les aplicacions iPhone, com l'iOS i el Mac OS X, estan escrites en Swift o Objective-C. Swift es va presentar per Apple el 2014. És molt més intuïtiu que Objective-C i té una corba d'aprenentatge molt ràpida. Swift va tenir una acceptació molt ràpida per part de la comunitat de desenvolupadors. Amb Swift 3 la sintaxi es va estabilitzar i es va optar per una sintaxi molt més concisa i elegant que la d'Objective-C. Swift s'executa molt ràpid i es pot comunicar fàcilment amb codi Objective-C. Realitza comprovació de tipus fort, encara que la seva declaració no sempre és necessària gràcies a la seva capacitat d'inferir tipus.

Xcode ha anat afegint funcions en cada versió. El Xcode 8 (2016) proporciona suport per a gran varietat de proves, incloent-hi les proves d'interfície. El dissenyador d'interfícies gràfiques és molt potent i inclou un complet sistema de restriccions per a permetre que les interfícies s'ajustin a les diferents mides de pantalla.

Els desenvolupadors poden posar qualsevol preu a les seves aplicacions per sobre d'un mínim perquè es distribueixin per mitjà de l'App Store, del qual rebran el 70% per cada venda. De manera alternativa, poden optar per llançar la seva aplicació de manera gratuïta i no pagar així cap cost per llançar o distribuir l'aplicació excepte el cost de membre.

5.1.6. *Jailbreaking*

L'iOS ha estat subjecte a una varietat de diferents manipulacions centrades a afegir funcionalitat sense el suport d'Apple. Abans del debut de l'App Store el 2008, la raó principal per al procés d'escapar-se de la presó o *jailbreaking* (o *jailbreak*) era instal·lar aplicacions natives de terceres parts. Apple va dir que no dissenyaria actualitzacions de programari específicament perquè aquestes aplicacions deixessin de funcionar (sempre que no fossin aplicacions que fessin desbloqueig de SIM[17]), però el cas és que amb cada actualització de l'iOS el *jailbreak* semblava deixar de funcionar.

[17] mòdul d'identificació de subscriptor o *subscriber identity module*

Des de l'arribada de l'App Store i les aplicacions de terceres parts, l'objectiu de la comunitat de *jailbreaking* ha canviat. El principal objectiu del *jailbreaking* és permetre la personalització del dispositiu, usar emuladors i millores que fa la comunitat com multitasca, l'Adobe Flash Player o accedir al sistema de fitxers de l'iPhone. La multitasca només és disponible en dispositius iOS de tercera generació en endavant, i les aplicacions en l'App Store no tenen permès modificar l'aparença del sistema operatiu.

Alguns *jailbreakers* també intenten compartir de manera il·legal aplicacions de pagament de l'App Store. Aquest objectiu ha causat alguna distensió dins de la comunitat de *jailbreaking*, a causa que no era l'objectiu original del *jailbreaking* i és il·legal. Hi ha també alguns usuaris que s'oposen a la censura de continguts d'Apple.

5.1.7. Gestió de drets digitals

La naturalesa tancada i propietària de l'iOS ha generat crítiques, particularment d'advocats de drets digitals com l'Electronic Frontier Foundation, l'enginyer informàtic i activista Brewster Kahle, l'especialista en lleis d'Internet Jonathan Zittraion i la Free Software Foundation, que va protestar en l'esdeveniment de presentació de l'iPad i ha fet de l'iPad el seu objectiu amb la seva campanya «Defective by Design». El competidor Microsoft també ha criticat el control d'Apple sobre la seva plataforma.

En el conflicte hi ha les restriccions imposades pel disseny de l'iOS, conegudes com a *DRM*[18], destinades a bloquejar els continguts que es compren a la plataforma Apple, el model de desenvolupament (que requereix una subscripció anual per a distribuir aplicacions desenvolupades per a l'iOS), el procés d'aprovació d'aplicacions centralitzat, i també el control general d'Apple sobre la plataforma per si mateixa. Particularment en disputa hi ha la capacitat d'Apple de deshabilitar o esborrar aplicacions de manera remota.

(18) gestió de drets digitals o *digital rights management*

Algunes veus dins de la comunitat tecnològica han expressat preocupació pel fet que l'iOS tancat representi una tendència cada vegada més important envers la visió d'Apple de la informàtica i fan notar el potencial d'aquestes restriccions per a reduir la innovació en programari.

No obstant això, també hi ha veus fora d'Apple que han mostrat el seu suport al model tancat de l'iOS. El desenvolupador de Facebook Joe Hewitt, que va protestar contra el control d'Apple sobre el seu maquinari com un «precedent horrible», ha argumentat després que les aplicacions tancades en l'iPad estan relacionades amb les aplicacions web i proporcionen més seguretat.

5.2. Android

L'Android és una pila de programari de codi obert per a dispositius mòbils que inclou sistema operatiu, programari intermedi i aplicacions bàsiques. Google Inc. va comprar l'empresa desenvolupadora inicial del programari, Android Inc., el 2005. El sistema operatiu de l'Android està basat en una versió modificada del nucli Linux. Google i altres membres de l'Open Handset Alliance col·laboren en el desenvolupament i llançament de l'Android. L'AOSP[19] està encarregat del manteniment i desenvolupament de l'Android.

(19) Android Open Source Project

En el quart trimestre del 2010, el sistema operatiu Android va ser la plataforma de telèfon intel·ligent més venuda del món, i va destronar el Symbian de Nokia de la primera posició per primera vegada en deu anys. Altres fonts indiquen que el Symbian estava encara lleugerament per davant en vendes si es tenien en compte alguns telèfons de models antics Symbian no-Nokia.

L'Android té una gran comunitat de desenvolupadors programant *apps*[20] que estenen la funcionalitat dels dispositius. El 2010 hi havia prop de 200.000 aplicacions i a principis del 2017 es va arribar als 2.800.000 milions d'aplicacions disponibles per a l'Android. Google Play és la botiga en línia gestionada per Google per mitjà de la qual també es poden baixar aplicacions de llocs de terceres parts. Els desenvolupadors programen principalment en el llenguatge Java, controlant el dispositiu mitjançant biblioteques Java desenvolupades per Google.

(20) programes d'aplicació o *application programs*

Llançament de l'Android

L'arribada de la distribució Android el novembre del 2007 es va anunciar amb la fundació de l'Open Handset Alliance, un consorci de setanta-nou companyies de maquinari, programari i telecomunicacions amb l'objectiu de desenvolupar estàndards oberts per a dispositius mòbils. Google va llançar la majoria del codi Android amb la llicència Apache, una llicència de programari lliure i codi obert.

La pila de programari del sistema operatiu Android consisteix en aplicacions Java que s'executen en un entorn d'aplicacions basat en Java i orientat a objectes a sobre de biblioteques base Java que s'executen en una màquina virtual Dalvik que fa compilació JIT[21] abans de la versió Lollipop. Des de la versió Lollipop s'usa ART (Android Runtime) en el moment de la instal·lació de l'aplicació per a transformar el *bytecode* Dalvik en codi binari depenent de l'arquitectura. També hi ha biblioteques escrites en C, que inclou el gestor de superfície, OpenCore Media Framework, el sistema gestor de base de dades relacional SQLite, l'API gràfica 3D OpenGL ES 3.0, el WebKit *layout engine*, el motor gràfic SGL[22], SSL, i Bionic *libc*.

(21) just a temps o *just in time*

(22) Scene Graph Library

Vegem algunes mètriques relacionades amb les primeres versions d'Android: el sistema operatiu Android consistia en 12 milions de línies de codi, incloent-hi 3 milions de línies d'XML, 2,8 milions de línies de C, 2,1 milions de línies de Java i 1,75 milions de línies de C++.

5.2.1. Història de l'Android

L'octubre del 2003, Andy Rubin, Rich Miner i d'altres van fundar Android Inc. a Palo Alto, Califòrnia (Estats Units).

En paraules de Rubin, l'objectiu era desenvolupar «dispositius mòbils més elegants que tinguin més en compte la localització i les preferències dels seus amos».

Entre altres empleats inicials importants s'inclouen Andy McFadden, que va treballar amb Rubin en WebTV, i Chris White, que va liderar el disseny i la interfície de WebTV, abans d'ajudar a fundar Android.

Rubin, cofundador de Danger Inc., Miner, cofundador de Wildfire Communications Inc. i vicepresident de tecnologia i innovació a Orange, i els altres empleats inicials van portar una considerable experiència en la indústria sense fils a la companyia. Malgrat els assoliments obvis del passat dels fundadors i primers empleats, Android Inc. va funcionar de manera reservada, admetent simplement que estava treballant en programari per a telèfons mòbils.

L'agost del 2005, Google va adquirir Android Inc. Els empleats principals d'Android Inc., incloent-hi Andy Rubin, Rich Miner i Chris White, van quedar-se a la companyia després de l'adquisició.

En el moment de l'adquisició, a causa del poc coneixement que es tenia sobre el treball d'Android Inc., es va conjecturar que Google estava planejant entrar al mercat dels telèfons mòbils.

A Google, l'equip liderat per Rubin va desenvolupar una plataforma per a dispositius mòbils basada en el nucli Linux. Google va posar al mercat la plataforma per als fabricants de dispositius mòbils i els operadors amb la premissa de proporcionar un sistema flexible i actualitzable. Google va alinear junts una sèrie de socis dedicats a components maquinari i programari i va indicar als operadors que estava obert a diversos graus de cooperació per part seva.

Les especulacions sobre la intenció de Google d'entrar al mercat de les comunicacions mòbils van continuar durant desembre del 2006. Informes de la BBC[23] i *The Wall Street Journal* van indicar que Google volia el seu sistema de cerca i les seves aplicacions en telèfons mòbils i estava treballant de valent per aconseguir-ho. Mitjans de comunicació escrits i en línia aviat van publicar rumors que Google estava desenvolupant un dispositiu amb la marca de fàbrica Google. Alguns van especular que mentre Google definia especificacions tècniques, estava mostrant prototips a fabricants de telèfons mòbils i operadors de xarxa.

(23)British Broadcasting Corporation

El setembre del 2007, *InformationWeek* va cobrir un estudi d'*Evalueserve* que indicava que Google havia registrat diverses patents d'aplicacions en l'àrea de la telefonia mòbil.

El novembre del 2007, es va anunciar a si mateixa l'Open Handset Alliance, un consorci de diverses companyies que inclou Texas Instruments, Broadcom Corporation, Google, HTC, Intel, LG, Marvell Technology Group, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel i T-Mobile. L'objectiu de l'Open Handset Alliance és desenvolupar estàndards oberts per a dispositius

mòbils. El mateix dia, l'Open Handset Alliance també va anunciar el seu primer producte, l'Android, una plataforma per a dispositius mòbils construïda sobre la versió 2.6 del nucli Linux.

El desembre del 2008, s'hi van unir catorze nous membres, incloent-hi PacketVideo, ARM Holdings, Atheros Communications, Asustek Computer Inc., Garmin Ltd., Softbank, Sony Ericsson, Toshiba Corp. i Vodafone Group Plc.

Amb l'excepció de breus períodes d'actualització, l'Android ha estat disponible amb una llicència de programari lliure de codi obert des d'octubre del 2008. Google va publicar tot el codi font (incloent-hi les piles de xarxa i telefonia) amb una llicència Apache. Google també manté pública la llista de problemes revisats perquè qualsevol la pugui veure i comentar.

5.2.2. Història de les versions de l'Android

L'Android ha vist diverses actualitzacions des del llançament original. Aquestes actualitzacions en el sistema operatiu base normalment arreglen errors i afegeixen noves funcionalitats. Generalment cada nova versió del sistema operatiu Android es desenvolupa amb un nom en codi basat en un article de postres.

A continuació descrivim algunes característiques d'algunes versions d'Android disponibles en l'actualitat.

- 2.0/2.1 (**Eclair**), que va millorar la interfície d'usuari i va introduir suport per a HTML5 i Exchange ActiveSync 2.5.

- 2.2 (**Froyo**), que va introduir millores de velocitat amb l'optimització del JIT i el motor Chrome V8 JavaScript, i va afegir suport per a Adobe Flash.

- 2.3 (**Gingerbread**), que refinava la interfície d'usuari, millorava el teclat de programari i característiques de copiar i enganxar, i afegia suport per a NFC.

- 3.0 / 3.2 (**Honeycomb**). Només treballava dispositius de pantalla gran com *tablets* i televisió.

- 4.0 (**Ice Cream Sandwich**). Es va llançar amb la idea d'unificar les capacitats de les dues versions anteriors (Gingerbread i Honeycomb) per disposar d'una que pogués treballar amb *tablets* i *smartphones*.

- 4.1 / 4.3 (**Jelly Bean**). Incorpora suport per Bluetooth de baixa energia, d'OpenGL ES 3.0 i de resolució 4K. També aporta millores en Wi-Fi (seguretat), en codificadors de vídeo (còdec VP8) i monitorit-

zació de notificacions. Va incloure moltes novetats en el conjunt d'aplicacions instal·lades: navegador web, càmera, Google Calendar, Google Maps, etc.

- 4.4 (**KitKat**). Redueix la RAM usada pel sistema operatiu a 512 MB de RAM i disminueix molt el consum de bateria. El canvi més important va ser la introducció d'Android Runtime (ART) com a entorn d'execució d'aplicacions. Es va introduir com un entorn alternatiu i es va mantenir l'antiga màquina virtual Dalvik com a entorn per defecte. La diferència entre Dalvik i ART és el moment en què es realitza la compilació. ART compila l'aplicació en el moment de la seva instal·lació mentre que Dalvik ho fa cada vegada que s'executa. En conseqüència, ART és molt més eficient.

- 5.0 / 5.1 (**Lollipop**). En aquesta versió ART va substituir totalment a Dalvik. Aquest canvi va provocar una millora del rendiment de la bateria gairebé del 85%. També es van introduir millores en les notificacions. Des del punt de vista del disseny i la usabilitat, la interfície va adoptar un nou paradigma «Material design».

- 6.0 (**Marshmallow**). Introdueix un nou sistema de permisos en el qual el permís no és necessari que se sol·liciti en el moment en què s'instal·la l'aplicació. L'aplicació pot preguntar per cada permís de forma individual en el moment en què farà servir la capacitat i l'usuari pot decidir si l'accepta o no. També va incloure una llibreria per al reconeixement d'empremtes digitals. Va incloure la capacitat de tractar l'emmagatzematge extern SD com a emmagatzematge intern. Aquest canvi va facilitar la instal·lació de moltes aplicacions que requerien molt espai d'emmagatzematge. Aquest era un dels grans problemes de les versions anteriors d'Android.

- 7.0 / 7.1 (**Nougat**). Incorpora la possibilitat d'executar múltiples aplicacions en una sola pantalla fent servir la pantalla partida. Permet donar resposta a les notificacions en la pròpia línia de la notificació en el control de notificacions. Incorpora suport per a la llibreria gràfica Vulcan. Aquesta llibreria es coneix com la successora d'Open GL i es caracteritza per una major eficiència en l'ús de tècniques de paral·lelització per a obtenir el màxim rendiment dels processadors.

- 8.0 (**Android O**). Augmenta el control sobre les aplicacions en segon pla per a controlar-ne el consum de recursos i allargar la vida de la bateria. Afegeix millores en el sistema de notificacions i en el sistema de gestió de la informació temporal guardada per les aplicacions. Permet que el sistema alliberi aquest espai si ho considera necessari.

5.2.3. Desenvolupament d'aplicacions per a Android

Per desenvolupar aplicacions per a Android és necessari el SDK associat a la versió d'Android per a la qual volem desenvolupar la nostra aplicació. Fins al 2014 la programació es realitzava utilitzant l'entorn de desenvolupament basat en codi lliure Eclipse.

El 2014 Google va publicar la primera versió estable d'Android Studio, l'entorn de desenvolupament integrat oficial per a la plataforma Android. L'entorn compta amb un editor basat en el programari IntelliJ IDEA de JetBrains i diverses eines com emuladors, dissenyador d'interfícies gràfiques, depurador de codi, etc.

Android Studio s'integra amb Android NDK. NDK és un conjunt d'eines que permeten implementar parts de les aplicacions usant llenguatges de codi natiu com C i C ++. Quan fem servir codi natiu Android Studio ens permet generar una versió de la nostra aplicació per a cada arquitectura de maquinari.

Glossari

AJAX *f* Acrònim d'*asynchronous JavaScript and XML* (JavaScript asíncron i XML), és una tècnica de desenvolupament web per a crear aplicacions interactives o RIA (*rich Internet applications*). Aquestes aplicacions s'executen en el client, és a dir, en el navegador dels usuaris mentre es manté la comunicació asíncrona amb el servidor en segon pla. D'aquesta manera és possible fer canvis sobre les pàgines sense necessitat de recarregar-les, la qual cosa significa augmentar la interactivitat, velocitat i usabilitat en les aplicacions.

ART Acrònim d'Android RunTime, l'entorn d'execució per a la plataforma Android que ha substituït Dalvik en les versions més recents.

Dalvik *f* Màquina virtual que s'utilitza en algunes versions de la plataforma per a dispositius mòbils Android. Ha estat dissenyada per Dan Bornstein amb contribucions d'altres enginyers de Google.

jailbreak *m* Jailbreak (en català, escapar-se de la presó o, més literalment, trencapresons) és un procés que permet als usuaris dels dispositius iPhone, iPod Touch, iPad i Apple TV de totes les generacions executar aplicacions diferents a les allotjades en l'App Store, el lloc oficial de baixada de programes per a aquests dispositius.

llicència Apache *f* Llicència de programari lliure (*Apache license* o *Apache software license* per a versions anteriors a la 2.0) creada per l'Apache Software Foundation (ASF). La llicència Apache (amb versions 1.0, 1.1 i 2.0) requereix la conservació de l'avís de *copyright* i l'advertència, però no és una llicència *copyleft*, ja que no requereix la redistribució del codi font quan es distribueixen versions modificades.

llicència MIT *f* Llicència de programari que ha emprat el Massachusetts Institute of Technology (MIT) al llarg de la seva història, i que potser s'hauria de dir més correctament *llicència X11*, ja que és la llicència que portava aquest programari que mostrava la informació de manera gràfica. Sia com a MIT o X11, però, el text és idèntic.

MIDP *m* El *mobile information device profile* (MIDP) és una implementació que permet escriure aplicacions que es poden baixar i serveis per a dispositius mòbils que es connectin a la Xarxa.

webcast *m* Fitxer multimèdia distribuït per Internet usant tecnologies de reproducció a temps real per a distribuir una única font de contingut a diversos observadors simultanis. Es pot distribuir en directe o per comanda.

Bibliografia

Marco, M. J.; Marco, J. M.; Prieto, J.; Segret, R. (eds.) (2010). *Escaneando la informática*. Barcelona: Editorial UOC. ISBN: 978-84-9788-110-4.

Tabor, M.; Vrdoljak, M. (eds.) (2016). *Guía a la galaxia de aplicaciones móviles*. Enough Software. http://www.mobiledevelopersguide.com

Enllaços d'Internet

http://en.wikipedia.org/

http://www.visionmobile.com/

http://developer.appcelerator.com/

http://www.phonegap.com/

http://softlibre.barrapunto.com/

http://www.nitobi.com/

http://msdn.microsoft.com/

https://channel9.msdn.com/

http://stackoverflow.com/

http://developer.apple.com/

https://developer.android.com/

https://ionicframework.com/

https://www.xamarin.com/

https://unity3d.com/