Univ POLITEHNICA din Bucuresti

Facultatea IMST

Centrul PREMINV

Sistemul de operare Android

CUPRINSUL

CAPITOLUL I

Istoric

1.1 Ce înseamnă Android?

Android este singurul sistem de operare mobil creat de Google. Transformă dispozitivul mobil într-un computer personal de buzunar. Android este echipat cu un browser Web complet şi capacităţi de navigare pe Internet, acces la peste 80.000 de aplicaţii prin Android Market, inclusiv Gmail, Pandora şi Facebook, plus capacitatea de a juca jocuri, de a trimite mesaje de tip SMS şi de a efectua apeluri de pe telefonul. Android este un software open source, ceea ce înseamnă că oricine poate face sistemul de operare mai bun. În acest fel, se beneficiaza nu numai de cunoştinţele dezvoltatorilor Google, ci şi de cele ale dezvoltatorilor terţă-parte. Se poate actualiza şi telefonul Android existent la versiunea 2.2 şi se pot obţine aceleaşi beneficii.
Android este o platform software și un sistem de operare pentru dispositive si telefoane mobile bazată pe nucleul Linux, dezvoltată inițial de compania Google,iar mai târziu de consorțiul comercial Open Handset Alliance.Android permite dezvoltatorilor să scrie cod gestionat în limbajul Java controlând dispozitivul prin intermediul bibliotecilor Java dezvoltate de Google. Aplicațiile scrise C și în alte limbaje pot fi compilate în cod masina ARM și executate, dar acest model de dezvoltare nu este sprijinit oficial de către Google.
Lansarea platformei Android la 5 noiembrie 2007 a fost anunțată prin fondarea Open Handset Alliance un consorțiu de 48 de companii de hardware, software și de telecomunicatii, consacrat.[1]

1.2 Logoul alternative

În iulie 2005 Goole a achiziționat Android, Inc, o mică companie de tip startup cu sediul în Palo Alto, California,SUA . Cofondatorii companiei Android, care au continuat să muncească la Google, au fost Andy Rubin (cofondator al Danger), Rich Miner(cofondator al Wildfire Communications, Inc, Nick Sears (fost vicepreşedinte al T-Mobile și Chris White (unul dintre primii ingineri ai WebTV). La acea dată se cunoștea foarte puțin despre Android, Inc., doar că făceau software pentru telefoane mobile. Aceasta a cauzat zvonuri că Google ar plănui să intre pe piața telefoniei mobile, deși era neclar ce funcție ar putea îndeplini în această piață.

La Google, echipa condusă de Rubin a dezvoltat un sistem de operare pentru dispozitive mobile bazat pe Linux, pe care l-au prezentat producatorilor de telefoane mobile si operatorilor de retele de telefonie mobila, cu perspectiva de a asigura un sistem flexibil, upgradabil.Google a raportat că a aliniat deja serie de parteneri producători de componente hardware și software la noul concept, și a semnalat operatorilor de rețele de telefonie mobilă că era deschis la diferite grade de cooperare din partea acestora. Mai multe speculații că Google ar fi putut intra pe piața telefoniei mobile au apărut în decembrie 2006. Rapoarte de la și Wall Street Journal au remarcat faptul că Google își dorea căutarea web și aplicațiile sale pe telefoane mobile și că lucra din greu către acest

țel. Presa și siturile de știri au publicat curând zvonuri că Google ar dezvolta un dispozitiv mobil marca Google. A urmat și mai multă speculație, susținând că în timp ce Google definea specificațiile tehnice, ar fi demonstrat prototipuri producătorilor de telefoane mobile și operatorilor de rețea. S-a raportat că până la 30 de telefoane prototip operau deja pe piață.

În septembrie 2007 Information Week a publicat un studiu al companiei Evalueserve care dezvăluia că Google a depus cereri pentru mai multe brevete de inventie în domeniul telefoniei mobile. [1]

1.3 Fondarea Open Handset Alliance

La 5 noiembrie 2007 a fost făcut public Open Handsen Alliance un cosortiu incluzând Google, HTC,Intel , Motorola, Qualcom,T-Mobile ,Sprint Nextel și Nvidia,cu scopul de a dezvolta standarde deschise pentru dispozitive mobile. Odată cu formarea Open Handset Alliance, OHA a dezvăluit de asemenea primul său produs, Android, o platforma pentru dispozitive mobile construită pe nucleul Linux, versiunea 2.6.

La 9 decembrie 2008, a fost anunțat că 14 noi membri au aderat la proiectul Android, incluzând Sony Ericson , Vodafone Group, Plc. ARM Holdings Plc, Asustek Computer Inc, Toshiba Corp si Garmin Ltd. Președintele și CEO-ul Google Eric Schmidt a avut nevoie de o bună bucată de timp în comunicatul de presă oficial pentru a elimina toate zvonurile și speculațiile precedente cu privire la existența unui telefon Google.[1]

1.4 Open Source

Începând cu 21 octombrie 2008, Android a fost disponibil ca Open Source. Google a deschis întregul cod sursă (inclusiv suportul pentru rețea și telefonie care anterior era indisponibil, sub licenta Apache). Sub licenta Apache producătorii sunt liberi să adauge extensii proprietare, fără a le face disponibile comunității open source. În timp ce contribuțiile Google la această platformă se așteaptă să rămână open source, numărul versiunilor derivate ar putea exploda.[1]

1.5 Ziua de nastere

Primul smartphone pe care a rulat sistemul de operare Android a fost HTC Dream, cunoscut şi sub numele de T Mobile G1. În februarie 2009, telefonul este lansat pe piaţă cu versiunea 1.5, alias Cupcake, dar care are câteva lipsuri. Totuşi, comparativ cu prima versiune, 1.0, Cupcake face posibilă afişarea tastaturii virtuale pe ecran şi personalizarea ecranului de pornire cu widget-uri. În Android Market stau deja la dispoziţie aproape 5.000 de aplicaţii suplimentare ce pot fi descărcate; dezvoltatorii au remarcat devreme potenţialul ascuns al platformei Google. Parte a filozofiei conceptului Android o reprezintă dotarea ulterioară cu exact funcţiile specifice de care are nevoie fiecare utilizator.

Fig.1.1 HTC Dream (Android 1.0)[3]

Fig.1.2 Cupcake(Android 1.5)-bazat pe kernel-ul Linux 2.6.27[3]

Chiar şi în prezent, premisă pentru accesul la Android Market o reprezintă deţinerea unei adrese de Gmail. În septembrie 2009, a apărut versiunea 1.6 Donut. Pentru prima oară, sistemul de operare Google suportă diferite dimensiuni ale display-ului, precum şi funcţia text to speech, iar prin intermediul VPN asigură o conexiune sigură la internet. Suplimentar, funcţia simplă de căutare s-a extins într-o căutare locală şi una online şi, mai mult, este implementată o bară de pornire rapidă pentru controlul conectivităţii. Iar Android Market de atunci – în tonuri de gri şi negru – a cărui ofertă aproape că se dublase deja, primeşte un design nou, cu mult verde şi subcategorii mai detaliate.[2]

Fig.1.3 Donut (Android 1.6) - lansat pe kernel-ul Linux 2.6.29 [3]

1.6 Primii pasi

Aproximativ în aceeaşi perioadă, Samsung trimite pe piaţă modelul Galaxy I7500, primul său model pe care rulează sistemul de operare Android – şi dezvăluie deja, că dezvoltatorii de hardware nu pot ţine pasul cu numărul mare al programatorilor de aplicaţii. Galaxy vine cu o versiune Android mai veche şi anume 1.5 – până în ziua de azi, multe modele încă mai sunt în

urma stadiului actual. Mai puţin de două luni mai târziu, la sfârşitul lunii octombrie 2009, Android 2.0 – Eclair este gata de lansare. În afara conturilor POP3 şi IMAP, pe smartphone se pot configura mai multe conturi Google de email şi căsuţa de email bussiness prin conexiunea Exchange. O noutate o reprezintă şi posibilitatea de a face zoom prin gesturi multitouch pe fotografii, documente şi o pagină de internet. În plus, camera telefonului mobil suportă flash LED şi zoom digital. În plus, pe lângă Bluetooth 2.1, o dată cu actualizarea Eclair, telefoanele pe care rulează Android sunt dotate cu navigaţie Google Maps. Motorola scoate primul său model Android pe piaţă, pe care rulează versiunea Eclair, şi anume Milestone. Pentru americanii de la Motorola sistemul de operare Android reprezintă colacul de salvare care îi aduce înapoi pe piaţa telefoanelor mobile.

Fig 1.4 Motorola Droid- Éclair (Android2.0/2.1) [3]

La începutul anului 2010 urmează Android 2.1 – noutăţile fiind marginale, este păstrat numele de cod Eclair. Există acum posibilitatea setării imaginilor de fundal animate şi vizualizarea informaţiilor detaliate privind intensitatea semnalului. În afară de aceasta,browser-ul este extins şi suportă HTML 5, precum şi stocare date.[2]

1.7 Invazia androizilor

Mai captivant este dispozitivul pe care este prezentat Android 2.1: cu modelul construit de HTC, Nexus One, Google lansează pe piaţă primul mobil sub marcă proprie, care nu este apreciat foarte tare în rândul consumatorilor şi dispare repede de pe rafturi; în prezent, Google cu modelul Nexus S construit de Samsung încearcă un nou start, de data aceasta succesul fiind mai mare. În afara acestui moment ceva mai slab, popularitatea sistemului de operare creşte: dispozitivele apar ca şi ciupercile după ploaie, iar Android şi-a pus definitiv amprenta pe peisajul smartphone-urilor. O dată cu actualizarea la versiunea 2.2 Froyo în luna mai 2010 şi a suportului unui nucleu de sistem nou care solicită tot mai puţine resurse, viteza de lucru este mult mai mare. Browser-ul web suportă acum şi Adobe Flash şi poate afişa paginile de internet fără limitări, suplimentar setările de securitate pentru conexiunea Exchange au fost extinse.

Cea mai interesantă noutate o reprezintă funcţia de tethering: smartphone-ul poate partaja conexiunea la internet, fie prin intermediul unui cablu USB, fie prin Wi-Fi. Şi în ceea ce priveşte aplicaţiile, există noutăţi: dezvoltatorii au acum posibilitatea, să implementeze o funcţie de push în aplicaţiile lor şi astfel să trimită informaţii direct pe smartphone.

Utilizatorii pot decide dacă doresc instalarea unei aplicaţii în memoria internă a telefonului sau pe un card de memorie SD, având posibilitatea mutării ulterioare a acestei aplicaţii între cele două locaţii. În plus, Android Market a primit o nouă înfăţişare şi deţine peste 50.000 de aplicaţii. În decembrie 2010 este lansată penultima versiune a sistemului Android pentru smartphone-uri şi anume 2.3, sau Gingerbread. Interfaţa de utilizator a acesteia a primit un uşor facelift, precum şi o nouă funcţie ce asigură un control mai bun al aplicaţiilor active. O dată cu lansarea versiunii

Gingerbread se poate integra un cip Near Field Communication (NFC) în dispozitiv, care pregăteşte smartphone-urile pentru plăţile de pe mobil.[2]

Fig. 1.5 Google Nexus One (a fost primul smartphone pentru a primi Android 2.2 Froy)[3]

Fig.1.6 Google Nexus S(a introdus Android 2.3 Gingerbread)[3]

De asemenea, sunt posibile serviciile de videotelefonie şi de telefonie prin internet prin operatorii SIP. Dezvoltatorii de jocuri se bucură de sprijinul senzorilor de mişcare (giroscop), iar cei ce petrec mult timp pe reţele de socializare se bucură de o mai bună integrare a acestora; între timp Android Market a trecut pragul de 200.000 de aplicaţii. De reţinut: datorită update-ului la versiunea Gingerbread 2.3.3, de la începutul anului 2011, s istemul de operare Android suportă procesoare dual core.

Fig.1.7 Motorola xoom (comprimat- a introdus Android 3.0 Honeycomb)[3]

La 22 februarie 2011- a fost lansat Android 3.0 (Honeycomb) SDK, bazat pe kernel-ul Linux 2.6.36.

Fig1.8 SamsungGalaxy Nexus (a introdus Android 4.0.x Cream Sandwich Ice)[3]

SDK pentru Android 4.0.1 (Cream Sandwich de gheaţă), bazat pe Linux kernel 3.0.1 a fost lansat public pe 19 octombrie 2011. Google Gabe Cohen a declarat ca Android 4.0 a fost "teoretic compatibil" cu orice Android dispozitiv 2.3.x în producţie, la acel moment. Codul sursa pentru Android 4.0 a devenit disponibil pe 14 noiembrie 2011

La 27 iunie 2012, Google a anuntat Android 4.1 (Jelly Bean). Bazat pe Linux kernel 3.1.10, Jelly Bean este un update incremental, cu scopul principal de a îmbunătăţi interfaţa cu utilizatorul, atât în termeni de funcţionalitate cat şi performanta[2]

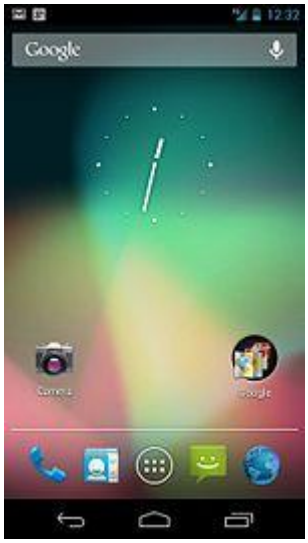

Fig. 1.9 SamsungGalaxy Nexus (Android 4.1)*[3]*

CAPITOLUL 2

Componentele unui smartphone

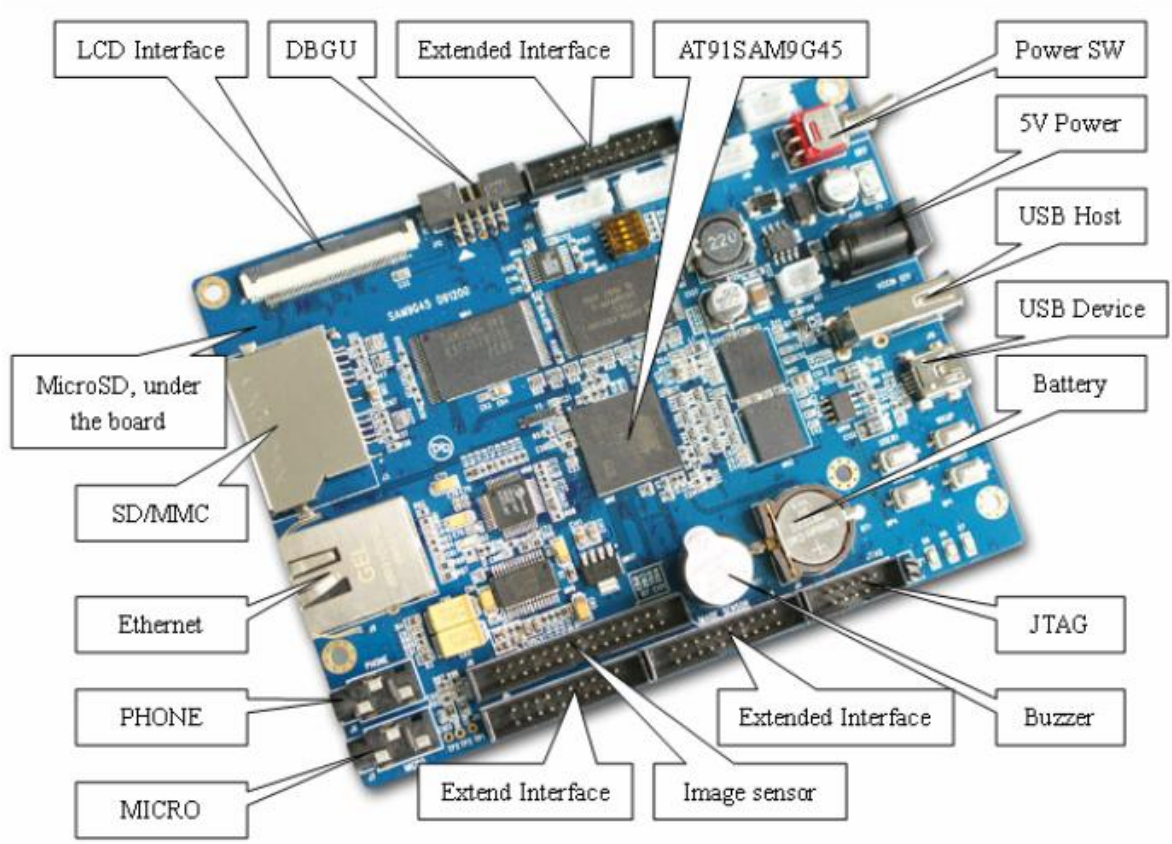

Fig. 2.10 Componentele unui smartphone[4]

1) Microprocesor
2) Coprocesoare Audio/Video
3) Memorie RAM
4) EcranTastatura
5) Dispozitive radio de emisie/receptive
6) Senzori
7) Conectori externi

Microprocesor:

- Arhitectura RISC(Reduced Instruction Set Computing) de tip ARM(Advance RISC Machine)
- Instructiuni simple (nu are virgula mobile)
- Consum redus de energie –Scalarea vitezei in functie de necesitati computationale

Exemple:Qualcomm MSM 8260nSnapdragon 1.2 GHz dual-core processor; SamsungHummingbird 1 GHz ARM Cortex-A8 processor

Coprocesoare Audio/Video:

- Procesoare specializate pe encodare si decodare audio/video
- Capacitate de encodare/decodare HD 1080p H.264
- Exemple:GeForce Ultra Low Power GPU(in cipsetul Nvidia Tegra)
- PowerVR SGX535 GPU(in iPhone 4)

Memoria RAM:

- SDRAM (Synchronus Dynamic Random AccesMemory)
- DDR1/DDR2(Double Data Rate)
- Capacitate:32MB-1GB

Memoria ROM:

- Spatiu special read-only
- Contine imaginea initiala a SO-ului
- Persista in urma unui Hard Reset
- Poate fi scrisa totusi ,in anumite conditii(update de firmware)[5]

Ecran:

- Coponenta esentiala
- Dimensiuni limitate:3,2 inch…4.3inch
- Rezolutii diverse
  240x320(QVGA); 320x480(HVGA); 480x800(WVGA); 640x960(Retina Display)[5]
- Ecran tactil:

Fig.2.11 Ecran rezistiv[5]

Fig.2.12 Ecran capacitiv[5]

1) Rezistiv:
   -constructie tip sandwich
   -necesita apasare (presiune)
   -Avantaje:consum mic de energie, precis
   -Dezavantaje:atingere puternica,sensibil la mediul exterior[5]

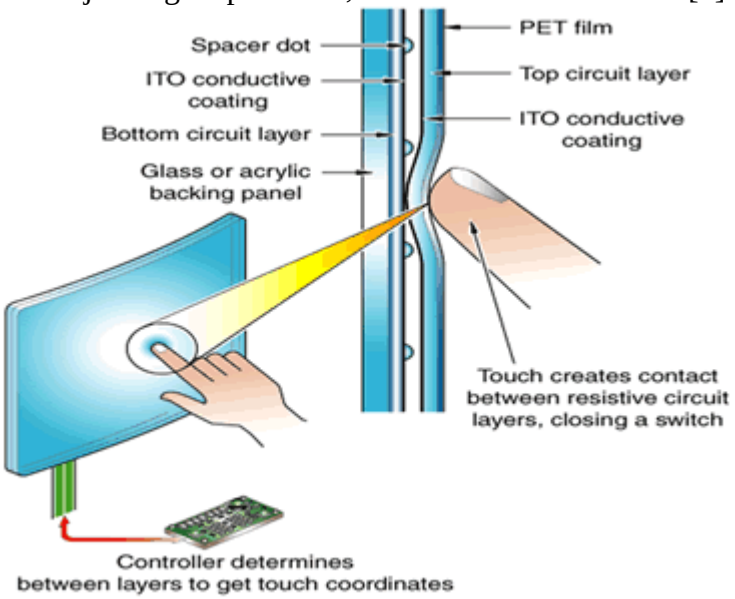

Fig.2.13 Tehnologia ecranelor resistive[5]

2) Ecran Capacitiv:
   -Un strat izolator(sticla/plastic)
   -Nu necesita apasare(presiune)
   -Avantaje:atingere foarte usoara,rezistent la mediul exterior
   -Dezavantaje:inexact, consum mai mare de energie

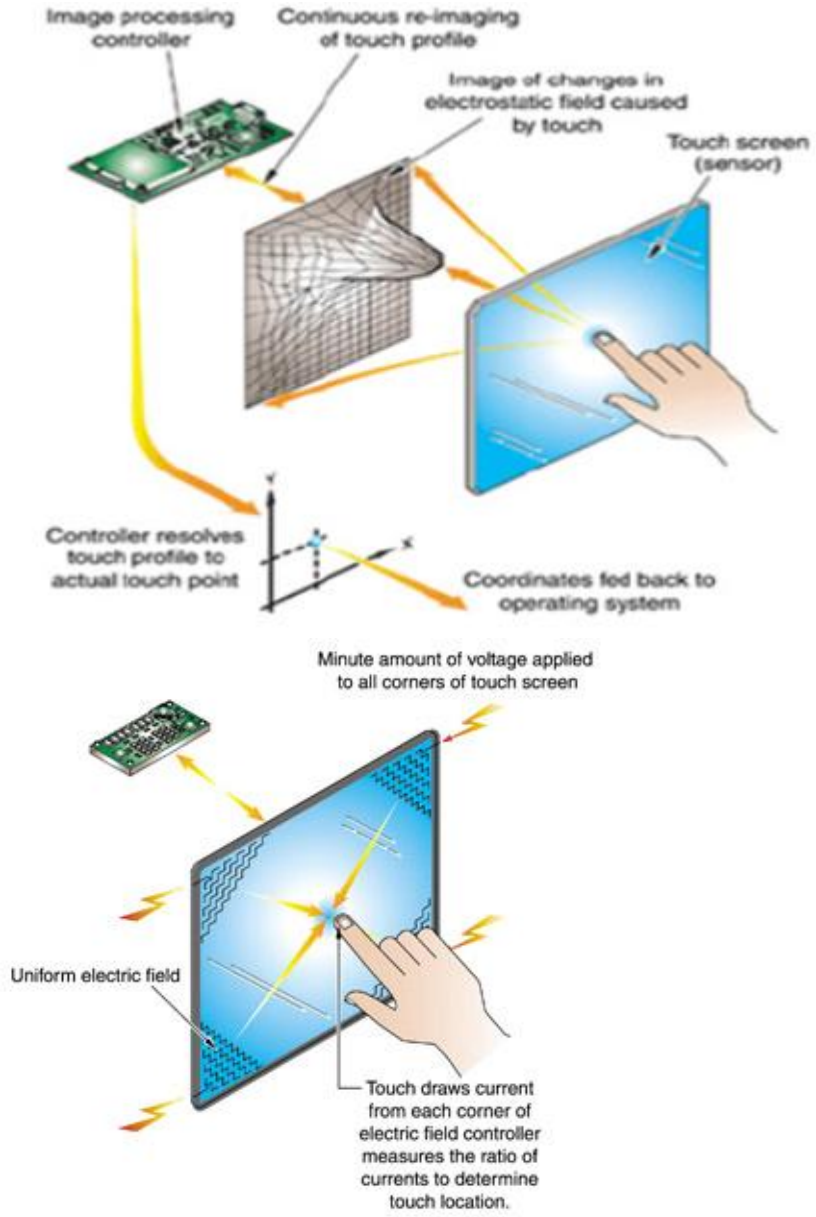

Fig.2.13 Tehnologia ecranelor capacitive[5]

Tastatura

- Tipuri

1) Telefon
   - Doar citire,*si#
   - 3-4 litere per tasta

2) Qwerty –tastatura completa
   - Alte butoane: Menu, Home,Back,Search

Dispozitive radio de emisie/receptive:

- Telefonie
  -GSM/UMTS/HSDPA
  -900 MHz/1800 MHz/2100 MHz

-Pana la 25 km

- Bluetooth
  -Comunicatie pe distante foarte scurte
  -Foloseste protocoale nestandard

Senzori

- GPS(Global Poisitioning System)
- Accelerometru
- Aparat foto
- Busola
- Senzor de proximitate[5]

CAPITOLUL 3

Sistemul de operare Android

3.1 Despre Android

Pana de curand ,telefoanele mobile au fost medii inchise construite pe sisteme de operare proprietare foarte fragmentate,sisteme care aveau nevoie de tool-uri de dezvoltare specific.Telefoanele deseori prioritizau rularea aplicatiilor native in detrimental aplicatiilor scrise de diversi dezvoltatori. Acest lucru a introdus o bariera artificial pentru dezvoltatori,acestia fiind nevoiti sa astepte dupa hardware din ce in ce mai puternic ,pentru dezvoltarea aplicatiilor.In android aplicatiile native si cele create de terti dezvoltatori sunt scrise folosind aceleasi API-uri si sunt executate in acelasi timp. Aceste API-uri ofera acces la hardware-ul telefonului,inregistrari video,comunicarea dintre aplicatii si grafica 2D si 3D.

Android are niste API-uri puternice,excelent documentate,o comunitate de dezvoltatori in continua expansiune si fara costuri in ceea ce priveste dezvoltarea si distributia aplicatiilor.Dupa cum dispozitivele mobile continua sa creasca in popularitate, Android ofera o oprtunitate interesanta pentru crearea de aplicatii inovative pentru telefoane, indiferent de experienta dezvoltatorului.

Dezvoltatorii ,care programau utilizand cod C sauC++ de nivel jos,trebuiau sa inteleaga hardware-ul specific pentru care dezvoltau,in general un singur dispozitiv,sau o gama de dispositive asemanatoare ale acceluiasi producator. Dupa cum tehnologia hardware si accesul la internet mobil au evoluat, aceasta abordare restrictiva a devenit depasita.[8]

Platforme precum sistemul Symbian au fost create pentru a asigura dezvoltatorilor un target hardware mai amplu. Aceste sisteme s-au dovedit a fi de success in incurajarea dezvoltatorilor de aplicatii pentru mobile sa dezvolte aplicatii care profitau mai mult de hardware-ul disponibil.

Aceste platforme ofera ceva acces la hardware-ul telefonului, dar tot au nevoie ca dezvoltatorul sa scrie cod C/C++ complex si sa foloseasca API-uri proprietare greu de folosit. Aceste dificultati sunt amplificate pentru aplicatiile care functioneaza pe diverse implementari hardware , in special cele care se folosesc de un anumit modul hardware , cum ar fi GPS-ul. Cele mai notabile progrese in dezvoltarea pe telefoane mobile a fost introducerea de MID-leti Java.MID-letii sunt executati intr-o masina virtual Java , un proces care nu tine cont de hardware-ul aflat pe telefon si permite dezvoltatorilor sa dezvolte aplicatii pentru o gama mai larga de telefoane. Din nefericire ,acest lucru creaza niste dificultati in ceea ce priveste accesul la hardware-ul dispozitivului.[8]

In dezvoltarea aplicatiilor mobile,era considerat normal ca aplicatiile terte sa primeasca acces diferit la hardwre si grile de executie , decat aplicatiile native scrise de catre producatorul telefonului. Astfel, MID- leti Java aveau acces restrictionat catre hardware si catre grilele de executie.

Introducerea de MiD-leti Java a marit randurile dezvoltatorilor , dar lipsa accesului la hardware si executia intr-o masina virtual a insemnat ca marea majoritate a aplicatiilor mobile sa

fie programe desktop normale sau website-uri si care nu se folosesc de avantajele oferite de catre un dispozitiv mobil.

Android se situeaza in noul val de sisteme de operare mobile create special pentru hardware in continua crestere de putere. Windows Mobile,Apple iPhone si Palm Pre asigura un mediu de dezvoltare mai bogat si mai simplu pentru aplicatiile mobile.Dar spre deosebire de Android , sunt construite pe sisteme de operare proprietare care in unele cazuri proritizeaza aplicatiile native in detrimental celor create de terti utilizatori,restrictioneaza transferul de date intre aplicatii terte si aplicatiile native ale telefonului si restrictioneaza sau controleaza distributia aplicatiilor catre platforme. [8]

Android ofera noi posibilitati pentru aplicatiile mobile deoarece efera un mediu de dezvoltare deschis construit peste un kernel Linux open source.Accesul la hardware ne este disponibil tuturor aplicatiilor prin intermediul unor librarii de API-uri, iar interactiunea dintre aplicatii, desi este contolata cu grija, este suportata in intregime.

In Android, toate aplicatiile au un statut egal. Aplicatiile terte si cele native sunt scrise folosind acceleasi API-uri si sunt executate in accelasi mediu de executie.Utilizatorii pot scoate si inlocui orice aplicatie nativa cu o alternativa oferita de catre terti dezvoltatori;chiar si tastatura si ecranul de start pot fi inlocuite[8]

3.2 Caracteristici si specificații ale sistemului Android

Tabel 1 Caracteristici si specificații ale sistemului Android

Configurații dispozitive	Platforma este adaptabilă la configurații mai mari, VGA, biblioteci grafice 2D biblioteci grafice 3D bazate pe specificația OpenGL ES 1.0 și configurații tradiționale smartphone.
Stocare de date	Software-ul de baze de date SQlite este utilizat în scopul stocării datelor
Conectivitate	Android suportă tehnologii de conectivitate incluzând GSM/EDG, CDMA, EV-DO, UMTS,Bluetooth si Wi-Fi.
Mesagerie instant	SMS și MMS sunt formele de mesagerie instant disponibile, inclusiv conversații de mesaje text.
Navigatorul de web	Navigatorul de web disponibil în Android este bazat pe platforma de aplicatii open source WebKit.
Mașina virtuală Dalvik	Software-ul scris în Java poate fi compilat în cod mașină Dalvik și executat de masina virtuala Dalvik care este o implementare specializată de mașină virtuală concepută pentru utilizarea în dispozitivele mobile, deși teoretic nu este o Mașină Virtuală Java standard.
Suport media	Android acceptă următoarele formate media audio/video/imagine:MPEG-4,H.264 , MP3,AAC,OGG, AMR, JPEG, PNG, GIF.
Suport hardware adițional	Android poate utiliza camere video/foto,touchscreen,GPS,accelerometru , și grafică accelerată 3D.
Mediu de dezvoltare	Include un emulator de dispozitive, unelte de depanare, profilare de memorie și de performanta, un plug-in pentru mediul de dezvoltare Eclipse.

Piața Android	Similar cu App Store-ul de pe I Phone, Piața Android este un catalog de aplicații care pot fi descărcate și instalate pe hardware-ul țintă prin comunicație fără fir, fără a se utiliza un PC. Inițial au fost acceptate doar aplicații gratuite. Aplicații contra cost sunt disponibile pe Piața Android începând cu 19 februarie 2009.
Multi-touch	Android are suport nativ pentru multi-touch , dar această funcționalitate este dezactivată (posibil pentru a se evita încălcarea brevetelor Apple pe tehnologia touch-screen) .O modificare neoficială, care permite multi-touch a fost dezvoltată.

[1]

3.3Arhitectura sistemului Android

Android este alcatuit din mai multe straturi care comunica intre ele. Exista 5 straturi principale:Linux Kernel, Libraries, Android Runtime, Application Framework si Applications. Fiecare strat foloseste serviciile aduse de stratul de sub el.

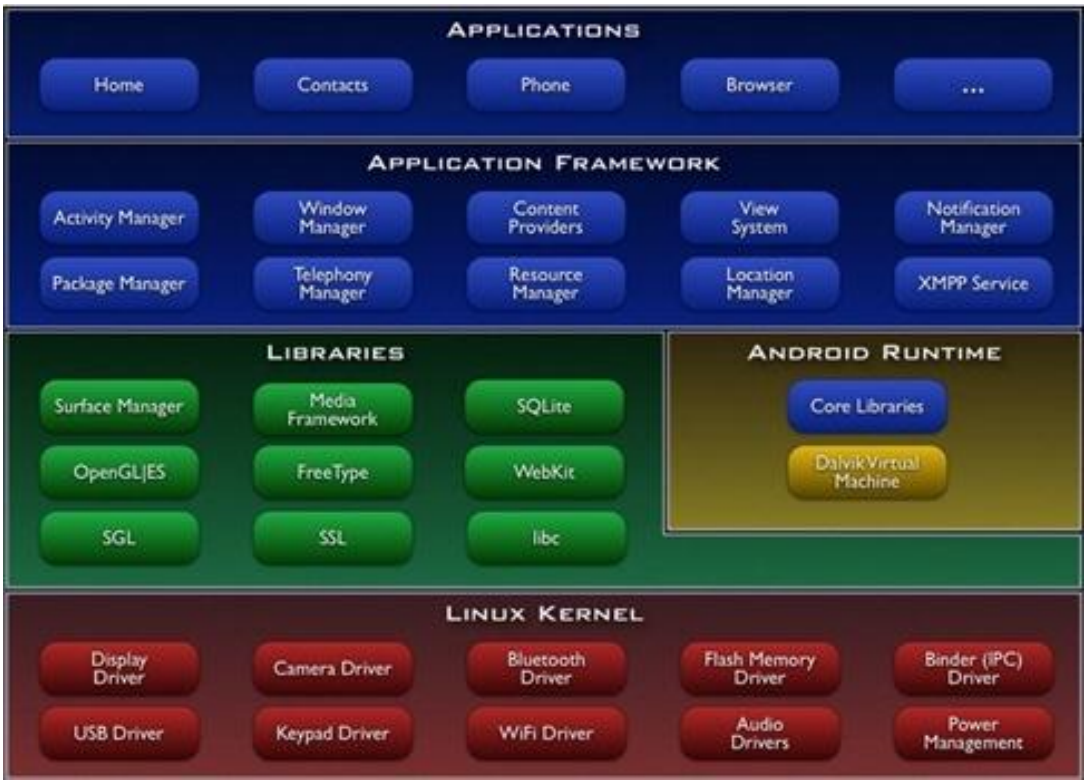

Fig.3.14 Arhitectura sistemului Android

Primul strat este **Linux Kernel** . Android este construit pe o fundatie destul de solida: un kernel de Linux.Acest layer este rezervat sistemului, programatorul si utilizatorul nu vor utiliza direct serviciile sale.[6]

Al doilea strat, **Native Libraries**, reprezintă bibliotecile native Android. Sunt scrise în C sau în C++ şi sunt compilate pentru un anumit hardware. Aici sunt bibliotecile responsabile de gestiunea bazelor de date (SQLite), redarea filmelor (Media Framework) şi afişarea fişierelor HTML (WebKit).

Android Runtime conţine maşina virtuală Dalvik şi bibliotecile Java. Aplicaţiile Android rulează pe maşina virtuală Dalvik, programele sunt scrise în Java şi compilate în bytecode.Fişierele .class sunt transformate în fişiere executabile Dalvik ".dex". Toate acestea,

pentru că aplicaţiile ruleaza pe un sistem cu memorie destul de limitată şi cu o putere de procesare mai mica.

Application Framework este partea cu care lucrează direct programatorul. Aceasta este preinstalată în Android, însa serviciile se pot extinde şi se pot crea propriile componente. Cele mai importante componente sunt:

- Activity Manager: controlează ciclu de viaţă al aplicaţiilor şi stiva de navigare a utilizatorului
- Content Providers: conţine datele ce sunt împărţite între aplicaţii (ex: contactele din telefon)
- Location Manager: acces la GPS
- Notification Manager: alerte trimise utilizatorului că ceva s-a întamplat în background

Applications reprezintă ceea ce vede utilizatorul şi ceea ce el foloseste: aplicaţiile şi widgeturile.[6]

3.4 Aplicatiile sistemului Android

Cele mai importante component ale unei aplicatii Android sunt:

3.4.1 Activity(Activitate)

Reprezinta o interfata cu utilizatorul,fereastra sau formular. O aplicatie Android poate avea una sau mai multe activitati:de exemplu o aplicatie de tip Agenda poate avea o activitate pentru a gestiona contactele, o activitate pentru a gestiona intalniri si una pentru a edita o intrare in agenda.

Fiecare Activitate are propriul sau ciclu de viata, independent de ciclul de viata al procesului asociat aplicatiei

Fiecare activitate are propria stare si datele acesteia pot fi salvate sau restaurate.Activitatile pot fi pornite de aplicatii diferite (daca e permis).Are un ciclu de viata complex deoarece aplicatiile pot avea activitati multiple si doar una este in prim-plan;utilizand managerul de activitati, sistemul Android gestioneaza o stiva de activitati care se gasesc in diferite stari(pornire,in executie, intrerupta,oprita, distrusa);

In SDK, activitatea este implementata folosind o subclasa a clasei Activity care extinde clasa Context;

Ciclul de viata al unei activitati

Ciclul de viata al unei activitati descrie starea in care o activitate poate fi la un moment dat:[7]

Running-Activitatea a fost creata (onCreate()),pornita (onStart()) si este afisata pe ecranul aparatului;in cazul in care activitatea a mai fost utilizata si aplicatia a salvat starea accesteia(onSaveInstanceState()),activitatea este reluata din acel punct(onRestoreInstanceState() si on Resume()); in aceasta stare utilizatorul interactioneaza cu activitatea prin intermediul interfetei dispozitivului(tastatura,touchscreen, display);

Paused-Activitatea pierde prim-planul (onPause()),deoarece o alta activitate este executata, cum ar fi o fereastra de dialog, de asemenea,in cazul, in care aparatul intra in modul sleep, activitatea este oprita temporar;activitatea isi poate relua executia(onResume()) si este plasata inapoi in prim-plan;

Stopped-Activitatea nu mai este in uz si pentru ca este oprita(onStop()) nu este vizibila;pentru a fi reactivata (ea déjà exista),activitatea trebuie sa fie repornita(onRestart() si onStart()) si reluata(onResume());

Destroved-Activitatea este distrusa (onDestroy()) si memoria s-a eliberat,deoarece nu mai este necesara sau sistemul are nevoie de memorie suplimentara pentru rutinele proprii sau pentru alte activitati;deoarece managementul memoriei este un aspect important pentru sistemul de

operare Linuxal dispozitivului mobil,procesul care gazduieste o activitate intrerupta, prita sau distrusa, poate fi terminat pentru a elibera memorie pentru noi activitati;doar procesele ce gestioneaza activitati sunt protejate;

Activitatea are mai multe stari intre care exista tranzitii clare. In ciuda faptului ca lucrurile pot arata complicat , in realitate ele sunt mult mai simple daca ne concentram pe urmatoarele elemente:

O singura activitate poate fi in prim- plan la un moment dat;
Doar sistemul gestioneaza starile si tranzitiile unei activitati si nu programatorul(nu in mod direct, deoarece atunci cand se lanseaza o activitate noua se modifica implicit starea activitatii curente)
Sistemul va anunta atunci cand activitatea isi schimba starea prin intermediul handler-elor(metode de forma onXXX())pentru evanimentele de tip tranzitie;programatorul poate adauga propriul cod pentru supradefinirea acestor metode:[7]

onCreate(Bundle)-apela; cand activitatea este creata folosind argumentul metodei de tip Bundle exista posibilitatea sa restabiliti starea activitatii,care a fost salvata intr-o sesiune anterioara;dupa ce activitatea a fost create, va fi pornita(onStart());

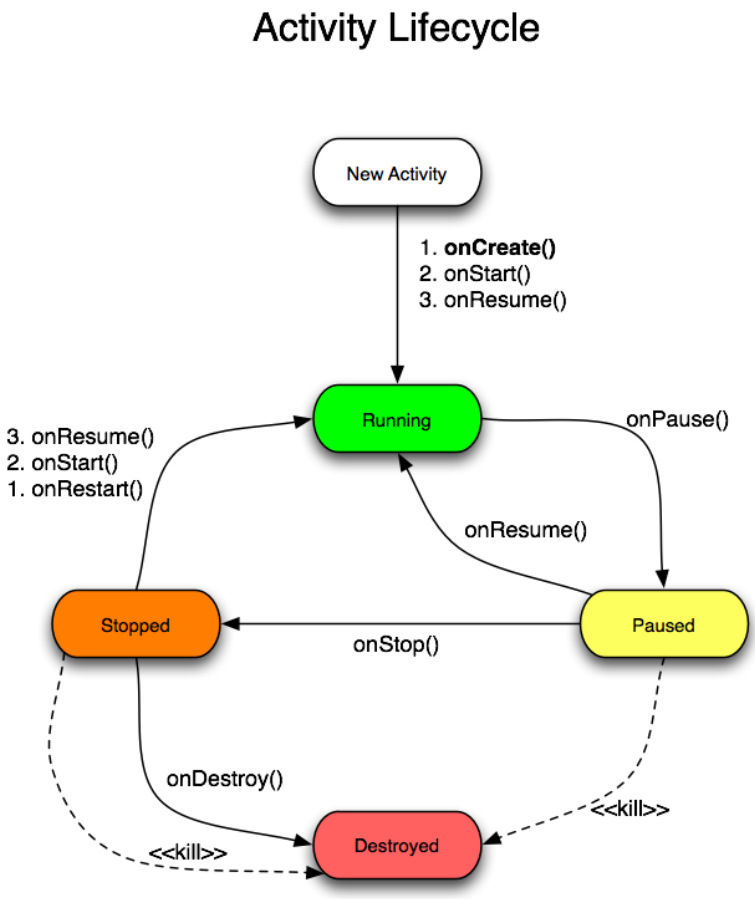

Fig 3.15 Ciclul de viata al unei activitati[6]

onStart()-apelata in cazul in care activitatea urmeaza sa fie afisata;din acest punct , activitatea poate veni in prim plan (onResume()) sau ramane ascunsa in fundal (onStop());

onRestoreInstanceState(Bundle)-apeleta in cazul in care activitatea este initializata cu datele dintr-o stare anterioara, ce a fost salvata; in mod implicit, sistemul restaureaza starea interfetei cu utilizatorul(starea controalelor vizuale, pozitia cursorului,etc);[7]

onResume()-apelata cand activitatea este vizibila iar utilizatorul poate interactiona cu aceasta;din aceasta stare ,activitatea poate fi plasata in fundal, devenind intrerupta(onPause);

onRestart ()-apelata in cazul in care activitatea revine in prim-plan dintr-o stare oprita(stopped); dupa aceasta activitatea este pornita(onStart() din nou;

onPaused()-apelata atunci cand sistemul aduce in prim-plan o alta activitate;activitatea curenta este mutate in fundal si mai tarziu poate fi oprita(onStop() sau repornita si afisata(onResume()); acesta este un moment bun pentru a salva datele aplicatiei intr-un mediu de stocare persistent(fisiere, baze de date)deoarece dupa aceasta faza de activitate poate fi terminata si distrusa fara a se anunta acest lucru;[7]

onSaveInstanceState(Bundle)-apelata pentru a salva starea curenta a activitatii;in mod implicit , sistemul salveaza starea interfetei cu utilizatorul;

onStop()-apelata in cazul in care activitatea nu mai este utilizata si nu mai este vizibila deoarece o alta activitate ineractioneaza cu utilizatorul; din acest punct, activitatea poate fi repornita(onRestart()) sau distrusa (onDestroy();

onDestroy()-apelata in cazul in care activitatea este distrusa , iar memoria sa eliberata; acest lucru se pate intampla in cazul in care sistemul necesita mai multa memorie sau daca programatorul termina explicit activitatea apeland metoda finish() din clasa Activity;

Deoarece tranzitiile dintre straturi sunt descrise prin apeluri catre diferite metode de tipul onXXX(),ciclul de viata al unei activitati poate fi descris de succesiunea posibila a acestor apeluri. [7]

3.4.2 Intent(Intentie)

Reprezinta o entitate folosita pentru a descrie o operatie care urmeaza sa fie executata;
Oarecum similar cu conceptual de event-handler din NET sau Java;
Un mesaj asincron utilizat pentru a activa activitati sau servicii;
Gestionata de o instant a clasei Intent

3.4.3. Service(Serviciu):

- Un task care se executa in fundal, fara interactiunea directa cu utilizatorul;
- Gestionata de o instant a clasei Service;
- Content provider(Furnizor sau manager de continut);
- Un API folosit pentru a gestiona datele private ale aplicatiei
- Un sistem de management de date ce descrie o alternativa la sistemul de fisiere, baze de date SQLite sau orice alta solutie de stocare persistenta;
- Implementata de o subclasa a clasei ContentProvider;
- solutie pentru a partaja si controla (pe baza de pemisiuni) transferul de date intre aplicatii(de exemplu , sistemul Android ofera un furnizor de continut pentru datele de contact);
- Broadcast receiver;
- component care raspunde la anunturi difuzate(propagate)la nivel de sistem;
- Oarecum similar cu conceptual de handler global (sau evenimente de sistem);
- Implementata de o subclasa a clasei BroadcastReceiver

3.5 Despre aplicatie

Aplicatia este de tip „Whiteboard” . O astfel de aplcatie permite mai multor utilizatori de terminale Android sa imparta o coala de hartie virtual folosita pentru a desena sau a scrie ceva . Ideea este ca informatia sa se propage automat intre terminale,intr-un timp cat mai scurt , preferabil in timp real,astfel incat toti utilizatorii sa aiba desenat acelasi lucru pe ecranul telefonului.De principiu , functia de a desena sau scrie ceva pe coala de hartie virtual, nu ar trebui sa fie limitata la un singur terminal Android. In functie de setarile facute de utilizator, vor exista doua moduri de functionare si de interconectare ale aplicatiei:

Un singur utilizator modifica informatia, aceasta fiind apoi propagata catre celelalte terminale. Mai multi utilizatori modifica informatia,pe rand, si aceasta este propagate intre terminale.Aplicatia va avea un mecanism de protectie astfel incat, doi utilizatori sa nu poata modifica informatia in paralel.[8]

Aplicatii preinstalate livrate cu sistemul: un mediu de dezvoltare software pentru a crea aplicatii; include diferite unelte , plug-in-uri sau documentatia necesara de aplicatii.Ceea ce intregeste cu adevarat Android este filozofia open source a sistemului, care permite ca orice dezvoltator sa poata fixa orice deficienta in interfata cu utilizatorul sau orice problema de design a aplicatiilor native prin scrierea unor extensii sau prin inlocuirea aplicatiei. Android ofera dezvoltatorilor oportunitatea de a crea aplicatii pentru telefoanele mobile exact cum au fost imaginate de fiecare.[8]

3.6 Rutare (Sistemul de operare Android)

Rutarea este un proces ce permite utilizatorilor de smartphone-uri, tablete și alte dispozitive pe care rulează sistemul de operare Android să obțină control privilegiat (cunoscut ca „acces la rădăcină”) în interiorul subsistemului Android. Rutarea este deseori efectuată cu scopul de a depăși limitările pe care purtătorii și producătorii hardware le pun pe anumite dispozitive, rezultand în abilitatea de a modifica sau înlocui aplicațiile de sistem și setările, de a rula aplicații specializate care solicită permisiuni la nivel de administrator sau de a efectua alte operații care altfel sunt inaccesibile unui utilizator Android normal. Rutarea este analoagă cu dispozitive jailbreaking care rulează sistemul de operare Apple Ios sau Sony PlayStation 3. Pe Android, rutarea poate să faciliteze, de asemenea, eliminarea completă sau înlocuirea sistemului de operare al dispozitivului, de obicei cu o lansare mai recentă a sistemului de operare.

Deoarece Android a fost derivat din nucleul Linux, rutarea unui dispozitiv Android este similară în practică pentru a accesa permisiuni administrative pe Linux sau orice alt sistem de operare asemenător Linux-ului cum ar fi FreeBSD sau OS X.[9]

3.6.1 Descriere

Rutarea permite tuturor aplicațiilor instalate de utilizator să ruleze comenzi privilegiate care sunt tipic indisponibile dispozitivelor în configurația lor de baza. Rutarea este cerută pentru operații mai avansate și potențial periculoase incluzând modificarea sau ștergerea fișierelor de sistem, eliminarea aplicațiilor instalate de purtători sau producători și acces la nivelul de jos al hardware-ului însuși (rebutare, controlarea statutului luminilor sau reverificarea intrărilor). O

instalare tipica instaleaza de asemenea aplicatia Superuser care supervizeaza aplicatiile carora li s-au dat privilegii.
A doua operatie, deblocarea verificarii bootloader a dispozitivului este ceruta pentru a elimina sau inlocui sistemul de operare instalat.

Spre deosebire iOS jailbreaking, rutarea nu este necesara pentru a rula aplicatii distribuite in afara magazinului Google Play, uneori numit „sideloading". Android-OS suporta aceasta caracteristica nativa in doua moduri: prin intermediul optiunii „Unknown source" din meniul setari si prin intermediul Android Debug Bridge. Totusi unii purtatori cum ar fi AT&,previn instalarea aplicatiilor care nu se gasesc in magazin in firmware,desi mai multe dispozitive (incluzand Samsung Imfuse 4 G) nu fac subiectul acestei reguli, si AT&T a ridicat de atunci restrictia pe mai multe dispozitive vechi.

Din 2012 optiunile implicite ale Amazon Kindle Fire sunt cele din Amazon Appstore in locul celor din Google Play, desi la fel ca multe alte dispozitive Android, Kindle Fire permite incarcarea aplicatiilor din surse necunoscute si aplicatia „easy installer" din Amazon Appstore face acest lucru usor. Alti vanzatori de dispozitive Android pot bloca alte surse in viitor. Accesul la aplicatii alternative poate solocita rutare, desi nu e intotdeauna necesara.
Termenul „bricking" este folosit pentru a descrie un dispozitiv care a avut propriul software modificat incorect pana in punctul in care nu mai functioneaza cum a fost initial conceput.[9]

3.6.2 Proces

Procesul de rutare variaza mult in functie de dispozitiv dar de obicei include exploatarea unei slabiciuni de securitate in firmware a dispozitivului,si apoi copierea su binary intr-o locatie din calea procesului curent (e.g./system/xbin/su) si acorda permisiuni executabile cu comanda shmod. O aplicatia de supervizare precum SuperUser sau SuperSU poate regulariza si loga cererile de permisiuni ridicate de la alte aplicatii.Multe ghiduri tutoriale si procese automate exista pentru dispozitivele populare Android ce faciliteaza un proces de rutare rapid si usor;

De exemplu la scurt timp dupa ce T-Mobile G1 a fost lansat s-a descoperit repede ca orice se tasta era interpretata ca o comanda intr-un shell privilegiat al radacinii. Desi Google a lansat repede o modalitate de a fixa eroarea o imagine semnata a vechiului firmware s-a scurs ceea ce a dat utilizatorilor posibilitatea de a degrada si de a folosi exploatarea originala pentru a castiga acces la radacina. De indata ce exploatarea a fost descoperita o imagine pesonalizata de recuperare care trece de verificarea semnaturii digitale al unui pachet de reactualizare a firmware-ului se poate ilumina. In schimb folosind recuperarea personalizata poate fi instalata o reactualizare modificata a firmware-ului care include utilitarele(de exemplu Superuserapp) necesare pentru a rula aplicatiile ca radacina.
Dispozitivele Google-branded Android, Nexus One, Nexus S si Galaxy Nexus pot fi deblocate la incarcare prin simpla conectare a dispozitivului la un calculator in timp ce sunt in modul bootloader si ruleaza programul Fastboot cu comanda „fastboot oem unlock".Dupa acceptarea avertizarii ca bootloader-ul va fi deblocat o noua imagine sistem poate fi scrisa direct in flash fara nevoia unei exploatari.

Recent Motorola, LG Electronics si HTC Corporation au adaugat caracteristici de securitate dispozitivelor lor la nivel hardware in incercarea de a preveni dispozitivele Android vandute de a fi rutate. De exemplu Motorola Droid x are o securitate a bootloader-ului care va pune telefonul in „recovery mode" daca firmware nesemnat este incarcat in dispozitv si Samsung Galaxy S II va afisa un indicator triunghiular galben daca firmware-ul dispozitivului a fost modificat. [9]

3.6.3 Reactia industriei

Pana recent, raspunsul producatorilor de tablete si smartphon-uri si purtatori mobili a fost de a nu sprijini dezvoltarea firmware-ului tertiar. Producatorii si-au exprimat ingrijorarea cu privire la functionarea improprie a dispozitivelor care ruleaza software neoficial si costuri de sprijin. In plus firmware precum CyanogenMod ofera uneori caracteristici pentru care altfel ,purtatorii ar solicita premium (e.g., tethering). Ca rezultat au fost introduse obstacole tehnice cum ar fi bootloader-uri blocate si acces restrictionat la permisiunile radacinii in multe dispozitive. De exemplu, la sfarsitul lui decembrie 2011, Barnes and Noble si Amazon.com, Inc. au inceput sa grabeasca reactualizari firmware over-the-air automate, 1.4.1 la tabletele Nook si 6.2.1 la Kindle Fires, care au inlaturat abilitatea utilizatorilor de a castiga acces la radacina dispozitivelor. Reactualizarea 1.4.1 la tableta Nook a inlaturat de asemenea abilitatea de incarcare a aplicatiilor din surse diferite de cele oficiale ale magazinului Barnes[9]

3.6.4 and Noble app fara modificare.

Totusi software-ul dezvoltat in comunitate a devenit popular in ultima vreme si urmand o afirmatie a Librarian of Congress (US) care permite folosirea dispozitivelor mobile „jailbreaking", producatorii si purtatorii au devenit mai maleabili cu privire la distributiile CyanogenMod si alte firmware neoficiale, precum HTC, Samsung, Motorola si Sony Ericsson, acordand sprijin si incurajarea dezvoltarii.

In 2011, necesitatea de a eluda restrictiile hardware pentru a instala firmware neoficial s-au diminuat datorita cresterii numarului de dispozitive cumparate cu bootloader-uri deblocate sau deblocabile similare seriei de telefoane Nexus. Producatorul de dispozitive HTC a anuntat ca va sprijini dezvoltatorii de software de pe piata prin construirea bootloader-urilor pentru noile dispozitive neblocabile.

3.6.5 Legalitate

Pe 26 iulie 2010, biroul Copyrigt U.S. a anuntat o noua scutire facand oficial legala rutarea unui dispozitiv si rularea aplicatiilor tertiare neautorizate precum si abilitatea de a debloca orice telefon mobil pentru utilizarea la purtatori multipli.[9]

CAPITOLUL 4

Masina virtuala Dalvik

Masina virtuala (VM) este un mediu software care poate fi un emulator, un sistem de operare sau o virtualizare hardware completa, care are o implementare de resurse fara ca hardware –ul actual sa fie present.

Deoarece un emulator permite aplicatiilor si sistemelor de operare sa ruleze pe un hardware care are arhitectura procesorului diferita de a celuilalt, in timp ce un sistem de operare VM virtualizeaza serverul pe sistemul de operare si in cazul virtualizarii hardware doua sau mai multe sisteme de operare diferite pot rula simultan pe acelesi hardware.Astfel ideea principala a masinii virtuale este de a furniza un mediu care poate executa instructiuni diferite de cele asociate cu mediul care gazduieste hardware si software,

4.1 Tipuri de masini virtuale

VM-urile actioneaza ca un hardware care executa operatii ca si cum hardware-ul este prezent.VM poate fi impartita in doua categorii bazate pe lucru si functionalitate:

1.VM bazata pe stiva(foloseste instructiuni pentru a incarca intr-o stiva de executie)

2.VM bazata pe registri (foloseste instructiuni de codificare in registri sursa si destinatie)

In continuare facem o scurta comparatie intre cele doua categorii.

4.1.1 VM sistem vs VM proces

Masinile virtuale se divid in doua mari categorii, i.e. VM sistem (cunoscuta de asemenea ca masina virtual hardware) si VM proces (cunoscuta de asemenea ca masina virtual de aplicatii). Clasificarea este bazata pe folosirea lor si nivelul de corespondenta cu masina fizica asociata.VM-ul sistem simuleaza complet stiva hardware de sistem si suporta executia complet a unui sistem de operare.Pe de alta parte, VM proces adauga un layer peste sistemul de operare care este folosit pentru a simula mediul de programare pentru executia proceselor individuale. Masinile virtuale sunt folosite pentru a face share si a specifica resurse de sistem adecvate software-ului (pot fi sisteme de operare multiple sau o aplicatie); si aceste software sunt limitate de resursele lor furnizate de VM. Masina virtuala monitor(de asemenea cunoscuta ca Supervizor) este software-ul layer actual care furnizeaza virtualizarea. Supervizorii sunt de doua tipuri, depinzand de asocierea lor cu hardware-ul subiacent. Supervizorul care are controlul direct al hardware-ului subiacent este cunoscut ca VM nativa sau bare- metal; In timp ce VM gazduita este un soft layer diferit care se executa intr-un sistem de operare si prin urmare are asociere indirecta cu hardware-ul subiacent.VM sistem abstractizeaza Instruction Set Architecture (ISA), care este putin diferita de platforma hardware reala. Avantajele principale ale unei VM sistem include consolidare (permite coexistenta sistemelor de operare multiple pe un singur calculator cu o puternica izolare intre ele), furnizare de aplicatii, intretinere high availability si disaster recovery.In afara de avantaje ulterioare in ce priveste aspect de dezvoltare avantajele lor sunt ca permit salvare, repornire rapida si acces la depanare mai bun.[10]

Aplicatia sau VM proces permit executie normal a aplicatiei in interiorul sistemului de operare subiacent pentru a suporta un singur proces. Putem crea mai multe instante ale VM-lui proces pentru a permite executia mai multor aplicatii asociate cu mai multe procese. VM proces este creata cand incepe procesul si se opreste cand se termina procesul. Scopul principal al VM proces este de furniza dependent pe platforma (in termeni de mediu de programare),aceasta insemnand permiterea executiei unei aplicatii in acelasi mod pe orice platforma hardware si software subiacenta. In comparative cu VM sistem (unde abstractizarea low-level a ISA este furnizata), VM proces abstractizeaza limbaje de programare high-level.VM process este implementata folosind un interpretator; totusi performanta comparabila cu a limbajelor de programare bazate pe compilator este atinsa prin metoda just-in –time compilation .
Doua din cele mai populare exemple de VM-ri proces, Java Virtual Machine(JVM) si Common Language Runtime sunt folosite pentru a virtualiza limbajul de programare Java si mediul de programare .NET Framework.[10]

4.1.2 VM bazate pe stiva si MV bazate pe registri

De multi ani, in arhitectura masinii virtuale, VM-urile bazate pe stiva erau alegerea arhitecturala datorata simplitatii implementarii VM si marimii executabilelor pentru arhitecturile pe stiva. VM bazata pe registri poate fi o alternative atractiva la VM bazata pe stiva datorita faptului ca numarul de instructiuni executate pe VM sa fie substantial redus. Un studiu care analizeaza VM diferite arata ca arhitectura bazata pe registri cere o medie de 47% mai putine instructiuni executate pe VM decat cele bazate pe stiva.Pe de alta parte codul registrilor este cu 25% mai mare decat codul corespunzator stivei, dar acest cost crescut al mai multor instructiuni VM datorate marimii codului mai mare implica numai 1.07% incarcari in plus pe masina reala per instructiune VM care sunt neglijabile.Performanta globala a VM bazata pe registri este aceea ca pe medie ia 32.3% mai putin timp pentru a executa standard benchmarks.

Datorita beneficiilor mentionate mai inainte, Google a ales VM bazata pe registri pentru platforma Android. In afara de problemele de licenta cu Sun. exista un numar de avantaje tehnice alegand VM bazata pe registri:
-VM bazata pe registri are avantaje de performanta intrinseca fata de VM bazata pe stiva cand e gazduita pe un procesor pipelined.
-Verificarea codului pe biti este adecvat pentru a fi mai rapid pe VM bazata pe registri (i.e,pornire rapida de mai multe ori) deoarece verificari ale integritatii inaltimii stivei vor fi mult simplificate.
- VM bazata pe registri va fi mai ingaduitoare cu codul incorect (in VM, generat de compilator, codul corupt in timpul transmisiei programului sau depozitului atacat de malware) decat VM bazata pe stiva.[10]

4.2 Masina virtuala Dalvik

Dalvik este o masina virtuala proiectata special pentru platforma Android. Numita dupa satul Dalvik din Islanda, a fost conceputa de Dan Bornstein.Spre deosebire de masinile virtuale bazate pe stiva,arhitectura Dalvik este bazata pe registri. Este optimizata pentru a folosi mai putin spatiu. Interpretatorul este simplificat pentru executie mai rapida.Executa propriul cod Dalvik pe biti decat cod Java pe biti.

4.2.1 Motivatia din spatele Dalvik

Toate sistemele mobile au caracteristici precum, putina RAM,CPU cu performanta redusa , memorie interna flash lenta, puterea bateriei limitata. Deci a fost nevoie de VM care sa poata furniza o performanta mai buna cu resurse limitate.Asa a aparut Dalvik, proiectata sa ruleze pe

nucleu Linux, care furnizeaza proces threading, prelucrand preliminar pentru executia mai rapida a aplicatiei, proceduri de securitate bazate pe User ID si comunicare intre procese. Dalvik lucreaza cu dispozitive ARM cu putine resurse (masini RISC avansate),are arhitectura de procesor pe 32 biti bazata pe calculator cu instruire redusa dezvoltate de ARM limitate). Procesoarele ARM sunt folosite din cauza arhitecturii lor simple ce le face potrivite pentru dispozitivele de putere joasa precum telefoanele mobile. Dalvik poate fi de asemenea rulata pe sistemele x86.

4.2.2 Atribute ale VM Dalvik

4.2.2.1 Arhitectura

Dalvik este o arhitectura bazata pe registri care face codul aplicatiei sa ruleze mai rapid si cu performanta eficienta. Trebuie sa opereza cu cod Dalvik pe biti decat cu cod Java pe biti.Vm bazata pe biti are avantaje potentiale fata de VM bazata pe stiva asa cum s-a discutat in sectiunea anterioara. Discutam acum functiile curente suportate:
-Format de fisier de executie Dalvik
-Multime de instructiuni Dalvik
-J2ME CLDC API
-Multi-threading

Platformele suportate sunt:
-Sistemele Baseline au intentia de a fi savoarea UNIX-ului
-Linux
-BSD
-Mac OSX

4.2.2.2 Runtime framework si biblioteci suportate

Runtime framework este un layer unde un programator incepe sa-si construiasca aplicatiile. Consta dintr-o masina virtuala care executa cod Dalvik pe biti(adica este construit din cod Java pe biti si nucleu API).

Telefoanele mobile din ziua de azi nu au suficiente resurse ca acelea ale unui PC, deci Java traditional (pentru ca solicita multe resurse) nu poate fi folosit pentru sisteme mobile, in general. Se cere prin urmare ca o multime de biblioteci cu functionalitati ale nucleului, optimizate pentru aceste dispositive cu capacitati limitate sa fie instalate.
IN cazul Dalvik , bibliotecile suportate includ:
-dalvik/libcore(programata in C/C++)
-dalvik/vm/native(programata inC/C++)
-OpenSSL (pentru criptare)
-zlib(gratuita, scop general, biblioteca,de compresare a datelor)
-ICU (pentru codificarea caracterului)
-pachete Java (incluzand java.nio,java.lang,java.util)
-Apache Harmony classlib (Inluzand Apache HttpClient)
VM Dalvik este scrisa in C portabil. Totusi are o componenta non-portabila a runtime-ului numita JNI call bridge .

4.2.2.3 LIcenta Dalvik

Dalvik si-a licentiat codul sursa sub licenta Apache, facandu-l atractiv si ieftin pentru purtatorii de telefoane mobile, deoarece o pot folosi si modifica fara sa plateasca taxa de licenta. In plus licenta Apache ii lasa sa includa orice parte a codului doresc. Comparativ cu masinile Java care sunt closed source facandu-le scumpe sau open source dar nu e in stilul afacerilor.

4 2.2.4 Probleme de securitate

Dalvik are layer suplimentar pentru securitate, incluzand separare de procese si permisiuni de accesare fisiere pe platforma Linux subiacenta. Dalvik este relativ sigura deoarece furnizeaza sharing de cod intre procese fara a da permisiuni de a modifica codul shared.[10]

4.3 Funcționarea și optimizarea VM Dalvik

4.3.1 Capacitățile unei VM tipice și probleme de interes

În cele ce urmează sunt descrise câteva probleme în perspectiva resurselor de sistem de care VM ar trebui să fie capabilă.

4.3.2 Memoria sistemului și timpul de procesare al sistemului

Mărimea minimă a memoriei pentru mașina virtuală ar trebui să se bazeze pe recomandările sistemului de operare. Deci partajarea codului pe biți (cunoscut de asemenea ca și clasa de date) este importantă pentru a evita folosirea în zadar a memoriei sistemului. Timpul de procesare al sistemului se referă la timpul de procesare cerut de software-ul sistemului care include sistemul de operare și orice utilitar care suportă programele aplicației. Unele aplicații fac dispozitivul insensibil, deci timpul total de procesare al sistemului trebuie redus.

4.3.3 Redundanța

Redundanța este importantă întrucât furnizează toleranță la erori, dar poate consuma foarte multă memorie deoarece solicită duplicarea mulțimii de date sau cod corector de erori care trebuie stocate în memorie. De obicei clasa de date este stocată în fișiere individuale, cauzând folosirea unei memorii prea mari și poate avea drept consecință redundanță mare, de exemplu, în stocarea șirurilor de caractere.[10]

4.3.4 Analizarea datelor

Analizarea verifică corectitudinea sintaxei și construiește structuri de date, mai ales arbore de analiză sau alte structuri ierarhice. La lansarea aplicației, clasele sunt încărcate în memoria sistemului și analizarea este făcută așa încât se adaugă o povară în plus la performanța sistemului.

4.3.5 Verificare

Verificarea codului pe biți este importantă pentru a verifica dacă integritatea unui fragment de cod este corectă. Deoarece sistemul Java run-time nu este conștient dacă codul este demn de încredere sau nu, dacă verificarea codului pe biți este făcută la lansarea unei aplicații va cauza încetinirea sistemului.

4.3.6 Optimizare

Pentru a salva viața bateriei, a crește performanța și a face sistemul să răspundă mai repede, Dalvik efectuează optimizarea codului pe biți. Optimizarea este importantă în cazul telefoanelor mobile deoarece au baterie și resurse limitate pentru efectuarea calculelor grele (comparativ cu PC-urile). Optimizarea este discutată în detaliu în următoarea secțiune.[10]

4.4 Dalvik optimizează și depășește problemele unei VM tipice

Pentru a depăși limitările memoriei, timpului de procesare al sistemului, redundanța și probleme similare, Dalvik efectuează optimizarea codului pe biți care implică următorii pași și care sunt în contrast cu reprezentarea capabilităților unei VM tipice.

4.4.1 Memoria sistemului

Pentru a minimiza folosirea memoriei sistemului, fișierele dex sunt read-only (în scopul securității care este dicutat mai tarziu) și este permisă de asemenea partajarea între procese. Aceasta evită repetiția de prisos a datelor și reduce folosirea memoriei.

4.4.2 Redundanța și spațiu

Dalvik depășește problema spațiului memoriei (consumată de redundanța și de fișiere separate pentru fiecare clasă) grupând multiplele clase într-un singur fișier dex. Aceasta salvează multă memorie pentru sistem.[10]

4.4.3 Timpul de procesare al sistemului

Timpul de procesare este datorat încărcării „Just-in-Time" a claselor în VM. Compilatoarele JIT au timp de procesare al memoriei deoarece au nevoie de un cod cache și structuri de date de sprijin. Aceasta ar putea chiar să facă sistemul insensibil în unele cazuri. Pentru a reduce timpul de procesare al sistemului, codul pe biți este optimizat ordonând codul pe biți la ajustarea alinierii cuvintelor înaintea lansării aplicației. Dalvik este deci optimizat pentru a executa multe instanțe concurente chiar în memoria limitată a unui telefon mobil.

4.4.4. Verificare

După cum am mai discutat mai înainte, verificarea codului pe biți este un proces lent, deci putem face procesarea rapidă efectuând „pre-verificarea" acestui cod pe biți. Aceasta reduce, de asemenea timpul de procesare care va fi obținut doar prin lansarea aplicației.

4.4.5 Analizare

După cum am mai discutat deja, analizarea adaugă povară extra la sistem, așa că codul pe biți este rescris înaintea timpului care este cerut prin metoda de optimizare. Este ca NO-JIT care face execuția mai rapidă.

4.4.6 Prezentare generală a operațiilor efectuate de Dalvik

Codul aplicației este în forma fișierelor /jar sau .apk ce conțin fișiere cu meta date. Clasele .dex sunt extrase din arhivă înainte ca ele să fie folosite. Instrumentul dx convertește codul Java pe biți în cod Dalvik pe biți. Sunt efectuate și alte operații precum realiniere, optimizare și verificare în codul Dalvik pe biți.

4.4.7 Detalii

Primul pas este să creăm un fișier dex. Metoda este declarată pentru pregătirea DEX urmată de optimizare.

1. Creăm un fișier Dalvik-cache în folderul Dalvik cache care este $ANDROID_DATA/data/dalvik-cache. Notă: Pentru a lucra, crea,șterge și/sau modifica fișiere în cache sunt cerute privilegii adecvate.
2. Intrarea classes.dex este extrasă dintr-o arhivă zip.
3. Pentru acces ușor și rapid în sistemul curent intervine memoria asociată care include schimbul de biți și realinierea structurii. În plus sunt făcute verificări pentru a asigura că indicii datelor și ieșirile fișierelor cad într-o categorie validă. Nota:Aceste proceduri nu fac schimbari logice fisierelor DEX.

4. In ODEX obtinut (fisierul optimizat DEX). Datele precalculate sant adaugate header-ului (la inceputul fisierului), aceasta asigura spatiul pentru header care va fi adaugat mai tarziu.
5. Acum revenim la procesul de verificare si optimizare. Dar inainte de asta trebuie sa cunoastem un program numit dexopt. Este un program care creaza un fisier odex dar elimina fisierul classes.dex in apk. Efectueaza o VM initializare, incarca fisierele DEX din clasa path si apoi seteaza verificarea si optimizarea.Folosirea sa va fi clara dupa ce discutam despre procesul de verificare si optimizare.
6. Toate aceste clase sunt verificate si optimizate in fisierul DEX incarcand toate clasele in VM
7. verificate sau optimizate ceea ce poate cauza o blocare a resurselor. Acest proces este facut mai degraba intr-o VM separata decat pe o VM pe care ruleaza aplicatia. Toata aceasta procedura este facuta automat de dexopt pe care l-am descris anterior;
8. Identificarea instructiunilor inegale inainte de run time este ceruta ceea ce implica verificare scanand toate instructiunile din fiecare metoda detinuta de fiecare clasa din fisierul DEX.
9. Urmeaza optimizarea .Un interpretator VM efectueaza de obicei optimizare petru prima bucata de cod care e utilizata si obtinuta inlocuind referinte pool constante cu pointeri la structurile de date interne. Operatiile care functioneaza intotdeauna sunt inlocuite de formele lor mai simple. Discutam in detaliu despre optimizatorul Dalvik:
10. Functionalitatea optimizatorului:
    -Metoda de inlocuire a indexului cu un index vtable(pentru apelarile metodei)
    -Inlocuirea campului index cu un byte offset (pentru campul instantei)
    -Inlocuieste apelari ce au high-volumul cu inlocuiri „inline”. (pentru string.length())
    -Stergerea oricaror metode vide .Vor fi apelate doar cand obiectele le sunt alocate
    -Efectueaza toate precalcularile datelor, va salva spatiul heap si calculeaza timpul de fiecare data cand fisierul DEX este incarcat.
11. Dependente ale ODEX :
    Fisierul DEX optimizat are dependente pe :
    -Toate celelalte fisiere DEX din clasa path a bootstrap-ului.
    -CRC-32
    -Modificarea datei din intrarea fisierului original classes.zip.
    Lista de dependente include:
    -Calea intreaga la fisierul dalvik-cache
    -Semnatura fisierului SHA-1
    -N umarul versiunii VM.
12. DEX generat si lansarea viitoare :
    Cu unele framework-uri verificarea dificila dexopt si optimizarea nu functioneaza bine, asadar ele trebuie sa se bazeze pe abilitatea de a genera cod pe biti si de a-l executa.Pentru aceasta va fi dat intr-o lansare viitoare.[10]

4.5 Compararea VM Dalvik cu masina virtuala Java

Performanta VM urilor variaza in functie de sistemul de operare subiacent si specificatiile si arhitectura hardware.Aici vom discuta VM Dalvik care este ales pentru sistemele Android si vom face o scurta comparatie cu JVM.

4.5.1 Comparatia folosirii memoriei

JVM consuma majoritatea resurselor sale pe garbage collection si nu poate elibera mult din memorie cand se cere, i.e., cand este aruncata exceptia „Out of Memory”.Marimea Heap(spatiul memoriei unde se gasesc programele) este calculata potrivit memoriei fizice. JVM foloseste o portiune mare

de memorie pentru bibliotecile runtime create in memoria partajata. Comparate cu Dalvik, ea opereaza cu propriul cod pe bit care este optimizat pentru cerinte de minime de memorie. De asemenea, formatul fisierului Dalvik permite executie Dalvik directa . Dalvik are un layer aditional pentru securitate incluzand separare a proceselor si permisiuni ale fisierelor la platforma Linux subiacenta. VM Dalvik este securizata deoarece furnizeaza partajare de cod intre procese fara sa dea permisiuni sa editeze codul partajat.Fisierul asociat in memorie nu necesita crearea unei incarcari a structurilor de date doar pentru a fi capabil sa fie incarcat.[10]

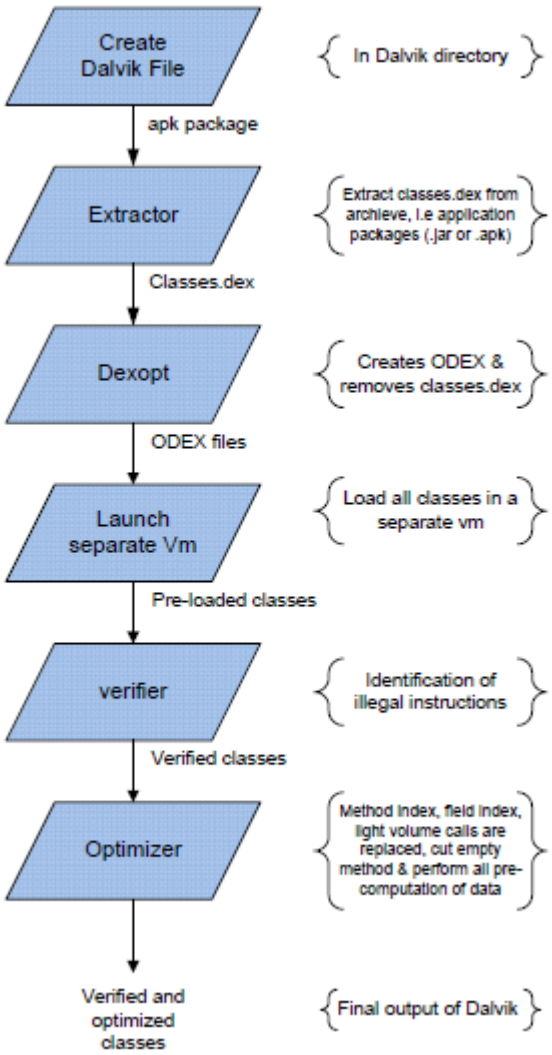

Fig.4.16 Reprezentarea fluxului Dalvik

4.5.2 Compararea arhitecturii

JVM este bazata pe stiva si opereaza cu cod Java pe biti.Arhitecturile bazate pe stiva sunt mai lente deoarece stiva este stocata in memorie. Spre deosebire de alte VM-uri care sunt bazate pe stiva, arhitectura Dalvik este bazata pe registri. Dalvik ar putea sa fie o VM foate interesanta chiar pentru folosirea pe desktop nu doar pe dispozitivele mai mici.Masinile bazate pe registri pot fi mai rapide decat cele bazate pe stiva.Mai putine accesari ale memoriei cresc de asemenea viteza unei masini bazate pe registri si furnizeaza multe oportunitati pentru executie paralela in timpul executiei scalarilor.

4.5.3 Instante multiple si comparatie JIT

JVM ruleaza implicit pe JIT. Aceasta face incarcarea aplicatiei mult mai lenta pentru dispozitivele cu resurse limitate cum ar fi telefoanele mobile. Dalvik este optimizata sa permita instante multiple pentru a le executa simultan chiar cu putina memorie. Motivul principal pentru care este optimizare ahead-of-time este acela ca JIT este costisitoare si imprevizibila atunci cand este necesara executia eficienta cu resurse limitate.

4.5.4 Compararea fiabilitatii

In sistemele standard Java runtime esecul unei singure componente poate avea impacte semnificative asupra altor componente. In cel mai rau caz o componenta daunatoare sau eronata poate sa distruga intregul sistem. Pe de alta parte Dalvik executa fiecare instanta in propriul proces separat. Procesele separate previn toate aplicatiile de distrugere in cazul in care VM se distruge.

4.5.5 Comparare intre biblioteci suportate

Urmatorul tabel descrie biblioteci diferite suportate de VM Dalvik si VM Java .

Libraries	Dalvik	Standard Java
java.io	Y	Y
java.net	Y	Y
android.*	Y	N
com.google.*	Y	N
javax.swing.*	N	Y
...	...	...

Tabel 4.2 Comparatie –biblioteci suportate de VM Dalvik si VM Java

4.5.6 Rezultatele compararii

Urmatorul tabel prezinta aspectele diferite ale ambelor VM

Criteria	Dalvik	JVM
Architecture	Register-based	Stack-based
OS Support	Android	All
Reverse Engineering-tools	a few(dexdump,ddx)	many (jad,bcel,findbugs,...)
Executables	DEX	JAR
Constant-Pool	per application	per class

Tabel 4.3 Aspecte diferite ale VM-urilor

CAPITOLUL V

Aplicatii Android

5.1 Advanced Task Killer

Advanced Task Killer sau, pe scurt, ATK, este o aplicaţie pe care aproape toţi utilizatorii de smartphone-uri cu Android o folosesc pentru că **rezolvă problema consumului de memorie RAM**, când sunt mai multe aplicaţii deschise simultan.

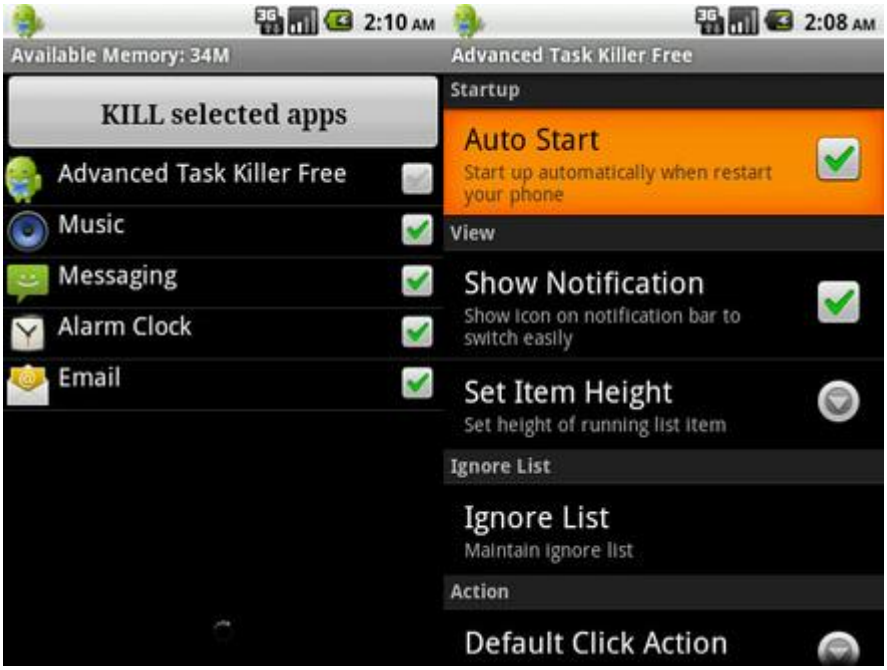

Fig.5 17 Advanced Task Killer

5.2 Gas Buddy

Gas Buddy este foarte apreciat de către persoanele care şofează în Statele Unite şi Canada. Programul **indică pe hartă cele mai apropiate benzinării,** dar oferă şi informaţii legate de preţurile carburanţilor utilizatori, membri ai comunităţii Gas Buddy.[11]

Fig.5 18 Gas Buddy

5.3 WhatsApp Messenger

Cu acest program **se pot trimite SMS-uri gratuite** prietenilor sau se poate conversa pe chat. Aplicaţia suportă inclusiv transfer de fişiere multimedia (imagini, video şi audio). Singura condiţie este ca şi destinatarul să aibă instalat WhatsApp Messenger.Există şi versiuni pentru platformele iOS, BlackBerry OS şi Symbian.

Fig 5.19 WhatsApp Messenger

5.4 Flash Player 10.2

Ultima versiune de Flash este recomandată celor care doresc o **experienţă Web completă.** Flash-ul consumă multe resurse, iar posesorii de dispozitive Android cu putere de procesare mai slabă vor întâmpina oarecare probleme.

5.5 Facebook for Android

. Spre deosebire de versiunile pentru telefoane simple, Facebook for Android se prezintă într-un format foarte apropiat de versiunea pentru browserele web PC / Mac / Linux.

Fig.5.20 Facebook for Android

5.6 YouTube

Folosind aplicaţia dedicată se **economiseşte traffic** , **interfaţa este optimizată** pentru ecranul telefoanelor mobile.

5.7 Kakao Talk

Aplicaţia suportă group chat cu un număr nelimitat de utilizatori, avertizează utilizatorii inclusiv vocal dacă primesc un mesaj şi permite transferul de fişere foto şi video[11]

Fig.5.21

5.8 Vodafone Guardian

Este o aplicatie creata pentru a bloca comunicatiile nedorite de pe telefonul mobil si pentru a restricționa accesul la internet sau alte functii ale telefonului.

Aplicatia a fost create pentru parinti cu scopul de a-I ajuta sa le ofere copiilor acces la un smartphone intr-omaniera responsabila si sigura pentru a preveni activitati de hartuire sau intimidare da catre personae necunoscute.

Cu Vodafone Guardian se pot seta atat reguli pentru apeluri si mesaje cat si pentru functionalitatile telefonului sau chiar pentru alte aplicatii.Pot fi si :

- Reguli pentru apeluri primate SMS-uri sau MMS-uriprimite
- Reguli pentru persoanele care nu sant in agenda telefonului si apelurile primite de la numerele private
- Reguli pentru functiile telafonului:WI-FI, Bluetooth, camerta foto, browser, setari etc.
- Reguli de functionare pentru aplicatiile instalate pe telefon, precum si restrictii clare pentru adaugarea si dezinstalarea aplicatiilor, atat di magazinul de aplicatii Google Play cat si din surse externe
- Aplicatia beneficiaza de trafic gratuit in retea

5.9 Free Music Downloade

Free Music Downloader foloseşte un motor de căutare care dă posibilitatea de a **descarca gratuit milioane de MP3-uri**. Aplicaţia dispune de un organizator pentru fişiere, dar şi de editare a acestora.[11]

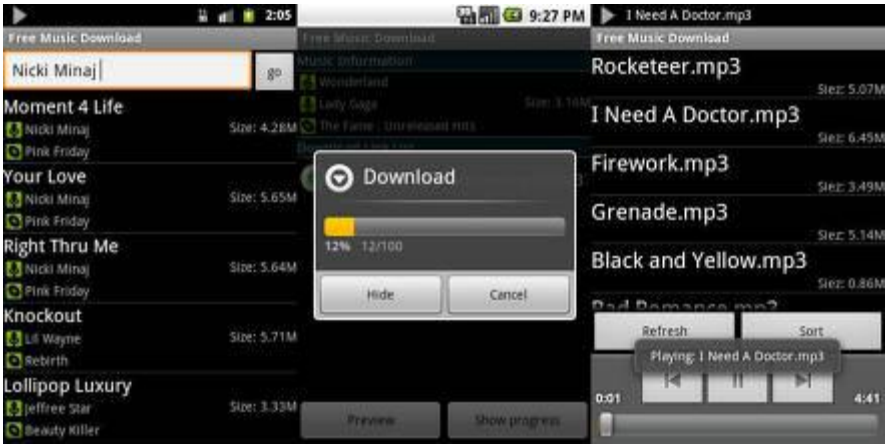

Fig 5.22 Free Music Downloade

CAPITOLUL VI

Concluzii

Platformele open source au castigat atentia utilizatorilor terminali datorita licentelor lor gratuite si sursei modificabile, a devenit o provocare sa concureze cu celelalte platforme care au castigat deja popularitate pe piata si care au castigat increderea producatorilor, dezvoltatorilor si utilizatorilor terminali.. Android este prima platforma Open Source gratis si personalizabila.Ofera o stiva intreaga ce include un sistem de operare, middle ware, aplicatii mobile cheie. De asemenea contine o multime bogata de API-uri care permit dezvoltatorilor tertiari sa dezvolte aplicatii utile.

Dalvik este masina virtuala proiectata sa lucreze pe sistemele de operare Android. Cand discutam despre platformsa Open Source primul lucru care ne vine in minte este Linux, deoarece este sursa gratis ce suporta multi-tasking, process threading, securitate si abilitate de a rula pe un sistem cu resurse limitate. Acesti factori au devenit motivatia pentru ca Android sa fie construit pe nucleul Linux. Am inclus anumite fapte si figuri care arata cum piata Android castiga atentia pietii mobile. Am inclus unele lucruri de baza despre masinile virtuale si apoi despre VM Dalvik in detaliu S-a discutat despre codurile pe biti, conversia de la codul Java pe biti la codul Dalvik pe biti, functionarea Dalvik si multe alte aspecte ale VM Dalvik care ar putea fi importanta cititorului.

A fost discutata comparare masinii virtuale Dalvik cu Java asa incat a putut fi prezentata o imagine clara pentru a intelege in ce mod Dalvik este mai buna decat JVM. Din acest studiu va fi usor cititorului sa traga o concluzie despre faptul ca VM este deschisa, de performanta eficienta si mai stabila.

Am furnizat o imagine adanca a codului pe biti pe care Dalvik il foloseste pentru operatiile sale..Am furnizat informatii pentru a intelege ca MV Dalvik poate genera idei cercetatorului sa faca Dalvik mai eficienta si mai rapida si pentru a genera codul Dalvik intr-un mod in care se incarca mai repede, deci facand executia aplicatiei chiar mai rapida.

Din comparatia intre fisierele dex si cele Java comprimate si necomprimate putem observa cat de putin spatiu al memoriei consuma.Executia unui cod pe biti clar il va face mai rapid decat inante. Un studiu de cerecetare pentru a aranja codul mai bine si ca sa fie mai usor executat de VM poate sa aduca o mare performanta reactualizarii.[10]

Viitorul aparţine Android

În februarie 2011, platforma a făcut un enorm pas înainte prin lansarea versiunii 3.0 Honeycomb optimizată pentru tablete: interfaţa de utilizator, aplicaţiile Google şi ecranul de pornire sunt optimizate pentru formatul şi noile posibilităţi ale tabletelor.Evoluţia continuă, Google, OHA şi dezvoltatorii de aplicaţii vor optimiza şi în continuare sistemul de operare şi îl vor adapta continuu la necesităţile ce apar – aceasta fiind singura modalitate prin care o platformă devine competitivă pe termen lung.[2]

Înfăţişarea sistemului de operare nu este aceeaşi – nici măcar cu aceeaşi versiune a sistemului de operare: aproape toţi producătorii oferă dispozitivelor pe care rulează sistemul Android o interfaţă proprie de utilizator.

Sistemul de operare în stare pură se întâlneşte foarte rar, dintre modelele mai noi amintim Google Nexus One, Motorola Milestone care nu mai prea este de actualitate, precum şi Samsung Nexus S. Sistemul de operare deschis oferă producătorilor posibilitatea de a-şi dota telefoanele pe care rulează Android cu interfeţe de utilizator, aplicaţii şi servicii proprii. Ca urmare, un smartphone de la HTC pe care rulează sistemul Android nu se deosebeşte doar din punct de vedere al mărcii, de exemplu, de un telefon Sony Ericsson pe care rulează Android, ci diferenţe se pot observa şi când se aruncă o privire asupra display-ului – chiar şi în cazul în care este vorba despre două smartphone-uri pe care rulează aceeaşi versiune a sistemului de operare Android.[1]

Prin intermediul diferitelor faţade, producătorii doresc să se departajeze de concurenţă. Utilizatorii trebuie să se identifice mai mult cu dispozitivul şi mai puţin cu sistemul de operare. Bineînţeles, conceptul de operare rămâne acelaşi la toate modelele pe care rulează Android: cu ajutorul gesturilor intuitive pe touchscreen, înregistrările meniului sunt parcurse rapid; pe ecranele de pornire se pot poziţiona widget-uri, scurtături directe, foldere şi link-uri favorite în funcţie de dorinţe, existând apoi şi posibilitatea mutării sau ştergerii acestora de pe ecranele de pornire. Astfel smartphone-urile pe care rulează Android sunt adevăraţi maeştri la capitolul personalizare, iar prin intermediul interfeţelor proprii ale producătorilor această artă este rafinată.[1]

În special HTC, cu ajutorul interfeţei de utilizator HTC Sense şi scenele predefinite a creat o posibilitate elegantă de a adapta smartphone-urile pe care rulează Android la necesităţile din rutina zilnică, ca de exemplu cu scene presetate pentru timpul petrecut la birou şi scene pentru după încheierea programului de lucru. Practică este şi afişarea sub formă de carusel sau elicopter, introdusă de HTC şi apoi preluată de mulţi alţi producători: prin apropierea degetului mare şi a arătătorului pe unul dintre ecranele de pornire, pot fi vizualizate simultan toate ecranele de pornire în miniatură şi pot fi selectate direct. Plusvaloare oferă şi aplicaţiile individuale, precum HTC Friendstream, o aplicaţie care este lansată şi sub forma unui widget şi care centralizează statusurile de pe Facebook, Twitter şi Flickr într-un singur flux de actualizări.

Una dintre priorităţile Motorola o reprezintă relaţiile interumane: serviciul Motoblur poziţionează statusurile şi mesajele, cu ajutorul widget-urilor, direct pe ecranul de start, astfel încât este oferită o privire de ansamblu rapidă. La acest capitol acţionează exemplar şi Sony Ericsson cu Timescape: aplicaţia integrată în interfaţa de utilizator colecţionează toate rezultatele noi şi le afişează pe ecran, în ordine cronologică, prin intermediul unui widget cu aspect 3D elegant. Samsung şi-a denumit interfaţa de utilizator Touchwiz şi pe lângă widget-uri suplimentare oferă şi un task manager pe smartphone-urile sale pe care rulează Android. Comparativ cu interfaţa HTC Sense, la modelele Samsung cu Touchwiz meniul principal poate fi personalizat individual; aplicaţiile instalate nu trebuie neapărat sortate în ordine alfabetică sau cronologică, ci pot fi ordonate după dorinţe.

LG mai face un pas înainte cu interfaţa Optimus: aici, în cadrul meniului principal, nu se poate alege liber doar ordinea aplicaţiilor, ci se pot defini mai multe categorii, astfel încât aplicaţiile instalate se pot sorta şi poziţiona în funcţie de tematică în meniu. Acer a găsit o modalitate oarecum nonşalantă şi împarte interfaţa proprie numită Acer UI, prin structurarea pe diferite nivele, ceea ce la început mai degrabă complică operarea decât să o uşureze.[1]

Bibliografie

1. <ro.wikipedia.org/wiki/Android_(sistem_de_operare)>
2. <(www.connect.ro/.../ android-totul-despre-sistemul-de-operare-google/>
3. <(*www.realitatea.net/istoria-**sistemului-de-operare-android**-de-la-ecler-l...)*>
4. <www.androider.ro./.../prezentarea_inglobarea_sistemului_de_operare_...>
5. < http://mobilephones.pk/reviews/what-are-capacitive-and-resistive-touch-screens/>
6.<*smilingmouse.ro/curs-2-**arhitectura-sistemului-android**/*>
7. <www.itcsolutions.eu/.../android-tutorial-concepte-activitati-si-resurse-....>
8. <www.scribd.com/doc/70765152/33/Masina-virtuala-Dalvik>
9. *<http://en.wikipedia.org/wiki/Rooting_(Android_OS)>*
10.*<imsciences.edu.pk/.../Analysis-of-**Dalvik**-V..>*
11.*<_www.go4it.ro/.../top-**aplicatii-android**-2011-cele-mai-folosite-si-apre...>*