Stephan Zeisberg

NEXT GENERATION MOBILE PHONE PLATFORMS

Ein Einblick in die Systemarchitekturen aktueller Smartphones

Motivation

Technologischer Stillstand in der Entwicklung mobiler Betriebssysteme

- unzureichende Unterstützung mobiler Datendienste
- schlecht bedienbare Geräte (kleine Displays, Übertragungsgeschwindigkeit)
- geringer Funktionsumfang
- gestiegene Anforderungen aufgrund neuer Hardware

Next Generation Mobile Phone Platforms

- leichte Bedienbarkeit
- höhere Individualisierung
- Verstärkung der mobilen Onlinenutzung
- hoher Funktionsumfang

T-Mobile G1 & iPhone 3G & Neo Freerunner

[Bildquelle(mitte):http://www.apple.com/iphone]
[Bildquelle(links)://www.portel.de/fileadmin/pics/T-Z/T-Mobile-G1-1-09.jpg]
[Bildquelle(rechts):http://mobile-place.info/home/images/stories/vijesti/openmoko/neo-001.jpg]

T-Mobile G1 & iPhone 3G & Neo Freerunner

- Hardware der Smartphones

	Neo Freerunner	T-Mobile G1	iPhone 3G
Arbeitsspeicher	128 MB DDR SDRAM	192 MB DDR SDRAM	128 MB DRAM
CPU	ARMv4 , 400Mhz	ARM 1136, 528Mhz	ARM 1176, 400Mhz
Maße(HxBxT) in mm	120 x 62 x 18.5	115 x 55 x 16	115 x 61 x 11.6
Bildschirm	2.84-Zoll Touchscreen	3.2-Zoll Touchscreen	3.5-Zoll Touchscreen
Funkverbindungen	WiFi 802.11b/g, Bluetooth, GSM/EDGE, UMTS/HSDPA, GPS	WiFi 802.11b/g, Bluetooth, GSM/EDGE, UMTS/HSDPA, GPS	WiFi 802.11b/g, Bluetooth, GSM/EDGE, UMTS/HSDPA, GPS
Gewicht	185g	158g	135g

Smartphones - Allgemein

Apple iPhone 3G

- Betriebssystem **iPhone OS**
  - 4-Schichtenarchitektur(Core OS, Core Services, Media, Cocoa Touch)
  - hybrider Kerneltyp
  - Entwicklung unterliegt Apple

T-Mobile G1

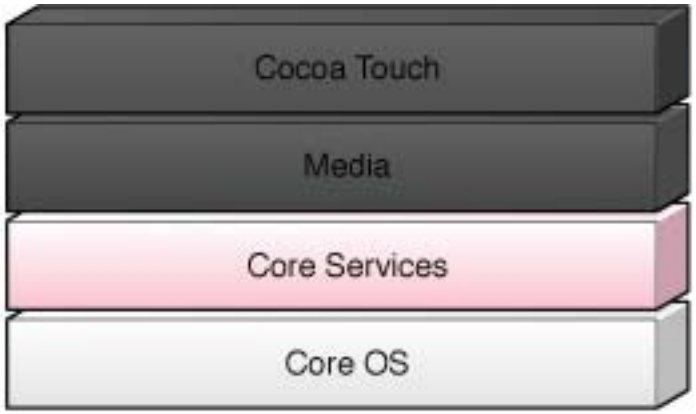
- Betriebssystem **Android OS**
  - 5-Schichtenarchitektur(Kernel, Libraries, Android Runtime, Application Framework, Applications)
  - monolithischer Kerneltyp
  - es erlaubt Entwicklern Anwendungen in Java zu schreiben

Neo Freerunner

- Betriebssystem **Openmoko**
  - 4-Schichtenarchitektur(Kernel, Core Services, User Interface, Application Framework)
  - monolithischer Kerneltyp
  - Softwareentwicklung erfolgt mit den in der Linux-Welt bekannten Sprachen

iPhone OS – Systemarchitektur Spezifikation

- iPhone OS System-Stack-Diagramm
  - für Systemarchitektur ausschlaggebend sind unteren 2 Ebenen



- beinhalten die fundamentalen Schnittstellen
  - dies sind C-basiert
  - darunter z.B. Netzwerkdienste, Bonjour-Services

[Bildquelle:http://developer.apple.com/iphone/gettingstarted/docs/iphoneosoverview.action]

iPhone OS – Systemarchitektur Spezifikation

- die Kernelumgebung

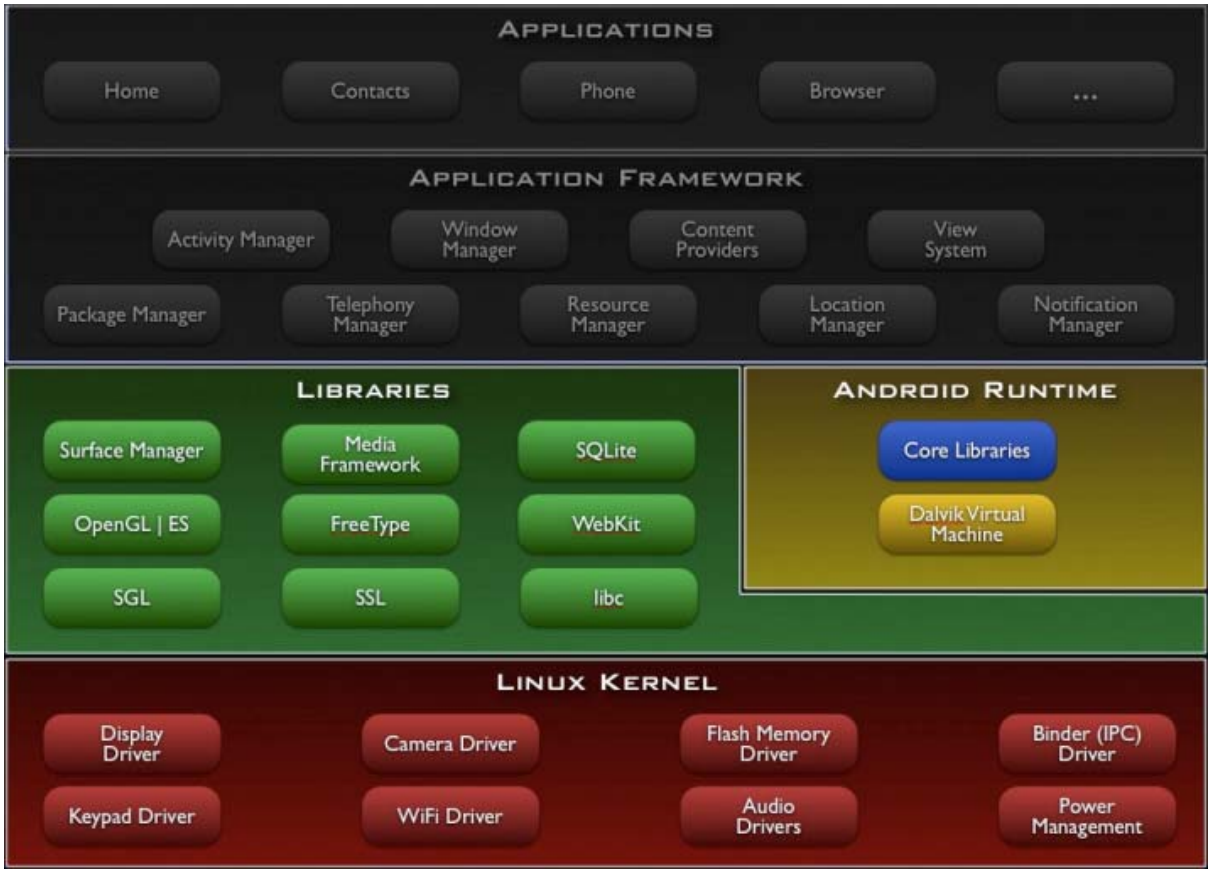
- darin befinden sich Treiber und Basisschnittstellen
- Kernel ist XNU-Kernel
  - hybrid Kernel, FreeBSD + Mach-3.0
    - FreeBSD : Rechte-Multiusermanagement, Prozessaufteilung, TCP/IP, Synchronisierung
    - Mach-3.0 : Speichermanagement, Multitasking, Debugging
- weitere Aufgaben:
  - Dateisystem- sowie Netzwerkverwaltung
  - interne Prozesskommunikation

iPhone OS – Core Services

- stellt fundamentale Systemdienste für Applikationen bereit
- unterteilt in Frameworks
    - Adress Book Framework
        - beinhaltet Kontaktinformationen, eigenes UI
    - Core Foundation Framework
        - Dateimanagement, Servicefunktionen
    - CFNetwork Framework
        - Netzwerkkommunikation(FTP, HTTP)
    - Security Framework
        - Datenschutz
    - Core Location Framework
        - Positionsbestimmung
- Entwickelt wird in Programmiersprache Objective-C
    - objektorientierte Erweiterung der Sprache C, angelehnt an Smalltalk

Android – Systemarchitektur Spezifikation

- System-Stack-Diagramm beschreibt die Android Systemarchitektur



[Bildquelle:http://code.google.com/android/what-is-android.html

Android – Linux Kernel

- modifizierter Linux Kernel
  - Grundlage ist Kernel 2.6.24
  - Warum Linux Kernel?
    - immenser Funktionsumfang + Linux Kernel ist Opensource
    - ausgereiftes Speicher- und Prozessmanagement
    - bewährtes Treiber- und Sicherheitsmodell
  - Modifizierungen:
    - keine GNU libc Unterstützung
    - Alarmfunktion, Low-Memory-Killer, Kerneldebugger
    - erweitertes Energiemanagement

Android – Libraries

- Bionic Libc
  - nicht kompatibel zur GNU libc Bibliothek
  - Geschwindigkeitsverbesserungen
- Function Libraries
  - □(SQLite) embedded optimierte Datenbank
  - (Webkit) Open Source Browser Grundlage
  - (Media Framework) Unterstützung für Audio/Video
- Surface Manager
  - Surface Flinger regelt Applikationszugriff auf das Display
  - Audio Flinger regelt den Output von Audiogeräten
- Hardware Abstraction Libraries
  - definieren Schnittstellen, welche Android verpflichten Hardwaretreiber zu implementieren

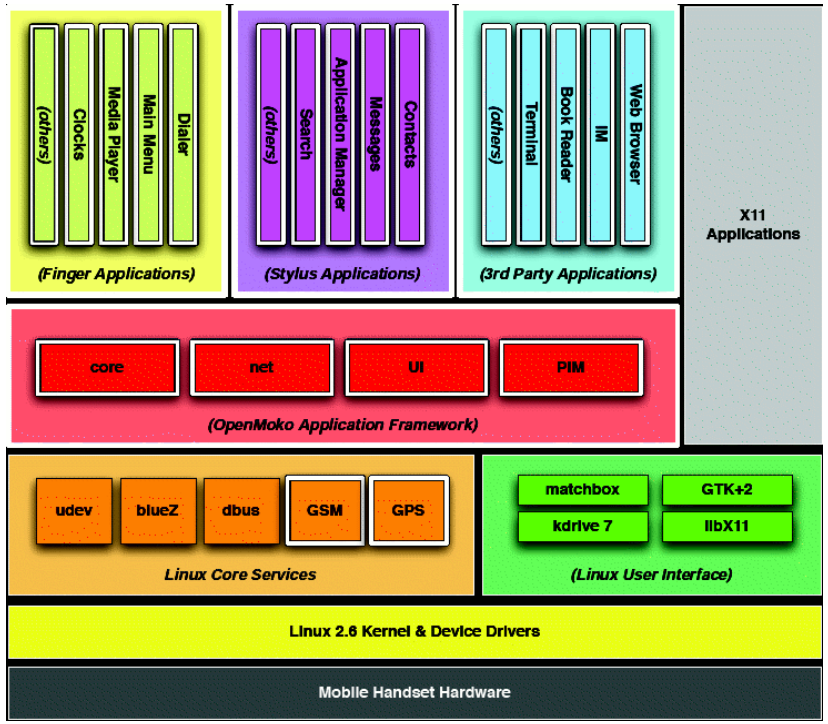
Android – Android Runtime

- Core Libraries
  - enthalten Programmierschnittstellen für Java-Programme
  - diese unterstützen Entwicklungshilfen für z.B. Daten- und Netzwerkzugriff

- Dalvik Virtual Machine
  - Alle Applikationen laufen auf dieser virtuellen Maschine
  - führt Java Anwendungen aus
  - wandelt Java Applikation in .dex um
  - .dex ist embedded optimiert
  - Programmiersprache zur Applikationserstellung ist Java

Openmoko – Systemarchitektur Spezifikation

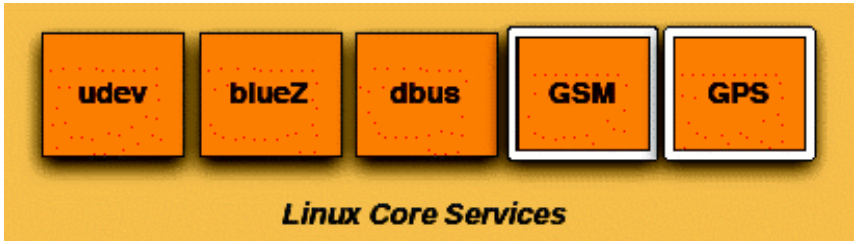
- Openmoko wurde von der Openmoko Inc. entwickelt
  - Besonderheit:
    - Softwareplattform die komplett unter Lizenzen steht, die als „freie Software" gelten
    - auch Hardware Open-Source
      - Schaltpläne
      - Gehäuse CAD-Zeichnungen
- System-Stack-Diagramm



[Bildquelle:http://www.linuxdevices.com/files/misc/openmoko_2007_software_stack.jpg]

Openmoko – Linux Kernel & Core Services

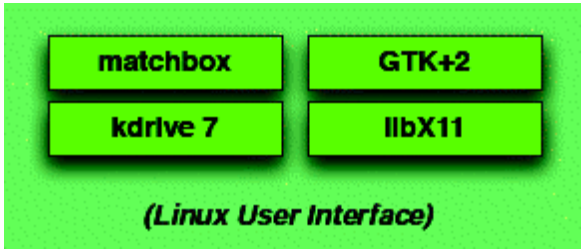
- Vanilla 2.6.21.3 Linux Kernel
  - embedded optimiert
  - erweitert durch z.B. USB-, Touchscreen- sowie SD-Unterstützung

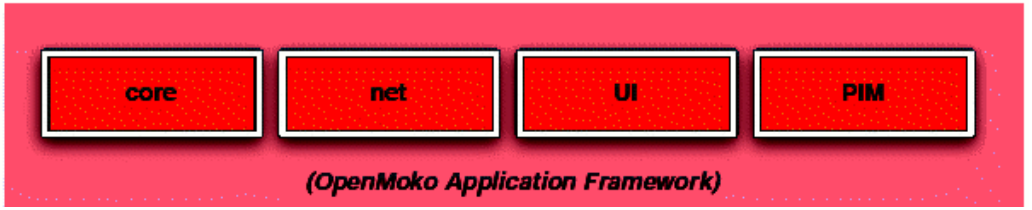


- Ebene für Systemdienste
  - Module und Daemons für GSM und GPS
  - Dbus System(ipc)
  - Gerätemanager
  - Bluetooth

Openmoko – User Interface & Application Framework

- Das User Interface
  - Windowmanager matchbox
  - Audiofunktionalität gewährleistet durch ALSA
  - Gimp Toolkit, Bibliothek zur Erstellung einer GUI





- Application Framework
  - libmokocore
    - senden von Nachrichten zwischen Applikationen
    - Speicherung von Konfigurationsdaten
  - libmokoui
    - regelt das einheitliche Aussehen der Applikationen
  - PIM
    - Kontaktdatenverwaltung
  - libmokonet
    - Schnittstelle für Netzwerkverbindungen über Bluetooth und GPRS

Systemarchitektur Vergleich

- Kernel Vergleich
  - Android und Openmoko benutzen aktuelle Linux Kernel
    - beide Hersteller optimierten den Linux Kernel
    - Kernel ist Open Source
    - Version 2.6 wurde für embedded Betrieb verbessert
    - Linux Kern bietet bereits integrierte Treiber
  - iPhone OS verwendet hybriden XNU Kernel
    - leicht erweiterbar aufgrund seiner Struktur
    - Kernel Aufbau bringt hohe Geschwindigkeit
  - Linux Kernel ist monolitisch, kann somit Vorteile eines Mikrokernels nicht nutzen
  - Mikrokernel ist langsamer

Systemarchitektur Vergleich

- Vergleich Programmierung
  - iPhone OS verwendet Objective-C → weitreichend unbekannt
  - Android benutzt Java, allerdings mit Dalvik Bytecode
  - Openmoko unterstützt in Linux-Welt bekannten Programmiersprachen(Python,C,..)
  - Alle Hersteller stellen ein SDK zur Verfügung

- Libraries
  - Alle Plattformen wurden ähnlich ausgestattet(SQLite, Webkit)
  - Bibliotheken wurden optimiert für den Einsatz auf Smartphones

Systemarchitektur Vergleich

- Vergleich der Ziele
  - Openmoko Projekt und Konzept hebt sich von Android und dem iPhone OS ab
    - Alle Bestandteile des Betriebssystems sind quelloffen
    - Neo Freerunner Schaltpläne sowie CAD-Zeichnungen wurden veröffentlicht
    - Fokus liegt auf benutzereigener Konfigurierbarkeit
  - Android auf ersten Blick ähnliche Ziele wie Openmoko
    - keine grenzenlose Offenheit wie bei Openmoko
    - Applikationsentwicklung nur in Java(in Verbindung mit DVM)
    - unter dem Android Open Source Projekt wurde Kernel offengelegt
    - Google und die OHA wollen Online Applikation publik machen
  - iPhone OS ist Gegenpart der anderen Plattformen
    - Systementwicklung unterliegt allein Apple
    - entwicklerunfreundlich, nur in Verbindung mit Macintosh Computer möglich

Zusammenfassung

- Allgemein
  - Modifizierung bereits vorhandener Software(Kernel, Libraries)
  - Optimierung der Systembestandteile für den embedded Einsatz(z.B. Geschwindigkeit)
- OS X iPhone
  - iPhone OS ist ein Übertrag von Mac OS X auf ein eingebettetes System
  - hybrid-Kernel
  - abgeschottete Entwicklung
- Android
  - hohe Hardwarekompatibilität durch Systemaufbau
  - teilweise quelloffen
- Openmoko
  - Open-Source Philosophie
  - unausgereifte Software

Ende

Danke!!

Fragen?