Next Generation Mobile Phone Platforms

Stephan Zeisberg
Computer Systems and Telematics
Institute of Computer Science
Freie Universität Berlin, Germany
zeisberg@mi.fu-berlin.de

***Zusammenfassung*—In der Mobilfunkbranche waren lange Zeit Hersteller wie RIM oder Nokia alleinige Marktführer und technologische Spitzenreiter. Meine Arbeit beschäftigt sich mit den Next Generation Mobile Phones und wird diese aus technischer Sicht miteinander vergleichen. Es werden sowohl die Hardware als auch die System-Architekturen behandelt. Einleitend werde ich die drei zu vergleichenden Betriebssysteme iPhone OS, Android, OpenMoko und die dazugehörigen Mobiltelefone, vorstellen. Dazu ist eine spezielle Sicht auf die Hardware Spezifikationen, welche einen tieferen Einblick in den Aufbau der Systemarchitekturen ermöglichen, nötig. Die jeweiligen Kernel und System-Bibliotheken werden weitreichend erläutert. Anschließend werden die spezifizierten Architekturen miteinander verglichen und bewertet. Darauf folgt eine inhaltliche Zusammenfassung sowie ein Ausblick auf die zukünftigen Entwicklungen in der Mobilfunkbranche.**

I. Einleitung

Vor nicht allzu langer Zeit, war der Erwartungswert an ein Mobiltelefon nicht sehr hoch angelegt. Alles was man wollte, war ein Gerät mit dem man mit anderen Menschen telefonisch kommunizieren konnte.

Heutzutage rufen Mobilfunkhersteller wie Sony Ericson und Nokia oder auch Netzwerkbetreiber wie T-Mobile immer mehr Kunden dazu auf, Mobiltelefone zu kaufen, welche eine größere Bandbreite an Funktionen anbieten. Diese Geräte sollen Digitalkamera, persönlicher Organizer und internetfähig in einem sein.

Der Antrieb dafür ist eine Kombination von immer weiterentwickelteren Hardwarekomponenten, welche eine bessere Prozessor- und Speicherkapazität bieten sowie ständige Geschwindigkeitsverbesserungen im Bereich der drahtlosen Verbindungen.

Dies eröffnet den Herstellern von Mobilfunkgeräten viele Möglichkeiten bessere Produkte auf den Markt zu bringen.

Aus dem Grund, dass sich Hardware immer schneller entwickelte, erreichten die Betriebssysteme der mobilen Endgeräte ihre Limits. Diese mussten also einen weitaus größeren Funktionsumfang unterstützen können. Das Symbian OS setzte sich schnell als ein System durch, welches dies leisten konnte. Marktführende Unternehmen wie Nokia, Motorola oder Sony Erricson erkannten das und nutzten die Plattform für ihre Geräte. Dadurch erhielt das Symbian OS, bis heute, eine Vormachtstellung auf dem Markt für mobile Betriebssysteme.

Doch der immer noch zu geringe Funktionsumfang des Symbian OS, lässt das Betriebssystem seine Grenzen erreichen. Das auf einen Microkernel basierende Betriebssystem, findet man in verschiedene Version für unterschiedliche Mobiltelefone.

Das andauernde Wachstum der Branche sowie der Drang dazu, etwas neues zu entwickeln, ist wahrscheinlich der Grund, weshalb sogennante Next Generation Mobile Phones Platforms auf den Markt auftreten.

Unternehmen wie Apple oder Google setzen sich das Ziel mit neuen Konzepten zu überzeugen und somit Anteile vom Symbian OS beherrschten Markt für sich zu gewinnen.

Next Generation bedeutet, dass etwas bereits existierendes, in dem Fall das Mobiltelefon, neue Funktionen erhält. Das Unternehmen Apple machte dabei mit der Entwicklung des iPhones den Anfang.

Google und die Open Handset Alliance zogen 2008 mit dem T-Mobile G1 und der, sich darauf befindenden, Android Plattform nach.

Es stellt sich die Frage, was Mobiltelefone, wie das T-Mobile G1 oder das iPhone von anderen unterscheidet, vorallem in Bezug auf deren Systemarchitektur. Ebenso gilt es die Ansätze anderer Hersteller, beispielsweise Samsung, RIM oder Nokia, herauszuarbeiten.

II. Grundlagen

A. Begriffserklärungen für diesen Abschnitt

1) Kernel: Der Systemkern (engl. kernel) stellt die wichtigsten Funktionen eines Betriebssystems bereit. Alle auf dem Kernel aufbauenden Softwarebestandteile unterliegen dem im Systemkern festgelegten Prozess und Dateimanagement.

Abbildung 1. Das Apple iPhone 3G[1], das T-Mobile-G1[2], der Neo Freerunner[3]

2) Library: Ein Software Library bezeichnet eine Programmbibliothek in der Programmfunktionen zusammengefasst werden. Die Libraries dienen dem Zweck bestimmte Aufgaben zu erfüllen. Allerdings sind diese Funktionen als eine Art Hilfsmodul zu verstehen und nicht eigenständig lauffähig, im Gegensatz zu einem Programm.

3) Framework: Ein Framework fungiert als Gerüst für den Softwareentwickler, welcher eine Applikation erstellt. Der Aufbau der Anwendung unterliegt also der vorgegebenen Struktur des Frameworks.

B. Apple iPhone 3G Beschreibung

Das iPhone ist ein Produkt der Apple Inc. und wurde am 9. Januar 2007 auf der Macworld Conference & Expo offiziell vorgestellt. Die Firma aus Cupertino entwickelte das iPhone, da die Zahl der Mobilfunknutzer stetig anstieg und die Apple Software, wie zum Beispiel iTunes[1], nur auf Mobiltelefonen lief, die nicht Apples Produklinie entsprachen.

Am 29. Juni 2007 begann der Verkauf des iPhones der ersten Generation offiziell in den USA. Ein Jahr später folgte das iPhone der zweiten Generation mit der Bezeichnung iPhone 3G, welches nun zum 3G Netz kompatibel war.

Das iPhone 3G lässt sich über einen 3.5-Zoll großen Touchscreen bedienen und verfügt über keine physische Tastatur. Allein eine Home-Taste, ein Standby-Schalter, die Lautstärkeregler und ein Schalter für den Stummmodus sind die physischen Bedienelemente. [4]

Durch Sensortechnik macht Apple es dem Gerät möglich, energiesparend zu arbeiten. Zum Beispiel wird die Bildschirmhelligkeit automatisch verringert, wenn man das Gerät zum Ohr hin bewegt.

Des Weiteren verfügt das iPhone 3G über eine Zwei-Megapixel-Kamera, ebenso wie über einen 8 oder 16 GByte großen Speicher. Auf dem iPhone läuft das iPhone OS, ein von Apple entwickeltes Betriebssystem, welches ausschließlich in Zusammenhang mit dem iPhone erhältlich ist. Es stammt von dem Betriebssystem Mac OS X ab. [5]

C. iPhone 3G Systemarchitektur

Das iPhone OS, oder auch OS X iPhone, ist das Betriebssystem des Apple iPhone. Dieses basiert auf der Software Mac OS X und ähnelt ihr in Aufbau und Konzeption. Die grundlegenden Strukturen des iPhone OS werden anhand eines System-Stack-Diagramms (Fig.2) erläutert und beschrieben.

Das System-Stack-Diagramm besitzt vier Layer: Core OS, Core Services, Media und Cocoa Touch.

Die Core OS und Core Services Layer beinhalten alle fundamentalen Schnittstellen, welche das iPhone OS benötigt. Darunter versteht man Interfaces für Netzwerkdienste, Bonjour Services[2], low-level Datentypen und weitere. Diese Schnittstellen sind C-

[1]Eine Software zur Wiedergabe von Multimediainhalten

[2]Technik zur automatischen Erkennung von Netzwerkdiensten in IP-Netzen

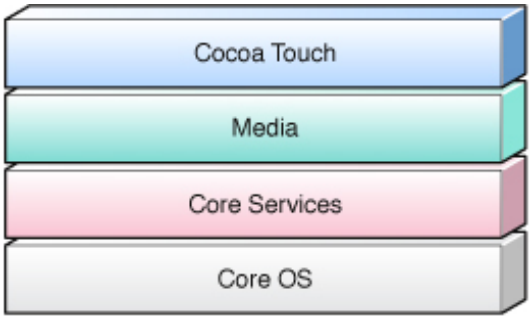

Abbildung 2. Die iPhone 3G System Architektur [6]

und Objective-C basiert und beinhalten Technologien, wie zum Beispiel SQLite oder CFNetwork. [5] [6]

1) Core OS: Unter dem Core OS versteht man die Kernelumgebung, worin sich die Treiber und Basisschnittstellen des Betriebssystems befinden. Der im OS X iPhone verwendete Kernel ist ein hybrid Kernel, genannt XNU (X is Not Unix).

XNU ist bekannt aus dem Open Source Betriebssystem Darwin, welches die Grundlage von Mac OS X bildet. Der XNU Kernel besteht aus den monolithischen FreeBSD[3] Kernel und den Mach-3.0-Kernel. Das Rechte- und Multiusermangement, sowie die Prozessaufteilung, TCP/IP und Synchronisierung, übernimmt der FreeBSD-Teil des Kernels. Für Speichermanagement, Debugging und Multitasking ist der Mach-Teil zuständig. [7] [8]

Eine der Hauptaufgaben des Core OS ist die Verwaltung des virtuellen Speichersystems und der Threads[4]. Die Verwaltung des Dateisystems und des Netzwerks, sowie die interne Prozesskommunikation, unterliegen ebenfalls dem Core OS.

Auf die sich im Kernel befindenden Treiber, besteht eingeschränkter Zugriff. Ausschließlich bestimmte System-Frameworks und Applikationen können darauf zugreifen. [6]

2) Core Services: Der Core Services Abschnitt des Betriebssystems stellt die fundamentalen Systemdienste für die Applikationen bereit. Alle sich in der Software befindenden Systemtechnologien beruhen auf der Grundlage der Core Services, welche sich in verschiedene Frameworks untergliedern.

Das Adressbook Framework beinhaltet alle Informationen über Kontakte. Diese Daten können von Applikationen, welche die nötigen Zugriffsrechte besitzen, abgerufen werden. Chat- und E-mail-Anwendungen können zum Beispiel die Adressbuchinformationen nutzen. Zu den Adressbook Framework gehört auch ein User Inferface Framework, welches eine Graphische Schnittstelle für ein Adressbuch bereitstellt. [9]

Dateimanagement und Servicefunktionen stellt das Core Foundation Framework bereit. Diese API ist C-basiert und beispielsweise zuständig für das Einstellungsmanagement. [10]

Das CFNetwork Framework bietet die Funktion mit Netzwerkprotokollen zu arbeiten. Dies ermöglicht die Kommunikation über FTP oder HTTP Server. [11]

Um die Sicherheit des Gerätes zu gewährleisten, steht das Security Framework zur Verfügung, welches die sich im iPhone OS befindlichen Daten möglichst gut schützen soll.

Ein interessantes Framework ist das Core Location Framework, dieses benutzt die, in der Hardware befindlichen GPS-Sensoren, um Längen- und Breitengrade zu bestimmen. Maps, eine standartmäßig installierte Applikation auf dem iPhone 3G, benutzt dieses Framework zur Positionsbestimmung. [12]

Das SQLite Library ermöglicht die Einbettung einer SQL Datenbank in eine Anwendung und ist optimiert für den schnellen Datenbankzugriff. [6]

3) Media: In dem Media Layer befinden sich sogennante high-level Frameworks. Diese sollen ein möglichst einfaches Erstellen von Animationen ermöglichen. Des Weiteren befinden sich Frameworks für den Audio Bereich in diesem Layer.

Eines der Graphic Frameworks ist das Core Graphics Framework oder auch Quartz, welches vektorbasiertes Zeichnen ermöglicht. Dieses bildet das grundlegende Darstellungsmodell des iPhone OS.

Das CoreAudio Framework stellt Informationen über Audiotypen und Dateiformate bereit. CoreAudio basiert auf OpenAL. Diese Bibliothek dient zur Erstellung von dreidimensionalen Soundeffekten und Raumklang. [13]

Das AudioToolbox Framework ist ein Teil von Co-

[3]frei erhältliches Unix Betriebssystem aus der BSD-Familie

[4]Ausführungsreihenfolge in der Abarbeitung eines Programms. Ein Thread ist Teil eines Prozesses

reAudio und beinhaltet Playback-, sowie Aufnahmefunktionen für Streams, als auch Audio-Dateien.

Dazu kommt noch das AudioUnit Framework, welches Funktionen für den Einsatz von Audiodateien beinhaltet.

Des Weiteren besitzt das iPhone OS ein MediaPlayer Framework, das für die Unterstützung der Filmwiedergabe verschiedenster Formate zuständig ist. [6]

4) Cocoa Touch: Der Cocoa Touch Layer stellt das UIKit und das Foundation Framework bereit. Diese bilden die Basiswerkzeuge und Infrastruktur um graphische, eingabegesteuerte Applikationen zu implementieren.

Das UIKit ist ein Objective-C basiertes Framework, welches jeder Applikation im iPhone OS ein bestimmtes Set an Funktionen liefert, beispielsweise ein User Interface Management oder auch eine Unterstützung für Text- und Webinhalte.

Alle Software-Entwickler die Anwendungen für das iPhone OS schreiben müssen auf die objektorientierte Programmiersprache Objective-C zurückgreifen. [6]

D. T-Mobile G1

Das T-Mobile G1 von HTC[5] und Google wurde am 22. Oktober 2008 in den USA veröffentlicht. Es ist das erste Mobiltelefon, welches das Betriebssystem Android unterstützt. [14]

Es verfügt über einen 3.2-Zoll großen Touchscreen und lässt sich über ihn bedienen. Anders als beim Apple iPhone wird dieser durch eine herausschiebbare vollständige QWERTY-Tastatur unterstützt.

Das Gerät verfügt außerdem über eine 3.2-Megapixel-Kamera und einen 1 GByte microSD Speicher, welcher bis auf 8 GByte erweiterbar ist. [15]

Das darauf laufende Betriebssystem Android ist ein Projekt der Open Handset Alliance. Zu der OHA gehören 48 Unternehmen wie Intel, Google, T-Mobile oder auch Motorola [16]. Die Android Plattform basiert auf einem Linux Betriebssystemkern und ist teilweise quelloffen.

E. Android Systemarchitektur

Die Android Systemarchitektur wird in diesem Abschnitt anhand eines System-Stack-Diagramms (Fig.3) erläutert. Dieser Stack besteht aus genau fünf Ebenen: der Kernelebene, den Libraries, der Android Runtime, den Application-Framework und den Applications.

Im folgenden werden wir die unteren vier Layer, also die Kernelebene, die System-Libraries, die Android Runtime sowie die Application Frameworks betrachten.[17]

1) Linux Kernel: Die Android Systemarchitektur wurde auf der Grundlage eines Linux Kernels entwickelt. Hierbei ist zu beachten, dass Android kein Linux ist. Der verwendete Kernel 2.6.24 wurde weitreichend modifiziert. Der Android-Linux Systemkern hat zum Beispiel keine Unterstützung der GNU C-Bibliothek, der glibc[6].

Der Grund dafür, das System auf der Grundlage eines Linux-Kernels aufzubauen, war der immense Funktionsumfang und die Tatsache, dass Linux Open-Source ist. Die bereits vorhandenen Eigenschaften, wie das ausgreifte Speicher- und Prozessmanagement, machten die Entwicklung um ein vielfaches einfacher. Ein weiterer Grund sich für einen Linux-Kernel zu entscheiden, war, das auf Benutzerrechte ausgelegte Sicherheitsmodell. Des Weiteren bietet der Systemkern ein bewährtes Treibermodell, das eine hohe Funktionalität in Bereichen von Audio und Video leistet, sowie die Unterstützung von Shared-Libraries.

Weitere Modifizierungen waren nötig um den Android Kernel als eingebettetes System[7] nutzen zu können. Es wurden z.B. Erweiterungen wie eine Alarmfunktion, ein Low Memory Killer[8], ein Kernel Debugger und ein erweitertes Energiemanagement hinzugefügt. Da mobile Endgeräte aufgrund ihrer Größe eine geringe Kapazität aufweisen und eine Batterie benötigen, wurden sogenannte wake-locks und andere energietechnische Funktionen eingeführt. Die wake-locks erhöhen oder verringern die Leistung des Displays, um energieeffizienter arbeiten zu können. [18] [19] [20]

2) Libraries: Alle sich in der Android Systemarchitektur befindlichen Libraries wurden in der Programmiersprache C oder C++ geschrieben.

Die sogenannte Bionic Libc Bibliothek ist eine Erweiterung der C-Standart Bibliothek. Diese wurde modifiziert für eingebettete Systeme und ist nicht kom-

[5]High Tech Computer Corporation, Taiwanesischer Mobiltelefonhersteller

[6]Freie Implemtierung der Standart C-Bibliothek

[7]Ein eingebettes System (engl. embedded system) bezeichnet einen elektronischen Rechner, der in einem technischen Kontext eingebunden ist. Diese Systeme unterliegen gewissen Rahmenbedingungen, wie z.B. geringem Energie- und Platzbedarf

[8]Klassifiziert die Prozesse ihrer Priorität nach in Gruppen, und beendet ,bei Aufruf, den Prozess in der schwächsten Gruppe

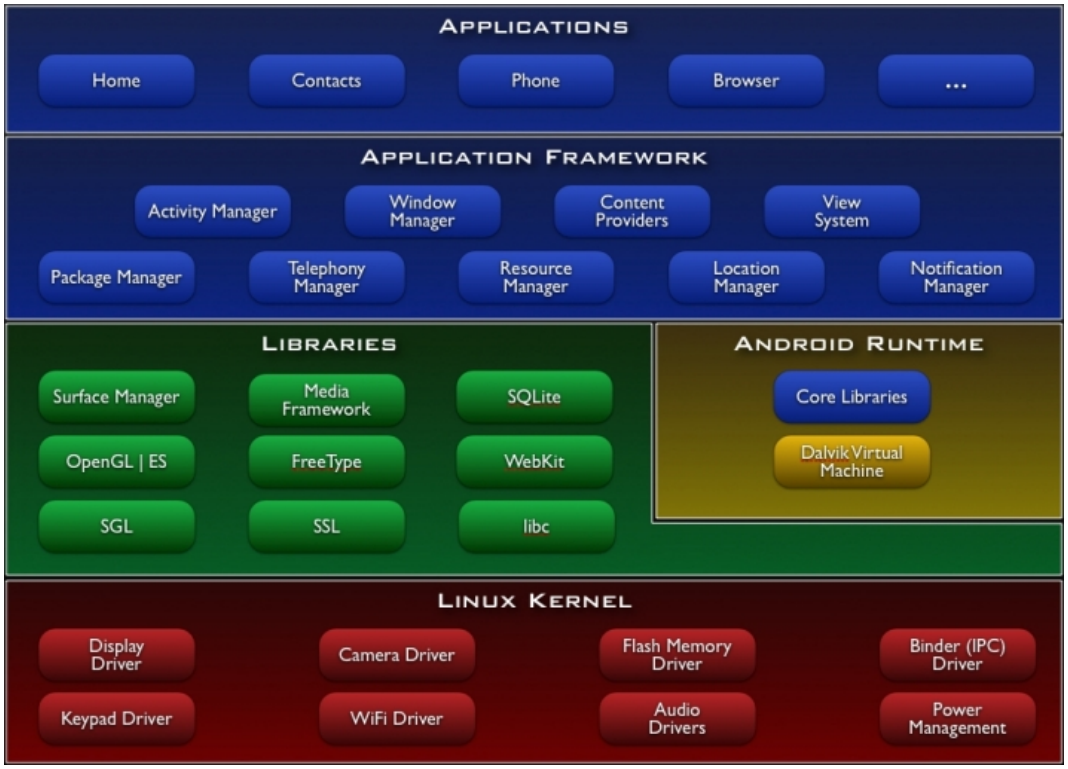

Abbildung 3. The Android System Architecture[17]

patibel zu der GNU Libc. Allem voraus wurde die Geschwindigkeit verbessert, sodass auch ein kleiner CPU schnell darauf zugreifen kann. Dies ist ausschlaggebend, da die Bibliothek in jedem Prozess geladen wird. Die Entscheidung für eine eigene Libc Implementation hat auch Lizensgründe. Die Entwickler wollten so die GPL aus dem User-Space raushalten.

Zu den Function Libraries zählt man SQLite, das Media Framework und Webkit.

SQLite ist die Programmbibliothek, welche ein Datenbanksystem enthält. Die Datenbank wurde speziell für den embedded-Einsatz entwickelt. Bei einer solchen Art von Datenverwaltungssystem wird keine Server-Software benötigt. Die Applikation kann direkt in die Anwendung integriert werden. SQLite hat auch nur eine Größe von wenigen hundert Kilobyte. Des Weiteren ist die Möglichkeit gegeben, die Datenbanken verteilt auf mehreren Dateien zu speichern. Dies erleichtert den Austausch zwischen Verschiedenen Systemen. [21]

Die Media Framework Bibliothek dient zur Handhabung von Audio- und Videodateien und basiert auf der PacketVideo Open Core Plattform. Durch die Bibliothek werden alle gängigen Audio- und Videoformate unterstützt.

Als Webkit bezeichnet man eine Open Source Bibliothek, die eine Grundlage bildet, auf der man einen Web-Browser entwerfen kann. Android benutzt genau so ein Webkit. Der grösste Vorteil der Bibliothek liegt jedoch darin, dass auch auf embedded systems die volle Größe der Webseite angezeigt und hohe Geschwindigkeit im Webseitenaufbau gewährleistet werden kann.

Zusätzliche Bibliotheken sind Surface Flinger und Audio Flinger.

Die Surface Flinger Bibliothek ermöglicht den Applikationszugriff auf das Display. Die Oberflächen einer Applikation können dadurch kombiniert werden. Kombinationen von 2D- und 3D-Inhalten sind ebenfalls möglich. Audio Flinger ist das Pendant nur für Audio Output Geräte, also z.B. Kopfhörer, Bluetooth oder Lautsprecher.

Für die sichere Datenübertragung im Internet wurde SLL eingefügt. Dieses hybride Verschlüsselungsprotokoll hat den Vorteil jedes höhere Protokoll auf Basis des SSL-Protokolls implementieren zu können. Somit ist eine System- und Anwendungsunabhängigkeit gewährleistet.

Die Hardware-Abstraction-Libraries bilden einen komplett neuen Layer im System Stack Diagramm. Diese C/C++ Bibliotheken definieren weitere Schnittstellen, welche Android verpflichten, für z.B WiFi, GPS oder eine Kamera, Hardwaretreiber zu implementieren. Dadurch wird die Portierung auf reelle Hardware erleichtert. Die Hardware-Abstraction-Libraries haben

quasi den Zweck eines SDK[9] nur für Hardware. [18] [20]

3) Android Runtime: Die Core Libraries bestehen aus Programmierschnittstellen für Java Programme. Diese Bibliotheken unterstützen leistungsfähige, einfache und bekannte Entwicklungshilfen für zum Beispiel Datenstrukturen, Datenzugriff, Netzwerkzugriff und Graphiken.

Alle Applikationen laufen auf der Dalvik Virtual Machine(DVM). Auf ihr werden Java Anwendungen ausgeführt, welche vorher in das kompakte Dalvik Executable Format, kurz .dex, umgewandelt wurden. Das konvertierte Format ist embedded-optimiert. Es nutzt also die geringe Speicher- und Prozessorkapazität bestmöglich aus. Im Bereich der Programmierung setzt Android also auf die Sprache Java.

Die DVM unterstützt des Weiteren mehrere virtual machine Prozesse pro Gerät. Die Datenstrukturen sind wiederum für embedded systems optimiert. [20] [22]

4) Application Framework: Das Application Framework beinhaltet die Klassen und Services, welche benötigt werden, um Applikationen zu erstellen. All diese wurden in der Programmiersprache Java geschrieben.

Das Activity Manager Framework ist für die Verwaltung der Applikationen konzipiert. Es stellt Funktionen bereit, um das Wechseln zwischen den Anwendungen zu ermöglichen. Unterstützt wird dieses durch das Package Manager Framework, welches eine Art Hilfe für den Activity Manager darstellt. Es hat die Aufgabe Packete und Informationen für den Activity Manager zu laden.

Die Hardware Services liefern Zugriff auf low-level Hardware API's[10]. Diese werden den Entwicklern zur Verfügung gestellt, um Anwendungen zu entwerfen. Dadurch kann man in seinen Applikationen zum Beispiel WiFi-Services oder Bluetooth-Services integrieren. [17] [18]

F. Neo Freerunner

Der Neo Freerunner ist der Nachfolger des Neo 1973 und unterstützt das Betriebssystem Openmoko. Die Entwicklung des neuen Telefons unterlag dem Ziel, eher den Massenmarkt zu erreichen, als dies sein Vorgänger tat. Allerdings ist auch der Freerunner, sowie das Neo 1973 kein Telefon für den allgemeinen Handy-Nutzer, sondern eher für fortgeschrittene Computer-User und Entwickler gedacht.

Der Neo Freerunner wurde am 3. Juli 2008 vorgestellt. Das Gerät verfügt über einen Touchscreen mit einer Auflösung von 480x640 Pixel, des Weiteren ist ein GPS-Modul, Bluetooth und Wireless Lan integriert. [23]

Auf dem Neo Freerunner läuft das Betriebssystem Openmoko. Diese Software ist noch unausgereift, beruht aber auf einem interessanten Konzept. Als Telefon ist es nur begrenzt einsetzbar, da die Entwicklung nicht genug vorangeschritten ist. Diese Software steht zwar noch nicht für Stabilität, aber dafür hinter einer kompletten Open Source[11] Philosophie. [24]

G. Openmoko Systemarchitektur

Die Openmoko Systemarchitektur basiert auf Open Source Technologien. Diese Technologien wurden meist nicht direkt von der Openmoko Inc. entwickelt, sondern existierten bereits.

1) Kernel: Der Openmoko Kernel basiert auf den Vanilla[12] 2.6.21.3 Linux Kernel. Dieser wurde ebenfalls, für eingebettete Systeme erweitert. Jene Erweiterung umfasst zum Beispiel Unterstützung für USB, SD und Touchscreen.

Der Kernel wird mit Hilfe des Bootloaders uBoot geladen, der ebenso Einsatz in anderen Linux embdedded systems findet. Der Bootloader initialisiert die Hardware, übergibt dann die Boot-Parameter an den Kernel und startet ihn. [26]

2) Linux Core services: In diesem Layer befinden sich bestimmte Systemdienste. GPS- und GSM-Daemons[13] und -Module gewährleisten die Kommunikation über GSM und GPS. Des Weiteren befindet sich ein Dbus System, welches Applikationen erlaubt miteinander zu kommunizieren, kurz IPC(inter process communication), in diesem Bereich. Ein Gerätemanager und ein Dienst für Bluetooth sind ebenfalls implementiert.

3) The User Interface: Das in Openmoko verwendete UI verfügt über einen Windowmanager namens Matchbox. Dieser fand bereits Verwendung bei Produkten anderer Hersteller. Matchbox ist optimiert für den Einsatz auf eingebetteten Systemen.

[9]Packetsammlung zur Softwareerstellung(engl. Software Development Kit)

[10]Programmierschnittstelle (engl. Application Programming Interface

[11]eine Lizensansammlung für Programme, deren Quelltext öffentlich zugänglich ist

[12]von umgangssprachlich engl. vanilla für einen Standard Kernel ohne Extras(wie zum Beispiel ein Distributionskernel)

[13]Programme, welche im Hintergrund Dienste zur Verfügung stellen

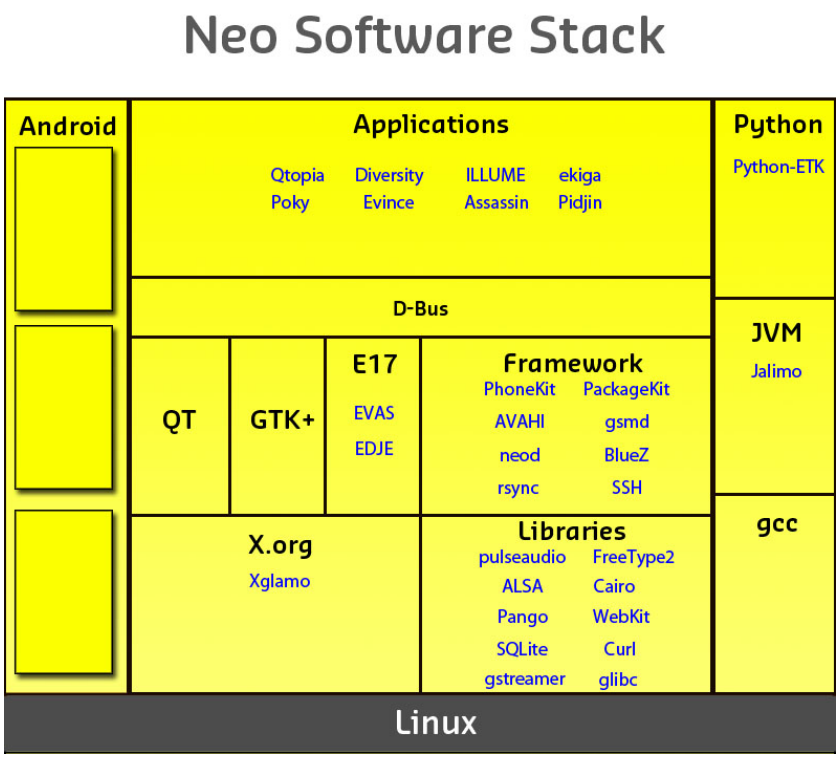

Abbildung 4. The openmoko System Architecture [25]

Die Audiofunktionalität wird gewährleistet durch das Advances Linux Sound Architecture, kurz ALSA, welches standartmäßig im Linux Kernel vorhanden ist.

Ein Teil des User Interface ist das GTK, das Gimp-Toolkit. Dies ist eine freie Komponentenbibliothek, mit deren Hilfe man eine GUI[14] erstellen kann. [27]

4) Application framework: Das Applikationsframework unterteilt sich in die folgenden vier Frameworks. Das Core oder auch libmokocore ist zuständig für das Senden von Nachrichten zwischen Applikationen. Des Weiteren liest und speichert es Konfigurationsdaten. Für das einheitliche Aussehen der Applikationen sorgt das sogenannte UI oder libmokoui[15]. Zuständig für Kontaktdaten ist der Personal Information Manager, kurz PIM. Das .NET oder libmokonet dient als einheitliche Schnittstelle für Netzwerkverbindungen über Bluetooth oder GPRS.

Programmiertechnisch stehen für Openmoko Software Entwickler prinzipiell alle in der Linuxwelt bekannten Programmiersprachen zur Verfügung, darunter zum Beispiel Java, C oder auch Perl. [26]

[14]graphische Benutzerschnittstelle(engl. Graphical User Interface)
[15]High Level API für das graphische Userinterface

III. Vergleich

Bei allen drei Systemarchitekturen handelt es sich um neu entwickelte Konzepte. Man kann viele Gemeinsamkeiten, aber auch Unterschiede entdecken.

Beim Vergleich der Kernel fällt auf, dass bei Openmoko, sowie bei Android aktuelle Linux Kernel verwendet werden. Dadurch wollen beide Hersteller gewährleisten, die Hardware effizienter anzusprechen und Leistungen zu steigern. Der Linux Kernel 2.6 wird immer mehr in Bezug auf den embedded Nutzen verbessert, somit wird er immer attraktiver für Hersteller, die Mobilfunkprodukte entwickeln. Des Weiteren spielt die Tatsache, dass der Kernel Open Source ist, einen außschlaggebenden Punkt in der Philosophie des Openmoko- und Androidprojektes. Der Linux Kernel bietet außerdem eine sehr gute Grundlage. Die vorhandenen Treiber für WLAN, Bildschirm, Kamera, Tastatur, Audio, Flash Speicher und viele mehr sind bereits integriert. Die Tatsache, dass der Linux Systemkern 2.6 ein auf Benutzerrechte ausgelegtes Sicherheitsmodell bereitstellt, war ein ausschlaggebender Punkt bei der Entscheidung für den Kernel. [28]

Openmoko und Android haben allerdings den Betriebssystemkern noch optimiert und für ihre eigenen Bedürfnisse angepasst.

Das iPhone 3G hingegen verwendet einen hybriden Systemkern, einen XNU Kernel. Apple entschied sich

aus Geschwindigkeitsgründen für die hybride Variante. Die modulare Struktur des Betriebssystemkerns lässt sich sehr leicht erweitern, und hat dadurch Vorteile gegenüber rein monolithischen Kerneln. Ebenso werden Geschwindigkeitseinbußen vermieden, die ein reiner Mikrokernel, wie ein purer Mach, verursachen würde.

Beide Kernelsysteme sind sehr ausgereift und bieten viele Möglichkeiten. Jedoch muss ein Linux Kernel auf die Vorteile eines Mikrokernels verzichten, zu denen beispielsweise gehört, dass Betriebssystem-Dienste in den Benutzeradressraum verschoben werden können. Gegenteilig dazu ist der Mikrokernelteil des XNU-Kernels langsamer als ein rein monolithischer Kernel. Dies liegt daran, dass Betriebssysteme, welche auf einem Mikrokern basieren mehr Kontextwechsel benötigen als rein monolitische.

Somit enstanden bei allen drei Systemen Betriebssystemkerne die Funktionen mitbringen, das jeweilige System energieeffizient und doch leistungsstark zu unterstützen.

Im Bereich der Programmierung bietet Openmoko die größte Kompatibilität. Es stehen quasi alle in der Linuxwelt bekannten Programmiersprachen zur Verfügung. Allerdings werden primär C/C++ und Phython eingesetzt.

Für das iPhone 3G lassen sich Applikationen in der relativ unbekannten Sprache Objective-C schreiben. Dies stößt bei vielen Programmierern auf Kritik, da wenige diese Sprache kennen. Objective-C oder auch ObjC, erweitert die Programmiersprache C mit objektorientierten Sprachmitteln. Die Syntax von dieser Erweiterung trennt sich von C Syntax komplett ab und ist an die Programmiersprache Smalltalk angelehnt. [29]

Android hingegen lässt den Entwickler in Java programmieren, allerdings produziert die VM keinen Java Bytecode, sondern seinen eigenen Dalvic Bytecode. Dies spaltet die Android Plattform von Java ab. Jedoch ist dieser Bytecode optimiert für den Einsatz von Android. Da der verwendete ARM[16] Prozessor und die Dalvik Virtual Machine nicht wie die Java VM, als Stapelmaschine arbeiten, sondern als Registermaschine, benötigt man weniger Schritte um den Bytecode auszuführen.

Die Hersteller oder Entwickler liefern Hilfsmittel wie zum Beispiel ein Software Development Kit, mit dessen Hilfe neue Anwendungen erstellt werden können. Solche Hilfsmittel unterscheiden sich nicht sonderlich in ihren Funktionen voneinander.

Hardwaretechnisch unterscheiden sich der Neo Freerunner von Openmoko, das T-Mobile G1 von HTC und das Apple iPhone 3G nur in geringem Maße. Alle drei Hersteller setzen auf Komponenten mit ähnlicher Leistung. Die Tabelle in der Grafik Nr. I erläutert dies.

Vergleicht man die Ziele der drei Systeme, so lassen sich große Differenzen erkennen. Openmoko ist daran interessiert dem Kunden vollkommene Offenheit zu gewähren. Alle Bestandteile sollen Open Source sein. Alle Lizenzen sind so ausgerichtet, den User selbst an dem Projekt teilhaben lassen zu können. Es wurden sogar Schaltpläne des Gerätes unter der Creative Common Lizenz veröffentlicht. So können Ingenieure nun frei über die nötigen Informationen verfügen, um beliebige Funktionen und Hardware hinzufügen zu können. So erleichtert sich Openmoko auch die Fehlersuche am eigenen Mobiltelefon. Der Fokus des Gerätes liegt darauf, es sich nach eigenem Ermessen zu konfigurieren. Softwareentwicklern ist es frei überlassen in welcher Programmiersprache sie Applikationen für Openmoko entwerfen.

Android hingegen scheint auf den ersten Blick ein ähnliches Ziel wie die Openmoko-Plattform zu verfolgen. Doch die grenzenlose Offenheit, die ein Openmoko-System bietet, besitzt es nicht. Für Softwareentwickler bietet Android eine nicht so riesige Bandbreite an Programmiersprachen wie Openmoko. Es wird allein auf die Sprache Java zurückgegriffen, in Verbindung mit der Dalvik Virtual Machine[26]. Unter dem sogenannten Android Open Source Project veröffentlichte die Open Handset Alliance am 22. Oktober 2008 den Android Kernel und bietet so den Entwicklern eine bessere Einsicht in das System. Der Grund für die Veröffentlichung von Android ist nicht allein der, dem User ein offenes System anzubieten, sondern Google und die Open Handset Alliance verfolgen das Ziel, ihre Online Applikationen über die Android-Plattform publik zu machen.

Das Apple iPhone 3G ist der absolute Gegenpart zu den doch sehr offenen Plattformen Android und Openmoko. Die Entwicklung des Systems unterliegt allein Apple. Das iPhone 3G ist zusätzlich nur in Verbindung mit einem Apple Computer programmierbar. Auch die Programmiersprache Objective-C, in Verbindung mit dem Cocoa Touch Framework, ist weitreichend unbekannt. Diese Tatsachen machen es schwierig, eine große Entwickler Community aufzubauen. [26] [30]

IV. Zusammenfassung

Das iPhone 3G und die Plattform iPhone OS wird mit aller Wahrscheinlichkeit durch den Namen Apple und der Produktvermarktung bestehen bleiben. Das iPhone ist ein Designprodukt mit fortschrittlicher Technik. Es bietet dem Kunden viele Features, jedoch grenzt sich Apple ab und herrscht über die Entwicklung. Das iPhone

[16]RISC-Konzept-, 32-Bit Mikroprozessor

	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

Tabelle I
HARDWAREVERGLEICH DES NEO FREERUNNER MIT DEM T-MOBILE G1 UND DEM IPHONE 3G

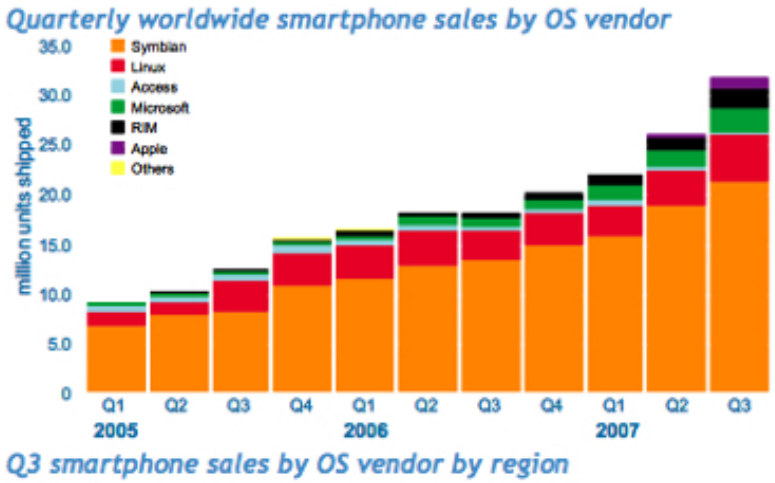

Abbildung 5. Weltweite Smartphone Verkaufszahlen

OS ist ein Übertrag des Betriebssystems Mac OS X auf ein embedded system. Daraus resultieren sehr ähnliche Applikationen und Funktionen für das OS X iPhone. Es wurde also, das bereits vorhande Betriebssystem modifiziert und angepasst. Zum Beispiel wurde der Kernel weitreichend optimiert, um die hohe Kapazitätsbeschränkung eines Smartphones besser ausnutzen zu können. Diese Modifizierungen nahmen auch Openmoko und Android vor, um ihren Kernel zu verbessern. Ebenfalls zu erkennen ist, dass die Libraries optimiert wurden. Diese erhielten Geschwindigkeitsverbesserungen sowie verbesserte Zugriffsgeschwindigkeiten um den kleinen CPU und Speicher gerecht zu werden.

In der Softwarearchitektur sieht man ebenfalls ähnliche Strukturen. Auf allen Palttformen findet man zum Beispiel die SQLite Bibliothek, welche sich durch embedded-optimierte Eigenschaften auszeichnet. Die Softwarearchitektur wurde also ebenfalls verbessert und an die gegebenen Hardwareverhältnisse angepasst.

Android wird man, im Gegensatz zum iPhone OS, in Zukunft auf mehreren Mobiltelefonen finden. Die Systemarchitektur ist so ausgelegt, hohe Hardwarekompatibilität zu gewährleisten. Die Open Handset Alliance verschließt sich nicht, sondern bietet Einblick in die Entwicklung. Diese Tatsachen können das Betriebssystem Android weit verbreiten.

Apple, die OHA und Openmoko veröffentlichten Software-Development-Kits. Dadurch geben sie Software-Entwicklern eine vereinfachte Möglichkeit, Anwendungen für die jeweiligen Plattformen zu schreiben.

Openmoko wird es wahrscheinlich nicht so leicht haben wie die beiden anderen Betriebssysteme, hinter denen große Namen wie Apple oder Google stehen, wenn das System nicht ausgereifter wird. Das Konzept ist gelungen. Openmoko könnte theoretisch auf jedem linuxfähigen Mobiltelefon laufen. Für Softwareentwickler ist Openmoko gut geeignet. Ein weiterer Vorteil ist das große Angebot an freier Software, was Openmoko auch für den allgemeinen Telefonnutzer attraktiv machen

kann.

Alles in allem, verbergen sich hinter den Next-Generation-Mobile-Phones kleine Computer, welche fortschrittliche embedded system Technik in sich tragen. Dazu kommen Funktionalität, Benutzerfreundlichkeit und gute Ideen. Die Hersteller versuchen durch die Entwicklung der Systeme neue, rasant wachsende Märkte zu erschließen. Auf der Grafik Nr. 5 von Canalys fällt stark ins Auge, dass die Mobilfunkbranche weiterhin stark wächst. Das weitverbreitetste Betriebssystem ist eindeutig das Symbian OS. Durch die Einführung der beiden Linux Betriebssysteme Openmoko und Android, kann diese Statistik in den nächsten Jahren etwas weiter zu Gunsten der Linux-Betriebssysteme ausfallen. Allerdings lassen Hersteller wie RIM oder Nokia diese Entwicklung nicht an sich vorbeigehen. Sie erweiterten längst ihre Produktpaletten um zum Beispiel das Blackberry Storm oder die neuen N-Series Telefone.

LITERATUR

[1] [Online]. Available: http://www.apple.com/iphone

[2] [Online]. Available: http://www.portel.de/fileadmin/pics/T-Z/T-Mobile-G1-1-09.jpg

[3] [Online]. Available: http://mobile-place.info/home/images/stories/vijesti/openmoko/neo-001.jpg

[4] "Apple iphone," zuletzt besucht - 12.2008. [Online]. Available: http://de.wikipedia.org/wiki/Apple_iPhone

[5] "iphone os," zuletzt besucht - 01.2009. [Online]. Available: http://en.wikipedia.org/wiki/IPhone_OS

[6] *iPhone OS Overview*. [Online]. Available: http://developer.apple.com/iphone/gettingstarted/docs/iphoneosoverview.action

[7] "Mac os x - xnu the kernel," zuletzt besucht - 01.2009. [Online]. Available: http://www.kernelthread.com/mac/osx/arch_xnu.html

[8] "Xnu," zuletzt besucht - 12.2008. [Online]. Available: http://en.wikipedia.org/wiki/XNU

[9] "Adress book programming guide," zuletzt besucht - 01.2009. [Online]. Available: http://developer.apple.com/documentation/UserExperience/Conceptual/AddressBook/AddressBook.html

[10] "About core foundation," zuletzt besucht - 12.2008. [Online]. Available: http://developer.apple.com/DOCUMENTATION/CoreFoundation/Conceptual/CFDesignConcepts/Concepts/AboutCF.html

[11] "Introduction to cfnetwork programming guide," zuletzt besucht - 01.2009. [Online]. Available: http://developer.apple.com/DOCUMENTATION/Networking/Conceptual/CFNetwork/Introduction/chapter_1_section_1.html

[12] "iphone os core location," zuletzt besucht - 01.2009. [Online]. Available: http://en.wikipedia.org/wiki/IPhone_OS#Core_Location

[13] "Core audio," zuletzt besucht - 01.2009. [Online]. Available: http://en.wikipedia.org/wiki/Core_Audio

[14] U. Hannemann, "Das google handy im praxistest," *Focus Digital*, 2008. [Online]. Available: http://www.focus.de/digital/internet/google/t-mobile-g1-das-google-handy-im-praxistest_aid_335417.html

[15] J. Topolsky, "T-mobile g1 review," *engadget*, 2008.

[16] "Open handset alliance - members," zuletzt besucht - 12.2008. [Online]. Available: http://www.openhandsetalliance.com/oha_members.html

[17] "What is android?" zuletzt besucht - 01.2009. [Online]. Available: http://code.google.com/android/what-is-android.html

[18] P. B. (Google), "Anatomy & physiology of an android," 2008. [Online]. Available: http://sites.google.com/site/io/anatomy--physiology-of-an-android

[19] "Linux kernel - neuerungen im 2.6 kernel," zuletzt besucht - 01.2009. [Online]. Available: http://de.wikipedia.org/wiki/Linux_(Kernel)#Neuerungen_im_Kernel_2.6

[20] G. Printemps, "Deep inside android," Tech. Rep., 2008. [Online]. Available: http://www.openexpo.ch/fileadmin/documents/2008Zuerich/Slides/33_Printemps.pdf

[21] "About sqlite," zuletzt besucht - 12.2008. [Online]. Available: http://www.sqlite.org/about.html

[22] "Dalvik virtual machine - introduction," zuletzt besucht - 12.2008. [Online]. Available: http://www.dalvikvm.com/#Introduction

[23] J. Ihlenfeld, "Neo freerunner - freies mobiltelefon für den massenmarkt," 2008.

[24] "Neo freerunner," zuletzt besucht - 12.2008. [Online]. Available: http://wiki.openmoko.org/wiki/Neo_FreeRunner/de

[25] "Neo software stack," zuletzt besucht - 01.2009. [Online]. Available: http://wiki.openmoko.org/images/7/7b/Software1.jpg

[26] T. Schmidbauer, Mechtler, Tech. Rep., 2008. [Online]. Available: stderror.at/docu/moko_wp.pdf

[27] "Why openmoko," zuletzt besucht - 12.2008. [Online]. Available: http://wiki.openmoko.org/wiki/Why_Openmoko

[28] A. R. Deshpande, "Linux kernel 2.6: the future of embedded computing," 2004. [Online]. Available: http://www.linuxjournal.com/article/7477

[29] "Introduction to the objective-c 2.0 programming language," zuletzt besucht - 01.2009. [Online]. Available: http://developer.apple.com/documentation/Cocoa/Conceptual/ObjectiveC/Introduction/chapter_1_section_1.html

[30] "Der xnu kernel von mac os x," zuletzt besucht - 01.2009. [Online]. Available: http://www.macmark.de/osx_xnu.php