Mobile Platforms

- Collections of central services and libraries with both reactive and proactive functions
- APIs typically logically centralized
- Distributed between elements of the environment
  - Multi-tier client-server
  - Peer-to-peer
  - Hybrids
- The platform running on the mobile terminal and the characteristics of the device determine how service is rendered for the end user
Wireless and Cloud

- Wireless hop is the limiting factor
  - Bandwidth, connectivity, reachability, costs

- Server side scalability can be achieved by using traditional solutions:
  - clusters, caching, geographical distribution, load balancing, data centers

- Cloud computing
  - Integration, offloading
  - Web apps vs. native apps
Challenges

- Fragmentation is a major problem
- Different types of fragmentation
  - device-level fragmentation
  - standard fragmentation
  - implementation fragmentation
- Security is also a problem
  - Sandbox environments and privileged operations require certification
  - Certification is difficult for developers
  - Current trend is towards application stores and more lightweight certification processes
Examples

- WAE
- Java ME
- Symbian
- Windows Phone 7
- iPhone
- MAEMO / MeeGo
- Android
- Summary
Wireless Application Environment (WAE)

- A suite of protocols and specifications for optimizing data transfer for wireless communication

WAP stack

- Focus on binary transmission
- WSP (Wireless Session Protocol)
  - HTTP replacement, “compressed”
- WTP (Wireless Transaction Protocol)
  - Request/response, more efficient than TCP
- WTLS (Wireless Transport Layer Security)
  - Based on TLS, may not be end-to-end with a gateway
- WDP (Wireless Datagram Protocol)
  - UDP replacement
Web Access with Gateway

Client

Web Browser

encoded request

wireless

encoded response

Gateway

Encoders

Decoders

Protocol

Gateways

request

response

Server

HTTP Server

CGI,
Web Access

- Data transformation
  - WAP gateway performs data transformation between WML (or XHTML) and HTML
- Data compression
  - Technique are used for dealing with images and other graphics
- Adaptability
  - User profile and device characteristics are stored in the WAP gateway
- Security
  - Secure Enterprise Proxy (SEP) using 128-bit encryption in WAP 1.2
- Service discovery and mobility support
  - WAP’s “walled garden” – WAP gateways are provided by ISP such as AOL
WAE: current status

- WAP Forum now in OMA (Open Mobile Alliance)
- WAP 2.0, is a re-engineering of WAP using a cut-down version of XHTML with end-to-end HTTP
- Gateway and custom protocol suite is optional.
- WAP used by many handsets
  - 1.2 version introduced WAP Push (typically using an SMS message)
- Typically versatile networking stacks with also IPv6 support
Java Micro Edition (Java ME)

- Java for consumer electronics and embedded devices
- A virtual machine and a set of APIs
- Configurations and profiles
  - Configurations
    - two-low level APIs and optimized VMs
      - CDC, CLDC
  - Profiles
    - API specification on top of a configuration for complete runtime
    - CLDC: MIDP
    - CDC: Foundation, Personal Basis, Personal
    - Profiles defined using Java Community Process (JCP)
Java Editions

- Java Card, which allows small Java-based applications to be executed on smart cards
- Java ME (Micro Edition, formerly J2ME), which specifies several different profiles, collections of libraries
  - for devices that are sufficiently limited that it is not feasible to support the full Java platform on them.
- Java SE (Standard Edition), which is the platform for general purpose desktop PCs.
- Java EE (Enterprise Edition), which includes the Java SE and a number of additional
  - APIs for multi-tier client-server enterprise applications.
Servers & enterprise computers

Servers & personal computers

High-end PDAs

TV set-top boxes

Embedded devices

Mobile phones & entry-level PDAs

Smart cards

Optional Packages

Java 2 Platform, Enterprise Edition (J2EE)

Optional Packages

Java 2 Platform, Standard Edition (J2SE)

Optional Packages

Personal Profile

Personal Basis Profile

Foundation Profile

Optional Packages

MIDP

Optional Packages

CLDC

KVM

Java Card

Card VM

Java Platform, Micro edition (Java ME)
Important JSRs

- 75 File Connection and PIM
- 82 Bluetooth
- 120 Wireless Messaging API (WMA)
- 135 Mobile Media API (MMAPI) Audio, video, multimedia
- 172 Web Services
- 177 Security and Trust Services
- 179 Location API
- 180 SIP API
- 184 Mobile 3D Graphics
- 185 Java Technology for the Wireless Industry (JTWI) General
- 205 Wireless Messaging 2.0 (WMA)
- 211 Content Handler API
- 226 SVG 1.0
- 229 Payment API
- 234 Advanced Multimedia Supplements (AMMS) MMAPI extensions
- 238 Mobile Internationalization API
- 239 Java Bindings for the OpenGL ES API
- **248 Mobile Service Architecture General**
- 256 Mobile Sensor API
- 287 SVG 2.0
MSA 248 for CLDC
MIDP 3.0

- MIDP 3 specified in JSR 271 will specify the 3rd generation mobile APIs.
  - AMS (Application Management System)
  - Multitasking
  - Provisioning and OTA
  - Shared libraries
  - Security and access control
  - Service framework
  - Inter-MIDlet communication
  - User Interface improvements

- A key design goal of MIDP3 is backward compatibility with MIDP2 content

- Approved in Dec, 2009. Not supported by current phones.
CDC Technology

- Connected Device Configuration (CDC); JSR 36, JSR 218
- Foundation Profile, (FP); JSR 46, JSR 218
- Personal Basis Profile, (PBP); JSR 129, JSR 217
- Personal Profile, (PP); JSR 62, JSR 216
- J2ME RMI Optional Package, (RMI OP); JSR 66
- JDBC Optional Package for CDC/Foundation Profile API; JSR 169
- Java TV API; JSR 927
- Java Secure Socket Extension for Connected Device Configuration (JSSE)
APIs

- Mobile Sensor API, Contactless Communication API, and Location API
- The Mobile Broadcast Service API supports the delivery of streaming multimedia to mobile phones
- Converged communications support is provided by the XML API and *IP Multimedia Subsystem (IMS) Services API*
- *The Mobile User Interface Customization API and Scalable 2D Vector Graphics API*
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Java ME MIDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>Java ME</td>
</tr>
<tr>
<td>Network scanning</td>
<td>No</td>
</tr>
<tr>
<td>Network interface control</td>
<td>No</td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes (multi-tasking support in MIDP 3.0)</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>No</td>
</tr>
<tr>
<td>Memory management</td>
<td>Limited</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Limited, extension</td>
</tr>
<tr>
<td>Location information</td>
<td>Extension</td>
</tr>
<tr>
<td>HTML 5</td>
<td>N/A</td>
</tr>
<tr>
<td>SIP API support</td>
<td>Extension</td>
</tr>
<tr>
<td>Open Source</td>
<td>No</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Fragmented</td>
</tr>
</tbody>
</table>
Symbian

- OS for handheld devices with limited resources
- User interface framework
- APIs (C++)
- Tools
- Operating System
  - Pre-emptive, multitasking, multithreading, memory protection
  - Event-based, active objects
  - Memory conservation, reliability, CPU optimizations
Software Components

- **Kernel**
  - Manages and controls access to hw
  - Hw-supported privileges, kernel mode

- **Application**
  - Program with a user interface
  - Runs in user mode in its own process

- **Server**
  - Program without a user interface
  - Manages resources, provides interface to clients
  - File server, window server, comms, ..

- **Engine**
  - Application part that manipulates data, typically separate DLL
The Symbian OS System Model contains the following layers:
- UI Framework Layer.
- Application Services Layer.
- Java ME.
- OS Services Layer: generic OS services, communications services, multimedia and graphics services, connectivity services.
- Base Services Layer.
- Kernel Services and Hardware Interface Layer.
Symbian OS 9.x

- Symbian OS 9.x, most applications must be signed using a centralized process provided by Symbian Ltd. in order to be installable and executable on a mobile phone.
- An unsigned applications have very limited features and at least in theory cannot perform harmful actions.
- One motivation for application signature process, called Symbian Signed, is to improve mobile phone security by preventing the installation and execution of unknown and possibly hazardous programs.
- Several viruses and trojan horse programs have been developed for the OS, for example Cabir, which have caused some concerns for the trustworthiness of mobile software and prompted a number of anti-virus products for mobile phones.
WebKit for S60

- The Nokia Web Browser is based on the S60WebKit
- WebKit contains the WebCore and JavaScriptCore components that Apple uses in its Safari browser.
- Based on KHTML and KJS from KDE’s Konqueror open source project, this software offers improvements in Web site usability on smartphones through the reuse of an existing desktop rendering engine that has been developed and optimized by a large open source community over many years.
- The Nokia Web Browser supports Dynamic HTML, AJAX applications, and W3C’s XHTML 1.0, DOM, CSS and SVG-Tiny.
- Other supported Web standards include SSL and ECMAScript; and Netscape style plug-ins such as Flash Lite and audio.
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Symbian Series 60</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>C++, Qt, Python, various</td>
</tr>
<tr>
<td>Network scanning</td>
<td>Limited</td>
</tr>
<tr>
<td>Network interface control</td>
<td>Yes</td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Yes</td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
</tr>
<tr>
<td>HTML 5</td>
<td>No (Widgets and Javascript API)</td>
</tr>
<tr>
<td>SIP API support</td>
<td>Yes</td>
</tr>
<tr>
<td>Open Source</td>
<td>Yes</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Some fragmentation</td>
</tr>
</tbody>
</table>
Windows Mobile

- Windows Mobile 6 was released by Microsoft at the 3GSM World Congress 2007 and it comes in three flavours
  - standard version for smartphones
  - a version for PDAs with phone functionality
  - a classic version for PDAs without phone features.
- Based on the Windows CE 5.0 operating system and has been designed to integrate with Windows Live and Exchange products.
- Software development for the platform is typically done using Visual C++ or .NET Compact Framework.
- When native client-side functionality is not needed, server-side code can be used that is deployed on a mobile browser, such as the Internet Explorer Mobile bundled with Windows Mobile.
Compact Framework

- The .NET Compact Framework is a subset of the .NET Framework and shares many components with the desktop software development environment.
- The framework includes:
  - optimized Common Language Runtime (CLR)
  - a subset of the .NET Framework class library, which supports features such as Windows Communication Foundation (WCF) and Windows Forms.
  - It also contains classes that are designed exclusively for the .NET Compact Framework.
- A platform adaptation layer exists between Windows CE and the common language runtime:
  - mapping the services and device interfaces required by the CLR and Framework onto Windows CE services and interfaces.
- Managed components developed using .NET languages, such as C#, are used to create the applications.
- It is also possible to use the Win32 API with .NET CF, which allows directly to access OS features.
The .NET Compact Framework CLR is made up of the following three components:

- Class libraries
- Execution engine
- Platform adaptation layer

The purpose of the class libraries is to provide a basic set of classes, interfaces, and value types, the foundation for developing applications in .NET.

The execution engine is the core component of the CLR. It provides the fundamental services needed for executing managed code.

- The execution engine includes components such as a JIT compiler, a class and module loader, and a garbage collector.

The PAL layer maps calls from the execution engine to the functions of the underlying operating system.
CLR

Common Intermediate Language

Source: http://en.wikipedia.org/wiki/Common_Language_Runtime
WP7

- Windows Mobile 7 was announced in 2010
- Limited APIs for third party applications
- WP7 store for applications
- Current development: C# and XNA, Silverlight 4, VB, on-going API work
- XNA is based on .NET
Development

- Simple applications with Silverlight
  - XML-based UI declaration and C# code
  - Executed in .NET CLR
  - UI widgets, event based input
  - Similar to Android Java and XML

- Games with XNA
  - C# code executed in .NET CLR
  - Direct 3D
  - Xbox

- SDK is free of charge
  - Microsoft Windows Phone Developer Tools
  - Windows 7 required
  - Add-on for Visual Basic
WP7 Software Architecture

Applications
- Your App UI and logic
- Frameworks
  - Silverlight
  - XNA
  - HTML/JavaScript
- CLR

App Model
- App management
- Licensing
- Chamber isolation
- Software updates

UI Model
- Shell frame
- Session manager
- Direct3D
- Compositor

Cloud Integration
- Xbox LIVE
- Bing
- Location
- Push notifications
- Windows Live ID

Kernel
- Security
- Networking
- Storage

Hardware Foundation
- Media
  - A-GPS
  - Accelerometer
  - Proximity
  - Wi-Fi
- Networking
  - Compass
- Storage
  - Light
- Graphics

HW
<table>
<thead>
<tr>
<th>Feature</th>
<th>Windows Mobile .NET</th>
<th>Windows Phone 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>C# and .NET, Win32, various</td>
<td>C#, Silverlight, XNA</td>
</tr>
<tr>
<td>Network scanning</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Network interface control</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes</td>
<td>Not supported for third party applications in WP7</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Yes</td>
<td>No?</td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HTML 5</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIP API support</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Open Source</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate</td>
<td>Certificate, app store (WP7)</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Some fragmentation</td>
<td>Not fragmented</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Summary**

- **Windows Mobile .NET**
  - Development: C# and .NET, Win32, various
  - Network scanning: Yes
  - Network interface control: Yes
  - Background processing: Yes
  - Energy and power monitoring and control: Yes
  - Memory management: Yes
  - Persistent storage: Yes
  - Location information: Yes
  - HTML 5: No
  - SIP API support: No
  - Open Source: No
  - 3rd party application installation: Certificate
  - Level of fragmentation: Some fragmentation

- **Windows Phone 7**
  - Development: C#, Silverlight, XNA
  - Network scanning: No
  - Network interface control: No
  - Background processing: Not supported for third party applications in WP7
  - Energy and power monitoring and control: No?
  - Memory management: Yes
  - Persistent storage: Yes
  - Location information: Yes
  - HTML 5: No
  - SIP API support: No
  - Open Source: No
  - 3rd party application installation: Certificate, app store (WP7)
  - Level of fragmentation: Not fragmented
The iPhone OS is a mobile operating system developed by Apple Inc. for their iPhone and iPod touch products.

The OS is derived from Max OS X and uses the Darwin foundation.

Darwin is built around XNU, a hybrid kernel that combines the Mach 3 microkernel, various elements of Berkeley Software Distribution (BSD) Unix, and an object-oriented device driver API (I/O Kit).

The iPhone OS is based on four abstraction layers, namely the Core OS layer, the Core Services layer, the Media layer, and the Cocoa Touch layer.

An universal application determines the device type and then uses the available features based on conditional statements.
The iPhone OS’s user interface is based on *multi-touch gestures*. Interface control elements consist of sliders, switches, and buttons.

Interaction with the OS includes gestures such as swiping, tapping, pinching, and reverse pinching.

Additionally, using internal accelerometers, rotating the device on its y-axis alters the screen orientation in some applications.
GUI ("Aqua")

API

Carbon
Quick-time
Carbon core

BSD  Classic  Cocoa touch

Application Services
Quartz  OpenGL  PrintCore...

Core Services
Core foundation  Core services  non-GUI API...

Core OS ("Darwin")

System utilities

Kernel ("xnu")

File systems
Networking NKE
POSIX
1/0 kit  Drivers

Hardware
Versions

- iOS 1 Web apps
- iOS 2 introduced the App Store
- iOS 3 single tasking, new features
- iOS 4 multitasking, FaceTime videoconferencing, iBooks, iAd, in-app purchases
The iPhone OS 4.0 was announced in April 2010 and it supports multitasking for 3rd party applications.

The key design principle is to offer APIs for specific background operations in order to be able to optimize overall system performance.

For example VoIP applications will be able to receive calls in the background.

Third party push servers are supported for sending notifications to applications.
iOS4 multitasking

- The new iPhone multitasking-specific APIs includes
  - support for background audio play
  - VoIP
  - location services
  - task completion, and
  - fast application switching
Application model

- Mac OS X model
  - Traditional main() starts UI event loop
  - View controller reacts to events
  - View objects manage visible objects
  - View controller unserializes views from a .nib file (UI builder)
  - All code is native

- Patterns
  - Model-View-Controller
  - Delegate
    - Implement application specific logic
Structure

Model
- Data Model Objects

Controller
- UI application
  - Event loop
- Application delegate
- Controller objects

View
- UIWindow
- Views and UI objects
iPhone Events

XCode

- Xcode is Apple’s IDE for iOS development
- Interface Builder
  - XML view definitions, .nib files
- Emulator for testing
- Pointer: developer.apple.com
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>iPhone OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>Objective-C</td>
</tr>
<tr>
<td>Network scanning</td>
<td>No</td>
</tr>
<tr>
<td>Network interface control</td>
<td>No</td>
</tr>
<tr>
<td>Background processing</td>
<td>No (Yes for 4.0)</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Monitoring since 3.0</td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
</tr>
<tr>
<td>HTML 5</td>
<td>Yes</td>
</tr>
<tr>
<td>SIP API support</td>
<td>Limited</td>
</tr>
<tr>
<td>Open Source</td>
<td>No</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate, Apple AppStore</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Minor fragmentation</td>
</tr>
</tbody>
</table>
MAEMO and MeeGo

- Open Source development platform for Nokia Internet Tablets and other Linux-based devices
- Previously MAEMO, integrated with Intel’s Moblin to create MeeGo
- MeeGo 1.1 for Atom and ARM
- www.meego.com
The Maemo platform from Nokia includes the Internet Tablet OS, which is based on Debian GNU/Linux and draws much of its GUI, frameworks, and libraries from the GNOME project.

Maemo is based on the Linux operating system kernel, which is a monolithic kernel that supports multiple hardware platforms.

It uses the Matchbox window manager, and like Ubuntu Mobile, it uses the GTK-based Hildon as its GUI and application framework. The Maemo platform is intended for Internet tablets, which are smaller than laptops, but larger and more versatile than PDAs.

A tablet may have a small keyboard, and central characteristics include a stylus and a touch-sensitive screen.

Graphical interfaces must be designed with the touch screen in mind.
MeeGo

- Versatile platform for mobile computing
- Linux-based, Qt is the key development environment
- The MeeGo includes a set of components called the content framework to gather and offer user metadata to application developers
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>MeeGo Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>C/C++, Qt APIs, various</td>
</tr>
<tr>
<td>Network scanning</td>
<td>Yes</td>
</tr>
<tr>
<td>Network interface control</td>
<td>Yes</td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Yes</td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
</tr>
<tr>
<td>HTML 5</td>
<td>Depends on WebKit version</td>
</tr>
<tr>
<td>SIP API support</td>
<td>Yes</td>
</tr>
<tr>
<td>Open Source</td>
<td>Yes</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Not fragmented</td>
</tr>
</tbody>
</table>
Qt

- Qt is a cross-platform application framework
  - Rapid creation of GUIs
- For Linux and Symbian application development
- The Qt API is implemented in C++ and most Qt developers use C++ (bindings for other languages)
- Extensions for using mobile functionality from within Qt code
  - access points, alarms, audio, calendar, camera, contacts, installer, landmarks, location, media, messaging, profile, resource access, sensor, settings, system information, telephony, vibration, other utilities etc.
Android I

- Mobile OS and application platform from Google
- Open Handset Alliance
- Linux kernel
- Open Source
- Uses Java to build applications (Java SE class library parts from Apache Harmony project)
- Optimized virtual machine called "Dalvik"
  - Runs .dex files (derived from .class or .jar)
  - Relies on underlying system for process isolation, memory mng, and threading
- Independent of Sun and JCP
- Java APIs for basic comms, location, SQLite, OpenGL, SyncML
Android includes a set of C/C++ libraries used by various components of the Android system. The capabilities of these libraries are exposed to developers through the Android application framework APIs.

The core libraries include:

- System C library, a BSD-derived implementation of the standard C system library (libc), adapted for embedded Linux-based devices.
- Media Libraries based on PacketVideo’s OpenCORE.
- Surface Manager that manages access to the display subsystem and seamlessly renders 2D and 3D graphic layers from multiple applications.
- LibWebCore, a web browser engine which powers both the Android browser and an embeddable web view.
- SGL, the underlying 2D graphics engine.
- 3D libraries, an implementation based on OpenGL ES 1.0 APIs.
- FreeType, bitmap and vector font rendering.
- SQLite, a lightweight relational database engine available to all applications through the framework API.
Android Fragmentation

- 7 API levels in active use starting from Android 1.6 to 3.0
- APIs are forward compatible
- Device features vary
- App must specify requirements

- Prior 3.0 versions intended for phones
- 3.0 is intended for tablets
Application model

- Application components classes
  - Activity: UI view
  - Service: background component
  - Content provider: shared app data
  - Broadcast receiver: reacting to system events

- Component-level garbage collection
Android SDKs

- Java SDK
- Native SDK
- Eclipse IDE for Linux, Mac, Windows
- Visual UI builder

More information:
- developer.android.com/sdk/index.html
Android: Key Components

- **AndroidManifest.xml.** This XML document contains the configuration that tells the system how the top-level components will be processed.

- **Activities.** An activity is an object that has a life cycle and performs some work. An activity can involve user interaction. Typically one of the activities associated with an application is the entry point for that application.

- **Views.** A view is an object that knows how to render itself to the screen.

- **Intents.** An intent is a message object that represents an intention to perform some action.
  - In Android terminology, an application has an intent to view a Web page, and generates an Intent instance in order to view the Web page using a URL. The Android system then decides how to implement the intent. In this case, a browser would be used to load and display the Web page.

- **Services.** A service is code that runs in the background. The service exposes methods for to components. Other components bind to a service and then invoke methods provided by using remote procedure calls.

- **Notifications.** A notification is a small icon that is visible in the status bar. Users can interact with this icon to receive information.

- **ContentProviders.** A ContentProvider provides access to data on the device.
Manifest

- AndroidManifest.xml defines the app
  - Activities and components
  - Device features
  - Intent filters, actions to associate with
  - Permissions needed / required
  - Minimum API support
Processes and Threads

- When the first of an application's components needs to be run, Android starts a Linux process for it with a single thread of execution.
- Can spawn additional threads
  - Thread class, Looper, Handler, ...
- RPC for interprocess communications
  - Java-based IDL: AIDL
Activity states

- An activity has four main states:
  - Active. An activity is active when it is in the foreground of the screen and at the top of the activity stack.
  - Paused. An activity is paused when it has lost focus, but is still visible. A paused activity is alive, but can be destroyed by the system if memory needs to be freed.
  - Stopped. An activity is stopped when it is obscured by another activity. The stopped activity retains its state, but it is no longer visible and can be destroyed by the system when memory is needed.
  - Destroyed/Inactive.

- If an activity is paused or stopped, the system can remove the activity from memory. This can happen in two ways, the system can ask the application to finish or simply destroy the process.
Activity starts

User navigates back to your Activity

Activity is running

New Activity is started

onPause()

Your Activity is no longer visible

onStop()

onDestroy()

Activity is shut down

Your Activity comes to the foreground

Process is killed

Other applications need memory

onRestart()

onCreate()

onStart()
Media Framework

- Android use OpenCore as core component of Media framework
- OpenCore supports MP3, AAC, AAC+, 3GPP, MPEG-4 and JPEG,
Media Framework
Media Framework

- Example:
  - `MediaPlayer mp = new MediaPlayer();`
  - `mp.setDataSource(PATH_TO_FILE);`
  - `mp.prepare();`
  - `mp.start();`
Media Framework

- OpenCore lib has a C/S Architecture.
- MediaPlayer invoke JNI to manipulate client.
- The client request to the server to control hardwares.
Media Framework

Java Scope
- android.media.MediaPlayer

MediaPlayer JNI

Native Scope
- MediaPlayer

Media Server

MediaPlayer Service
- PVPPlayer
- MidiPlayer
- VobisPlayer

AudioFlinger

CameraServices
Media Framework

Applications

Application

MediaPlayer

JNI

MediaPlayer

Binder IPC

AudioFlinger

Dynamic load

libaudio.so

ALSA

Kernel Driver

Libraries

MediaPlayer

Framework

Linux Kernel
Activity Manager

- Each user interface screen is represented by an Activity class.
- Each activity has its own life cycle.
- Activity uses Intent object to jump between them.
Intent and Intent filters

- Intent activates activities, services, and broadcast receivers.
- Intent can be used in explicit way or implicit way.
- The implicit way depends on parameters: **Action**, **Data**(url and MIME type), **Category**
Content manager

- Manage data
- Client+server architecture.
- **Content Resolver** provides API interface for applications.
- **Content Providers** is the server managing the DB tables and database content with different application.
Content manager

- URI identifies the data or the table

- A: Standard prefix indicating that the data is controlled by a content provider.

- B: The authority part of the URI; it identifies the content provider.

- C: The path that the content provider uses to determine what kind of data is being requested.

- D: The ID of the specific record being requested.
Service Lifecycle

Service is started by `startService()`
- `onCreate()`
- `onStart()`
- Service is running
  - The service is stopped (no callback)
- `onDestroy()`
- Service is shut down

Service is created by `bindService()`
- `onCreate()`
- `onBind()`
- Client interacts with the service
- `onRebind()`
- `onUnbind()`
- `onDestroy()`
- Service is shut down
Security and permissions

- Security between applications and the system is enforced at the process level through standard Linux facilities.
- Application cannot disrupt other applications, except by explicitly declaring the permissions for it.
- Each Android package is given its own unique Linux user ID.
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Android</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>Java, native code with JNI and C/C++</td>
<td></td>
</tr>
<tr>
<td>Network scanning</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Network interface control</td>
<td>Limited</td>
<td></td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes (services)</td>
<td></td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>HTML 5</td>
<td>Yes, support depends on version</td>
<td></td>
</tr>
<tr>
<td>SIP API support</td>
<td>Limited</td>
<td></td>
</tr>
<tr>
<td>Open Source</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Certificate, Android store</td>
<td></td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Some fragmentation</td>
<td></td>
</tr>
</tbody>
</table>
Update: Windows Phone 7 Fragmentation

- WP7 updates not available to all devices
- Microsoft update development process
  - OEMs work with Microsoft to customize the update with their handset firmwares
  - The carriers (who also have code on the OEM phones) check these updates
  - Same as with Android, Google tightening control
- Recent problems: phones bricked (out of battery when updating)
- Older problems: excessive background data transfer, update problems
Device Internals


- iPhoneWiki

- iPhone architecture
Kindle SDK (KDK)

For developing active content with Kindle

The KDK is comprised of two sets of APIs:

- Java version 1.4 Personal Basis Profile (PBP) APIs for mobile devices. PBP JavaDocs can be found at http://java.sun.com/javame/reference/apis/jsr217/
- Kindle custom APIs which complement the PBP APIs and provide UI components, JSON and XML parsers, HTTP and HTTPS networking, secure storage, and other features.

- Other APIs like audio and dictionary access will be available in a future release of the KDK.
- KDK JavaDocs can be found at http://kdk-javadocs.s3.amazonaws.com/index.html.
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Kindle SDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>Java, Personal Basis Profile</td>
</tr>
<tr>
<td>Network scanning</td>
<td>No</td>
</tr>
<tr>
<td>Network interface control</td>
<td>No</td>
</tr>
<tr>
<td>Background processing</td>
<td>No</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>No</td>
</tr>
<tr>
<td>Memory management</td>
<td>Limited</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Limited secure storage</td>
</tr>
<tr>
<td>Location information</td>
<td>No</td>
</tr>
<tr>
<td>HTML 5</td>
<td>N/A</td>
</tr>
<tr>
<td>SIP API support</td>
<td>No</td>
</tr>
<tr>
<td>Open Source</td>
<td>No</td>
</tr>
<tr>
<td>3rd party application installation</td>
<td>Kindle DRM</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Not fragmented</td>
</tr>
<tr>
<td>Development</td>
<td>Android Linux</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Java, native code with JNI and C/C++</td>
<td>Java, Native code with JNI and C/C++</td>
</tr>
<tr>
<td>Network scanning</td>
<td>Yes</td>
</tr>
<tr>
<td>Network interface control</td>
<td>Limited</td>
</tr>
<tr>
<td>Background processing</td>
<td>Yes (services)</td>
</tr>
<tr>
<td>Energy and power monitoring and control</td>
<td>Yes</td>
</tr>
<tr>
<td>Memory management</td>
<td>Yes</td>
</tr>
<tr>
<td>Persistent storage</td>
<td>Yes</td>
</tr>
<tr>
<td>Location information</td>
<td>Yes</td>
</tr>
<tr>
<td>HTML 5</td>
<td>Yes, support depends on version</td>
</tr>
<tr>
<td>SIP API support</td>
<td>Limited</td>
</tr>
<tr>
<td>Open Source</td>
<td>Yes</td>
</tr>
<tr>
<td>Level of fragmentation</td>
<td>Some fragmentation</td>
</tr>
</tbody>
</table>
Application Trends

- WP7
  - Native apps, cloud integration

- iOS
  - Native apps
  - Potential for Web apps

- Android
  - Native apps

- WebOS
  - Web apps with HTML5

- Blackberry
  - Native and Web apps
Web apps
HTML5

- HTML 5 is the next version of HTML
  - The first public working draft of the specification available in January 2008 and completion expected around 2012

- Improvements
  - Web Socket API, advanced forms, offline application API, and client-side persistent storage (key/value and SQL).

- HTML 5 support divides the platforms.
  - The iPhone platform has a very good support for HTML 5.
## Browser support

<table>
<thead>
<tr>
<th>Platform</th>
<th>Browser</th>
<th>Rendering engine</th>
<th>Flash</th>
<th>Widget engine</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>Android</td>
<td>WebKit</td>
<td>Yes (for Android 2.1)</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Apple iPhone</td>
<td>Apple iPhone Safari</td>
<td>WebKit</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BlackBerry</td>
<td>BlackBerry</td>
<td>Mango</td>
<td>No</td>
<td>Yes</td>
<td>Proprietary push technology</td>
</tr>
<tr>
<td>Opera Mobile</td>
<td>Symbian S60, Windows Mobile</td>
<td>Presto</td>
<td>Yes</td>
<td>Yes</td>
<td>Opera Mini supports other browser platforms using a proprietary proxy</td>
</tr>
<tr>
<td>Symbian S60</td>
<td>Symbian S60</td>
<td>WebKit</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Windows Mobile</td>
<td>Internet Explorer</td>
<td>Trident/MSHTML</td>
<td>Yes</td>
<td>No</td>
<td>Silverlight</td>
</tr>
</tbody>
</table>
JavaScript access

- The Open Mobile Terminal Platform (OMTP) group defines requirements and specifications that aim towards simpler and more interoperable mobile APIs.
- BONDI defines requirements governing Device Capability access by JavaScript APIs to promote interoperability and security of implementations.
- The recent 1.1 release of BONDI is compliant with the W3C Widgets: Packaging and Compliance specification.
BONDI Architecture

- Website
- Browser
- Widget
- Widget User Agent

Web Engine (WebKit, ...)

- JavaScript Extension
- JavaScript API and Access Control
  - Application Invocation, Network, Messaging, Communication Log, Media Gallery, Media Recording, Personal Information, Persistent Data, Location, User Interaction, Device Status, System Events, Policy Management, API Management, Extensions
- Device Capability Access Control
- Operating System

BONDI code
System bus

- An asynchronous system-wide event bus is a basic solution for interconnecting various on-device components.
- There is no single standard for this.
  - Android and Java ME use Java-specific events (Android Intent filtering)
  - MeeGo uses D-BUS
  - Palm's WebOS W3C Events
- One particular trend is to utilize URI-based conventions for naming system resources and services.
  - This is extensively used in the Nokia Platform Services, WebOS and the BONDI architecture.
Summary

- Fragmentation is a current problem
  - Device, standard, implementation
  - Standards efforts are addressing this (JSR 248, ..)
- APIs seem to be converging
  - Java is one of the key languages
    ✤ Java ME poses significant problems regarding local storage and access to device resources
  - Android allows better access (Java)
  - MeeGo, Qt, Symbian allow low-level access
  - SQLite, OpenGL ES, XML, Web services and REST
  - Browsers (HTML5) and Flash