Mobile Applications and Services
FALL 2014

- Mobile Platform Landscape -

Navid Nikaein
Mobile Communication Department

This work is licensed under a CC attribution Share-Alike 3.0 Unported license.
Early Stage
New Take-Alon
Telephone Give You
Pushbutton Calling to Any
Number

This amazing phone system could handle thousands of calls simultaneously, patching you directly into a phone exchange.

By John N. Free

Illustration by Thaddaeus Rimbord and Studio Studies

Take-along Telephone back in 1973!

New TAKE-ALONG TELEPHONES Give You Pushbutton Calling to Any Phone Number
Wireless Evolution

Motorola DynaTAC8000X 1983

Analog Voice

iPhone original 2007

Digital Voice + 100 Mbps Data + Multimedia + …

Galaxy SII 2011

Bionic 2011 (LTE)

iPad 2 2011

Kindle fire 2011

iPad air 2014
Landscape of Mobile Platforms

- Mobile Internet/Web: A Little bit of history
- Wireless System Evolution
- Mobile Application
- Mobile OS
- Mobile platform run time environment
- Current Trends
- Building a successful app evolution
- App stores
- Implication of technology
MOBILE INTERNET : A LITTLE BIT OF HISTORY
From PDA to Smartphone

- **Apple Newton**: 1st PDA introduced in 1992
  - Personal data management and productivity applications
  - No phone capability

- Offline and regular synchronization/backup as the initial model

- **Online** (or at least occasional online) went through several stages:
  - dial-up services (modem) for Internet connectivity
  - IrDA for synchronization with other devices
  - Bluetooth for synchronization with other devices
  - cell phone data services (GPRS, EDGE, 3G, 3G+) as advanced dial-up
  - Wi-Fi for using hotspots and home/office networks

- **Smartphone seems to be the new term for PDA**
  - Advanced computing ability and connectivity than that of a contemporary basic featured phone
  - Runs a complete OS
  - SDK platform for application developments

©Navid Nikaein 2014
Limited on-line Access

- **Objectives**
  - **User**: Easily enjoy Internet services on mobile phone
  - **Operators**: A way to attract new users and increase revenue (per-packet billing)

- **Combining access, content and service within operators’ domain**

- **Attractive in early stages and for several reasons**
  - Easier to implement and control
  - Easier to develop market models

- **However, often have a limited life-span**
  - Content and service maintenance is very expensive
  - Content and service are operator-dependent (user may leave)
  - Content and real world disconnection

- **Example**
  - iMode
  - WAP 1.x and WAP2.x (wireless application protocol)
iMode

- Mobile internet platform
- Mixing access, content, and service
- Developed and deployed by DoCoMo

**i-mode Timeline**

<table>
<thead>
<tr>
<th>Date</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb. 1999</td>
<td>i-mode service</td>
</tr>
<tr>
<td>Dec. 1999</td>
<td>i-mode handsets</td>
</tr>
<tr>
<td>Jan. 2001</td>
<td>i-appli service</td>
</tr>
<tr>
<td>Jul. 2001</td>
<td>i-area service</td>
</tr>
<tr>
<td>Nov. 2001</td>
<td>i-motion service</td>
</tr>
<tr>
<td>Jun. 2002</td>
<td>i-shot service</td>
</tr>
<tr>
<td>Jan. 2003</td>
<td>i-motion mail service</td>
</tr>
<tr>
<td>Jul. 2004</td>
<td>Osaifu-Keitai/wallet service</td>
</tr>
<tr>
<td>Sep. 2005</td>
<td>i-channel service launched</td>
</tr>
<tr>
<td>Nov. 2005</td>
<td>ToruCa info-capture</td>
</tr>
<tr>
<td>Dec. 2005</td>
<td>iD credit card brand</td>
</tr>
<tr>
<td>Apr. 2006</td>
<td>DCMX mobile credit services</td>
</tr>
<tr>
<td>Oct. 2006</td>
<td>keyword search service</td>
</tr>
</tbody>
</table>

©Navid Nikaein 2014
iMode

- Successful in Japan not only for market but also for social reasons
  - talking on the phone in the public is frowned
  - data-oriented services (chatting, messaging) are important
- No other mobile operator was able to establish a working market
- Leverage Internet technologies (packet-switched)
  - IP, TCP, HTTP, java, ssl
- Micro web browser
  - Compact HTML (cHTML), which is a subset of HTML
- Example in France
  - Vodafone Live (J-Phone/J-Sky) by SFR and ORANGE
  - iMode for Bouygues Télécom
WAP

- Internet content and advanced data services on wireless clients
  - no content landscape/environment as in iMode
  - WAP started with zero content!

- Designed for micro browsers

- Works across all wireless network technologies
  - Standardized by the WAP Forum (founded in 1997 by Ericsson, Motorola, Nokia, and others)

- The WAP specification includes the following:
  - WML (extended subset of HTML)
  - WMLScript (incompatible with JavaScript)
  - WBXML (binary encoding of XML)
  - WBMP (derived from PNG but incompatible)
  - Communication protocols (WCMP, WDP, WTP)

- New specifications for almost any Web standard!
WIRELESS SYSTEM EVOLUTION
Mobile Phone/Smartphone technologies

**Mobile/Cell phone technologies**
- GSM as the main technology in Europe and everywhere
- CDMA as the main technology in the U.S.
- 3G/3G+ for faster speeds (W-CDMA/UMTS/HSPA or CDMA2000)
- 4G for equivalent speed as the fast Ethernet (OFDMA/LTE)

**Several sub-technologies and sub-generations developed**
- GPRS for GSM networks (often referred to as 2.5G)
- EDGE for GSM networks (often referred to as 2.75G)
- EV-DO for CDMA2000 networks

**Devices often are limited to one family of standards**
- iPhone started as a GSM phone, now 3G/3G+/Wifi
- Kindle started with EV-DO and now 3G/Wifi for 150€ !
Wireless Systems Evolution

Most operators go to HSPA Evolution before going to LTE
Heterogeneous Network (HetNet) Deployment

Source: Jie Zhang
Heterogeneous Network (HetNet) Deployment

Number of connected devices

Divers use-cases

Source: Ericsson

©Navid Nikaein 2014
Network Evolution

- Evolution from separate packet-switched and circuit-switched core sub-domains to one common IP core
- New IP-based mobile core introduced in LTE: Toward ALL-IP-Architecture
- IPv4 towards IPv6 (Exhaustion of IPv4 addresse, Feb 2011)
  - Larger address space, multicasting, Stateless address autoconfiguration, Network layer security, mobility
User Evolution

- **User**
  - Human
  - Machine
  - Community

- **Heterogeneous user access anytime anywhere**
  - NFC, BT, WLAN, Wifi Direct, 2G/3G/4G

- **Heterogeneous user mobility**
  - Fixed, nomadic, mobile

- **Seamless**
  - Connectivity
  - Service
  - Content
MOBILE APPLICATIONS
Mobile computing

- Human-computer interaction that involves
  - Mobile communication
  - Mobile hardware
  - Mobile software

- Three classes of mobile computing
  - Portable computer
  - Mobile devices
  - Wearable devices
Mobile App

- A program that runs on a Internet-capable mobile and/or wearable device

- Many player
  - Apple, Google, Windows, Samsung, ...

- Distribution
  - Apple App store, Google Play, Windows phone store, Samsung Apps store, Blackberry world, ...

- Popular apps
  - Facebook, youtube, google maps, instagram, skype, ...
Mobile App

What powers the mobile app?

Application

Application Services
- Compute
- Storage

Management Services
- Network
- Database
What powers the mobile app?

Rich set of capabilities

- Location detection: GPS, WiFi, cell towers
- Orientation: direction from a digital compass
- Device positioning & motion: from an accelerometer
- Audio: input from a microphone; output to speaker
- Video & image: capture/input from a camera (dual and back)
- Device connections: through Bluetooth between devices
- Proximity: device closeness to physical objects
- Ambient Light: light/dark environment awareness
- RFID/NFC reader: identify & track objects with broadcasted identifiers
- Multi-touch sensors
- Haptic feedback: “feel” different surfaces on a screen
- Biometrics: retinal, fingerprint, etc.
- Push: real-time notifications “instant” to user
But with a different usage pattern

Establishing a Mobile Mindset

- Small Screen
- Battery-Powered
- Inconsistent Connectivity
  - Limited rate
  - Always On, Almost Connected
- Text/voice/action Entry
- Changing User Context
- Focused user interface
- Handed operation
Usage Pattern
Anywhere Anytime

- Personal handheld device
- Identifiable and locatable
- Wakable by user or network
- Always on your pocket, always on, always connected
- Minimal interaction in the order of few seconds
- Minimal attention
- Hand-free interaction in any situation and position
- Changing user context
- May modify the social experience both positively and negatively
Usage Pattern

But, ...

- Time and place play an important role in how mobile phones are used.

- Design for anywhere and anytime (mostly for killing the time)
  - 84% use them at home
  - 80% use them during miscellaneous downtime
  - 74% use them while waiting
  - 69% use them while shopping
  - 64% use them at work
  - 62% use them while watching
  - 47% use them during their travel
  - ...

- It is unlikely to get someone’s full attention
  - Partial attention or a short burst
  - Even in distraction-free environments where focus is possible, a simplified mobile experience making people feel comfortable and relaxed
Mobile Device types

- **Basic Phones**
  - Voice, SMS

- **Feature Phones**
  - 2 - 2.75G
  - Camera, ring tones, SMS novels

- **High-End Phones**
  - 3G
  - Email, multimedia, mobile TV, MP3, location

- **Smart Phones/Communicators**
  - 3.5 - 4G
  - Mobile TV, multimedia, productivity, multiplayer games, email, calendar, maps, location

**Smart Phones**

- Interactive game, VoD, wireless VPN

**New Users**
The Next Billion
Mobile Device types

Enterprise-centric

Media-centric

Productivity-centric

Price-centric (low-cost vs. luxury)

Educational-centric

→ Some are converging
MOBILE OS
OS Wars

- Does mobile market follows the PC market?
- There is a fragmentation in Mobile OS
  - Symbian
  - WP
  - iOS
  - Android
  - Palm OS/ WebOS
  - Meego
  - Tizen
  - Firefox OS
  - RIM/Blackberry
Mobile OS Landscape

- **Mass-market proprietary phones**
  - Closed / Featured / low cost phones
  - Single processor (80% modem + 20% Apps)
  - Built-in applications
  - No native SW after the production except for java
  - Proprietary OS Example
    - RTOS

- **Niche-market Smartphones**
  - Open / smart / high end phones
  - Multi-processor (nxModem + mxApp)
  - Built-in applications
  - Native SW & Apps can be added
Symbian: does not exist anymore

  - Descendent of Psion’s graphical multitasking OS, called EPOC, based on ARM processor
  - EPOC renamed to Symbian in 2000, and used 1st in Nokia communicator 9210
  - Versioning: Symbian 9.1/2/3 => Symbian 1/2/3/4
  - Endorsed by Nokia (for mid-range smartphone) and some other OEMs (SMEC, Samsung)

- **Non-brand specific, brand owend**
  - Brand owned by Nokia, but still operating as a separate open-source driven foundation
  - Symbian 3 is fully open-source
  - Over engineered, not trivial for non-engineers

- **As of Feb. 2011 → Nokia migrate from Symbian to windows phone OS**

- **SDK / Apps**
  - C++, now QT+
  - Nokia OVI store, and other brand specific store
Windows Phone / Nokia

  - First on PDA, Pocket PC from 2000 -2002 based on Windows CE (kernel)
  - Windows mobile from 2003 - Jan 2010
    - Windows CE kernel + Windows Mobile OS
    - Target mostly the professionals, now xbox, social, multimedia

- **Windows Phone 7 from October 2010**
  - Replacement of Windows Mobile
  - Move to consumers with WP7, Xbox, multimedia, social network, advertising platform
  - Endorsed by a large number of OEMs

- **WP 8 on June 2012**

- **Nokia/Windows lumia, Samsung focus 2 ,HTC rader/titan/trophy, Dell Venue, LG Quantum**

- **Non-brand specific and non open source**

- **SDK / APPs**
  - Windows Mobile: C/C++ SDK (Visual Studio), C# through .NET
  - Windows Phone: entirely managed code (C#), Sliverlight
  - Windows phone store (formerly windows marketplace)
iOS

  - Previously iPhone OS
  - Started with iPhone 2G (Edge) in 2007
  - Then iPod touch later in 2007
  - Then iPhone 3Gs (UMTS) with faster HW
  - OS ported on other devices and become iOS
  - Initially no App Store, only Web Apps
  - App store enabled by version 2G together with first SDK in 2008

- **Brand owned, very closed**
  - Scaled-down OS X, Touch version of MAC app framework

- **SDK / Apps**
  - Objective-C, and C
  - Xcode
  - Apple App Store
Android

- **History** ([http://en.wikipedia.org/wiki/Android_%28operating_system%29](http://en.wikipedia.org/wiki/Android_%28operating_system%29))
  - Developed by Android inc., acquired by Google
  - Closed until 2007, and opened in 2008
  - Open handset alliance in 2007
    - HW vendors, OEMs, operators, service providers
  - Start from Android G1, now more than 100 models
  - Praised as “the only open OS” at first
  - Now the new “evil”

- **Open OS, non brand-locked, open-source**
  - Endorsed by many OEMS
  - Custom Linux kernel + new libs
  - Google custom VM “Davlik” + java app framework

- **SDK / App**
  - Originally: java-based: eclipse, ADT + emulator
  - Now also an NDK (performance critical part of app)
  - Android play
  - App inventor: does not exist anymore
Open WebOS

  - A.K.A Garnet OS, used in Palm pilots, (1st PDA), ARM processor
  - Enable touch-screen apps
  - Successful App market long before App store
  - PalmOS => Garnet OS => ALP => WebOS
  - Acquired by HP -> released as open-source

- **Web-centric platform**

- **Open OS, owned by Palm inc. (HP)**
  - Linux + webkit + custom HTML+mojo JS FW

- **SDK / APP**
  - PDK/C++, JS/HTML/CSS
  - Brand specific stores
MeeGo OS

• **History** ([http://en.wikipedia.org/wiki/MeeGo](http://en.wikipedia.org/wiki/MeeGo))
  - Maemo by Nokia
  - Moblin branch from maemo by Intel
  - Remerged between Maemo+ Moblin

• **Open source**
  - Built on the top of Debian & Fedora
  - Maemo/Moblin Linux distribution
  - Linux+ middleware+QT+ Handset/netbook UX
  - Telephony libs come from Nokia

• **SDK/APP**
  - C++
  - QtCreator (embedded in Qt SDK Nokia)
  - Brand app store
**Tizen**

- **History**
  - Linux foundation, Tizen association
  - Samsung and Intel

- **Open source OS, and closed source SDK**
  - Followup of MeeGO, LiMo, and BADA

- **SDK/APP**
  - Web-based: HTML5, CC, JS
  - Native: C and C++

### Diagram

```
  Tizen (Intel/Linux Foundation)
     /   \
    /     \   
  LiMo (Linux foundation/Samsung)  Mer (Mer project/Qt-based)
     |       |
  MeeGo (Nokia/Intel)             mer
     |   |
  Moblin (Intel)  LiMo            Mer
        |  |                        |
  Maemo (Nokia)  Moblin (Intel)  Mer
```

- **Nemo Mobile**
  - (Open community)
  - Linux kernel + mer + UI

- **Sailfish OS**
  - (Jolla)
  - Linux kernel + mer + Jolla UI

- **Tizen NG**
  - (Intel/Samsung/Linux Foundation)
Firefox OS

- History
  - Boot to Genko (B2G)
  - Developed by Mozilla

- Opensource and free software
  - Linux-based kernel
  - Open web APIs (expose safely device and OS capabilities)

- SDK/APP
  - Firefox and simulator add-on
  - HTML5, CSS, JS, C++

- Firefox marketplace
Others

- **RIM Blackberry**
  - RTOS
  - Custom VM
  - Custom SDK, app store,
  - Java

- **Qualcom Brew**
  - RTOS
  - Custom framework with DLL support
  - Custom SDK, app store
  - Native and Java

- **Google Chrome OS**
  - Linux + webkit chorme core
  - JS and web framework

- **LiMo, openMoKo, Ophone,**
## Platform development Environment

<table>
<thead>
<tr>
<th>Platform</th>
<th>Language</th>
<th>Debuggers</th>
<th>Emulator</th>
<th>IDE</th>
<th>Platform Deployment</th>
<th>Package Installer</th>
<th>Development Tool Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>Java, C/C++</td>
<td>Integrated in Eclipse</td>
<td>Free Emulator</td>
<td>Eclipse plugin</td>
<td>Android only, Dalvik VM</td>
<td>apk</td>
<td>Free, Fee for Key</td>
</tr>
<tr>
<td>Blackberry</td>
<td>Java</td>
<td>Integrated in JDE</td>
<td>Free Emulator</td>
<td>JDE-BlackBerry</td>
<td>BlackBerry on(RIM API)</td>
<td>alx, cod</td>
<td>Free</td>
</tr>
<tr>
<td>BREW</td>
<td>C/C++ API</td>
<td>ARM Visual Std</td>
<td>No emulator</td>
<td>Visual std.</td>
<td>Handset Specific brew</td>
<td>OTA</td>
<td>Fee</td>
</tr>
<tr>
<td>iOS</td>
<td>Objective-C/C</td>
<td>Xcode IDE</td>
<td>Xcode IDE</td>
<td>Xcode</td>
<td>iPhone, iPod touch, iPad</td>
<td>App store</td>
<td>Free, Fee for key</td>
</tr>
<tr>
<td>JME</td>
<td>Java</td>
<td>Yes</td>
<td>Free emulator</td>
<td>Eclipse, Netbeans</td>
<td>Yes, through VMs</td>
<td>Jad/Jar, RRC(palm)</td>
<td>Free</td>
</tr>
<tr>
<td>Symbian</td>
<td>C++</td>
<td>Yes</td>
<td>Free emulator</td>
<td>Many choices</td>
<td>Compile per target device</td>
<td>SIS dev.</td>
<td>Commercial/Free</td>
</tr>
<tr>
<td>Web OS</td>
<td>JS,CSS, HTML, C/C++</td>
<td>Yes</td>
<td>Free emulator</td>
<td>Eclipse</td>
<td>Web OS/Palm</td>
<td>OTA, app store, URL, .ipk</td>
<td>Free</td>
</tr>
<tr>
<td>Mobile Windows</td>
<td>C/C++, .NET</td>
<td>Yes</td>
<td>Free emulator</td>
<td>Visual std.</td>
<td>Windows Mobile / CE</td>
<td>OTA, CAB, ActiveSync</td>
<td>Free CMD, eMbeDded VC++</td>
</tr>
</tbody>
</table>
Open OS Wars

Source: Openplug
Conclusion

- Fragmentation and application execution environment (AAE)
- Fragmentation is a big issue
  - Previously for device manufacturer (OEM) to maintain their SW (cost)
  - Now for application developer market
    - How to develop an application and deploy it to many platforms
- The OS is not everything
  - AAE on the top of OS?
- Is HTML5 and Web app are the solution to this?
  - Typekit, webworkers, web GL, webkit
- Most platforms are adopting the same browser core
  - Webkit
- WAC, phonegap, AIR, are some examples
Web apps vs Native apps

MOBILE PLATFORM RUN TIME ENVIRONMENT
Application Type

- Three main families:
  - Native app
  - Web app
  - Hybrid app
    - Mobile app
    - Cross-platform app
    - Standalone runtime
    - Mobile cloud runtime (MBaaS)

- Question:
  - What are the decision drivers?
Native Mobile Platform 1

Browser
(layout, rendering, DOM, Script)

Native Applications

Built-in Applications

API / Application Services / Libraries

HTTP

OS
HAL

runtime

WLAN/BT

Cellular Modem

Hardware

I/O

Memory

Initial Source: Erik Wilde
Native Mobile Platform 2

- Mobile Browser (Safari, chrome, android)
- Native Applications
- Built-in Applications
- API / Application Services / Libraries
- runtime
- OS
- HAL
- Hardware
- WLAN/BT
- Cellular Modem
- I/O
- Memory

Initial Source: Erik Wilde

©Navid Nikaein 2014
Web as a Platform

- Code is delivered on demand using Web standards
  - HTML, CSS, JS, DOM, HTTP

- Web technologies are migrating into the OS
  - More complete packages are available (WebKit)
  - WebOS treats Web technologies as the native API

- ChromeOS is using the browser as a platform
  - browser = WebKit + user controls (a.k.a. chrome)
Web as a Platform

Browser

Web App + embedded Browser

Built-in Applications

API / Application Services / Libraries

runtime

Webkit

HTTP

OS

HAL

WLAN/BT

Cellular Modem

Hardware

I/O

Memory

Initial Source: Erik Wilde
A Quick Comparison:
Pure HTML
What it looks like?

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="http://./styles.css" media="screen">
    <meta charset="utf-8">
    <script src="js/scripts.js"></script>
    <title>hello, MbbServ</title>
</head>

<body>
    hello, MbbServ Fall 2012
</body>

</html>
```

What would be the performance of a web application?

- Network time to fetch the application (JavaScript + HTML)
- JavaScript parse time
- Code execution time to fetch the data and render the home page of your application

What are the bottlenecks?

Asynchronous data fetch after initial download

```html
<script type="text/JavaScript">
function loadFile(url) {
    var script = document.createElement('SCRIPT'); // dom element
    script.src = url;
    document.getElementsByTagName('HEAD')[0].appendChild(script);
}
</script>

<script type="text/JavaScript">
function loadFile(url) {
    function callback() {
        if (req.readyState == 4) { // 4 = Loaded
            if (req.status == 200) { // good
                eval(req.responseText);
            } else
                // Error
        }
    }
    var req = new XMLHttpRequest();
    req.onreadystatechange = callback;
    req.open("GET", url, true);
    req.send(""; }
</script>
```
Current Web Capabilities

- **Web technology is opening to device support**
  - keyboard/mouse perspective is built into numerous technologies
  - output is visual and bitmapped

- **Many Web pages rely on non-standardized technologies**
  - *Flash* for rich media (mostly video playback)
  - *Silverlight* for *Rich Internet Application (RIA)* implementations

- **Web technology is limited to online scenarios**
  - offline capabilities are now added in various places
  - most Web-based apps by their very nature are online-oriented

- **Monetization model for a web app is not clear**
  - How to sell them?

- **HTML5 /CSS3 and evolving set of technologies solve partly such limitations**
**Missing Pieces**

- **Non-traditional interactions**
  - (multi-)touch screen for pointing interactions
  - accelerometer for measuring device movement
  - vibration for giving tactile feedback

- **Access to device hardware**
  - camera, compass, graphics, ...

- **Access to additional information sources**
  - phone API for phone calls and messaging
  - contacts, calendar, social networks, ...

- **Special graphics capabilities**
  - essential for advanced rendering (gaming, 3D visualization)
  - non-traditional bitmapped displays such as e-ink
Hybrid Apps: Cross platform environment

- **Write once, run anywhere (WORA)**
- **Build app with web tech, wrap it up, run**
- **Application and runtime compiled into a native application**
  - less dependency on installation status of the client
  - only reasonable with a lightweight runtime
- **PhoneGap acts mostly as glue for native functionality**
  - exposing currently unsupported functionality as JavaScript APIs
  - the actual functionality is supported/implemented in the OS
- **Other platforms:**
  - Worklight
  - Appmobi
  - Rhomobile
  - Appcelerator
  - widgetPad
  - MoSync
PhoneGap

PhoneGap Apps (HTML/CSS/JS)

Glue

API / Application Services / Libraries

WebKit

Built-in Applications

runtime

OS

HAL

Hardware

I/O

Memory

WLAN/BT

Cellular Modem
Hybrid Apps: Other Runtime Platform

- Write Once, run Anywhere!

- Traditional HTML supports only static content
  - initial browsers did not support scripting
  - the browser wars made scripting very heavy

- Multimedia content was required for commercial content
  - Java was an existing (but heavy) solution from enterprise IT
    - Performance and not supported by iPhone
  - Flash evolved from a more content-centric perspective
    - Security issues and support by iPhone
  - AIR engaging application beyond the browser across screen
  - SilverLight a Windows web application Platform (similar to flash)

- Today's browsers are runtime environments themselves
  - rich and fairly robust scripting environments
  - more limited access to OS resources than add-ons
  - Browser wars again appear here
  - the Browser as a Platform (ChromeOS)
# Java based Platform

## Browser

## Native Applications

## Built-in Applications

### Add-ons (BT, RMI, JDBC, etc.)

### Profile (MIDP, Personal, Foundation)

### Configuration (CDC, CDLC)

### VM (KVM, CVM)

## Hardware

### WLAN/BT

### Cellular Modem

## OS

## HAL

## I/O

## Memory
Embedded in the Browser

- Plugin architectures of browsers allow browser extensions
  - Applet, and object

- Browsers must have the extension installed for content to work
  - plugin installation is complicated and security-sensitive
  - fallback can be implemented but often is too costly and not supported

- Browser plugins are not subject to the same security policies
  - they run as native code (the plugin is installed as an OS-level library)
  - vulnerabilities of the plugin turn into vulnerabilities of the browser

- More lightweight approaches may be required for mobile devices
  - Flash Light is a version of Flash for limited devices
  - JavaFX and JavaFX Mobile are intended to replace Applets/J2ME
Browser/Flash Platform

Browser
  Flash Runtime

Built-in Applications
  API / Application Services / Libraries
    runtime

OS
  HAL

WLAN/BT
  Cellular Modem

Hardware
  I/O
  Memory

Initial Source: Erik Wilde
Standalone Runtime

- Removing the limitations of the browser
  - essentially recreating Java's *virtual machine* approach
  - often more friendly to Web technologies (built-in HTML/CSS/Scripting)

- Transitioning outside of a browser breaks the Web flow
  - standalone apps mean that there is no back button
  - it is harder to integrate these applications with others

- Most examples of standalone applications are black holes
  - they are not supposed to connect to the fabric of the Web
  - overcoming the black hole effect is easier in a browser
AIR / Flash Standalone Platform

Hybrid App  Browser  Built-in Applications

AIR/Flash Runtime  API / Application Services / Libraries

OS  HAL

runtime

WLAN/BT  Cellular Modem  Hardware  I/O  Memory
Mobile Cloud runtime

Application

Native Applications

Built-in Applications

Database server

Web server

Messaging service

API / Application Services / Libraries

runtime

HTTP

OS

HAL

WLAN/BT

Cellular Modem

Hardware

I/O

Memory
Mobile Cloud runtime

- Bring a rich computational and storage resources to the mobile users
  - Link the mobile app to the backend cloud APIs
- Better end-to-end life cycle targeting enterprise-graded apps
- Services
  - User management, storage, computation, notification, authentication, social integration, enhanced location service
### Decision Drivers

- Devices, platforms, and technologies are evolving fast
- Web-based applications will see better support in the near future
- Mobile will probably follow the path of the Web

<table>
<thead>
<tr>
<th>Decision criterion</th>
<th>Native apps</th>
<th>Mobile web</th>
<th>Cross platform tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality of UX</td>
<td>Excellent</td>
<td>Very good</td>
<td>Not as good as native apps</td>
</tr>
<tr>
<td>Quality of apps</td>
<td>High</td>
<td>Medium</td>
<td>Medium to low</td>
</tr>
<tr>
<td>Potential users</td>
<td>Limited to a particular mobile platform</td>
<td>Maximum including smartphones, tablets and other feature phones</td>
<td>Large - as it reaches to users of different platforms</td>
</tr>
<tr>
<td>App development cost</td>
<td>High</td>
<td>Low</td>
<td>Medium to low</td>
</tr>
<tr>
<td>Security of app</td>
<td>Excellent</td>
<td>Depends on browser security</td>
<td>Not good</td>
</tr>
<tr>
<td>Supportability</td>
<td>Complex</td>
<td>Simple</td>
<td>Medium</td>
</tr>
<tr>
<td>Ease of updating</td>
<td>Complex</td>
<td>Simple</td>
<td>Medium</td>
</tr>
<tr>
<td>Time-to-market</td>
<td>High</td>
<td>Medium</td>
<td>Short</td>
</tr>
<tr>
<td>App extension</td>
<td>Yes</td>
<td>Yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
Read this: http://mashable.com/category/mobile-app-trends-series/

CURRENT TRENDS
Recent Trends

- Build a Mobile App That Won't Drain Your Battery
  - No methodology for power-aware mobile app development
  - 75% of app-related battery drain originated from ad-serving
    - Apple iAds, and Google AdMob
    - Ad packages are closed
  - Apps are permitted to run in the background longer and are less likely to be shut down by the operating system (mainly in Android)
Recent Trends

- Web technologies are gaining more space in mobile app
  - Phonegap, adobe solution
  - RhoMobile, motorola solution
  - appcelerator
  - Dragonrad (mobile entreprise solution)
  - Mosync
  - Xamarin
  - ...
Recent Trends

- Development for (Certain) Tablets and wearable devices are Growing

- Proximity services and geo-targeted push notification and advertisement, and recommendation

- Mobile developers go to
  - Enterprise
  - Wearable technologies

- Mobile app analytics and data gathering

- Yet rule-based smarter app is emerging
Current Trends

- More mobile OSes
- More processing power to deploy
  - CPU and GPU and MPU
- More wireless connectivity to exploit
  - Multipath TCP → increase aggregated data rate
  - Load balancing → Wifi offloading
- More multiscreen/multidevice usage pattern
- More connected objects
- Smart Application / Artificial intelligent
### App Trends

<table>
<thead>
<tr>
<th>App Trends</th>
<th>Platform Architecture Needs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaming &amp; Video Apps</td>
<td>Powerful graphics processing, touch responsiveness</td>
</tr>
<tr>
<td>Productivity Apps</td>
<td>High performance MPUs, cross platform portability</td>
</tr>
<tr>
<td>Cloud Centric Apps</td>
<td>Low latency data sharing, web-connectivity</td>
</tr>
<tr>
<td>Social Networking Apps</td>
<td>Fast transcoding of videos and pictures</td>
</tr>
<tr>
<td>Object Recognition Applications</td>
<td>High-end image processing</td>
</tr>
<tr>
<td>Context Aware Applications</td>
<td>Smart App / Artificial intelligence</td>
</tr>
<tr>
<td>Multi Party Conferencing Applications</td>
<td>Video parallelism</td>
</tr>
<tr>
<td>Mobile Commerce Applications</td>
<td>High data security</td>
</tr>
</tbody>
</table>
Smart app

- What is and Why a smart mobile app?

- What are the ingredients for a smart app?
Smart app ingredients

- Awareness of user **environment** by adding **relevant information** on the present as a function of user **preferences/profile**
  - User becomes part of the experience
  - User experiences a better decision-making

- My environment (physical location/social context) may influence my actions

- Context is what make mobile device a powerful medium
Smart app ingredients

Smart App

Rules

Content

Context
Rules

- **Local view**
  - Event sources
  - Event filter
  - Event processing
  - Event

- **Network view**
  - Asynchronous notify & C/S approach
  - Pub sub model
  - Complex network
BUILDING A SUCCESSFUL APP: 9 RULES
Rule #1

- **Step 1: Decide what to build?**
  - I have my idea
  - But why are you building this
  - Who is the user? Not possible to build SW for everybody
  - What is the problem you are trying to solve?

- **User = you & Problem = your problem**
  - Extremely hard what the others want

- **Or, you may know some other user and know exactly their needs**
  - Build up the persona
    - Jane, 62,
    - Live in XX
    - Has three kids in YY
    - Love pictures
    - Wish to interacts with kids
  - Build up the usage scenario/use case

- **Principle 1: Know your user**
  - Who are you designing for?
  - What do you want them to do?
Rule #2

- **Step 2: visit app stores**
  - Understand what they are using know, and what they look like, and what they do
  - Find out what other people done, there is a lot of knowledge you can learn already by studying the existing apps
  - Why you are not satisfied with them
  - Observe existing work practices

- **Principle 2: innovation through copying**
  - Good artists copy, great artists steal
  - Make sure that you follow the conventions
Rule #3

- **Step 3:** explore possible solutions
  - Consider the constraints
    - small screen, touch input, typing is difficult, battery
  - Very short interaction with app: <1 min
  - Remember the user
  - Find the set of minimal features that the user needs
  - Try out new ideas before building software

- **Principle:** be minimal wrt to your user profile
  - Learn From Top Mobile Apps
  - Better a good app for small users that a poor app for big number of users
  - Then you can expand based on user feedback
Rule #4

- **Step 4: sketch**
  - Build an interface mock-up/story board/screenshots, e.g. sketchify
  - The more sketch => go deeper => become innovative after 10x
  - Showing to the user
  - Narrow your sketch down => stay with few of the sketch based on user feedback
  - Learn and getting better across sketching
  - Color, font, lines, images, balance

- **Principle: quality through quantity**
  - More time you put here fixing things, the less cost you have down the road to fix the same thing
  - Cost of making changes increases as you progress in your design
Rule #5

- **Step 5: build a prototype**
  - Wireframes (layout+behavior+flow of UI)
    - Your app on the paper: on page per screen
    - Your app on wireframes, e.g. omnigraffle similar to IB/App Inventor
      - Be pixel compatible with final version
    - Make a video of your app
  - Interactive web based prototype
  - Try your app before actually developing it
  - Use the feedback to improve the design
  - Test some basic actions
    - Give instruction to the users

- **Principle:** fail early to succeed sooner
Rule #6

- **Step 6: iterate**
  - Sometimes you need to go back and do it all again

- **Principle: nothing is precious**
  - You can throw anything away
Rule #7

- **Step 7: begin coding**
  - Bottom-up for if proof of concept needed
  - Otherwise top down with dummy data
  - First build the interface and then backend

- **Principle: top down approach**
Rule #8

- **Step 8: beta test your app**
  - Who is your beta testers: your friends
  - Mechanical turk: great tool, give your sys spec, your phone, etc
  - Ad hoc distribution / field study
  - Usability testing

- **Principle: test before your submit**
Rule #9

- **Step 9: release, bug fix**
  - Talk to users
  - Explore many designs
  - Iterate on user feedbacks

- **Principle: upgrade your app**
  - Learn from the user complains and feedbacks
  - Follow your app analytics
## Recap

<table>
<thead>
<tr>
<th>Steps</th>
<th>Principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Decide what to build</td>
<td>1) Know your user</td>
</tr>
<tr>
<td>2) Visit app stores</td>
<td>2) Innovate through copying</td>
</tr>
<tr>
<td>3) Explore possible solutions</td>
<td>3) Be minimal</td>
</tr>
<tr>
<td>4) Sketch</td>
<td>4) Quality through quantity</td>
</tr>
<tr>
<td>5) Build a prototype</td>
<td>5) Fail early to succeed sooner</td>
</tr>
<tr>
<td>6) Iterate</td>
<td>6) Nothing is precious</td>
</tr>
<tr>
<td>7) Begin coding</td>
<td>7) Top-down approach</td>
</tr>
<tr>
<td>8) Beta test your app</td>
<td>8) Test before submit</td>
</tr>
<tr>
<td>9) Release and bug fix</td>
<td>9) Upgrade</td>
</tr>
</tbody>
</table>
http://www.appbrain.com/stats/stats-index

APP STORE MARKET
Mobile Apps Stores

- **Platform vendors using apps as a marketing angle**
  - Apps market the platform
  - More apps = more use case => network effect

- **Control points owned by platform vendors**
  - Walled-garden vs. real-world
  - Merchandizing, distribution, fulfillment, transactions, policy, functionality may be cut off

- **Key advantages**
  - Simple through a unique portal
  - Safe through a trusted parties

- **Platform vendors looking to provide access to the realworld while keeping the benefits of the app model**
... a lot of players!
Example of 3rd party App Store!!
Renault Zoe

- Electric car equipped with Android
APP STORE STATISTICS
Mobile App Store Downloads, Worldwide, 2010-2016 (Millions of Downloads)

Source: Gartner (September 2013)
## App Store Statistics

<table>
<thead>
<tr>
<th>Smartphone App Statistics</th>
<th>iPhone</th>
<th>Android</th>
<th>Blackberry</th>
<th>Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total app downloads</td>
<td>27,000,000,000</td>
<td>29,000,000,000</td>
<td>2,400,000,000</td>
<td>4,100,000,000</td>
</tr>
<tr>
<td>Percent of app users who have never paid more than $1 for an app</td>
<td>45%</td>
<td>62%</td>
<td>63%</td>
<td>58%</td>
</tr>
<tr>
<td>Average number of downloaded apps per phone</td>
<td>88</td>
<td>68</td>
<td>49</td>
<td>57</td>
</tr>
<tr>
<td>Total number of apps in store</td>
<td>905000</td>
<td>850000</td>
<td>130000</td>
<td>220000</td>
</tr>
<tr>
<td>Total app store revenue in 2013</td>
<td>$6,400,000,000</td>
<td>$1,200,000,000</td>
<td>$550,000,000</td>
<td>$950,000,000</td>
</tr>
</tbody>
</table>

Source: PC Magazine, MobiThinking, IntoMobile, Google, Apple, Microsoft, Blackberry
Research Date: 2.3.2014
App Store Statistics
Apple & Google

**Price Point Distribution**
Google Play's pricing model allows for a larger number of price points (from top 100 apps as of 7/9/2013).

- **29 Different Price Points**
- **7 Different Price Points**

**Most Expensive Apps**
Few apps approach the max price point set by each marketplace (and even if you pay $999 for VIP Black on iOS, you can't activate the service until you verify that you have "assets and/or income in excess of $1 million.")

- A $99.99 price cap is imposed by Apple
- A $200 price cap is imposed by Google
App Category Market Share for All Devices

Percent of Downloads

- Games: 25%
- Entertainment: 10%
- Utilities: 7%
- Education: 5%
- Productivity: 5%
- Book: 4%
- Lifestyle: 4%
- Reference: 3%
- Healthcare & Fitness: 3%
- Travel: 2%
- Music: 2%
- Navigation: 2%
- Sports: 1%
- Business: 1%
- Finance: 1%
- Photography: 1%
- Social Networking: 1%
- News: 1%
App Category Market Share for All Devices
Apple & Google

When it comes to the top publishers for each platform, there’s little correlation between the number of apps published and the number of downloads generated (ordered by number of downloads).
Mobile app store revenue, worldwide, 2011-2017 (M$)

Paid for:
- 2011: 7,500
- 2012: 12,000
- 2013: 18,000
- 2014: 24,000
- 2015: 30,000
- 2016: 35,000
- 2017: 40,000

In-app purchases:
- 2011: 5,000
- 2012: 10,000
- 2013: 15,000
- 2014: 20,000
- 2015: 25,000
- 2016: 30,000
- 2017: 35,000

Advertising:
- 2011: 2,500
- 2012: 5,000
- 2013: 7,500
- 2014: 10,000
- 2015: 12,500
- 2016: 15,000
- 2017: 17,500
Mobile app store revenue, worldwide, 2011-2017

(%)
# App Store Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>iOS App Store</th>
<th>Android play store</th>
<th>Blackberry App World</th>
<th>WP Store</th>
<th>Open webos</th>
<th>Nokia OVI Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclusive App</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Maybe</td>
<td>No</td>
</tr>
<tr>
<td>Paid Apps</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Free Apps</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Phone Client</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Desktop Client</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>TBD</td>
<td>TBD</td>
<td>No</td>
</tr>
<tr>
<td>Multiple Device</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Billing system</td>
<td>iTunes</td>
<td>Google Checkout</td>
<td>PayPal</td>
<td>CC operators</td>
<td>TBD</td>
<td>CC. Operators</td>
</tr>
<tr>
<td>Return Policy</td>
<td>No</td>
<td>24 hours</td>
<td>No</td>
<td>24 hours</td>
<td>TBD</td>
<td>No</td>
</tr>
<tr>
<td>Specific Store</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>TBD</td>
<td>Yes</td>
</tr>
<tr>
<td>Developer Share</td>
<td>70%</td>
<td>70%</td>
<td>80%</td>
<td>70%</td>
<td>TBD</td>
<td>70%+Commission</td>
</tr>
<tr>
<td>Developer Fee</td>
<td>$99 annual</td>
<td>$25 1xtime</td>
<td>$200 1xtime</td>
<td>$99 Annual</td>
<td>TBD</td>
<td>Free</td>
</tr>
<tr>
<td>Min Price</td>
<td>$0.99</td>
<td>$0.99</td>
<td>$2.99</td>
<td>TBD</td>
<td>TBD</td>
<td>&lt; €1.99</td>
</tr>
</tbody>
</table>

©Navid Nikaein 2014
Hype Cycle 2009

Figure 1. Hype Cycle for Emerging Technologies, 2009

- Cloud Computing
- E-Book Readers
- Social Software Suites
- Microblogging
- Green IT
- Video Telepresence
- Mesh Networks: Sensor
- Online Video
- Corporate Blogging
- Wikis
- Location-Aware Applications
- Electronic Paper
- Tablet PC
- Idea Management
- Web 2.0
- Social Network Analysis
- Over-the-Air Mobile Phone Payment Systems, Developed Markets
- RFID (Case/Pallet)
- Speech Recognition
- SOA
- Human Augmentation
- Context Delivery Architecture
- Quantum Computing
- 3-D Flat-Panel Displays
- Home Health Monitoring
- Public Virtual Worlds
- Human Augmentation
- Technology Trigger
- Peak of Inflated Expectations
- Trough of Disillusionment
- Slope of Enlightenment
- Plateau of Productivity

Years to mainstream adoption:
- less than 2 years
- 2 to 5 years
- 5 to 10 years
- more than 10 years
- obsolete
- before plateau

Source: Gartner (July 2009)
Gartner Hype Cycle for Emerging Technologies 2011
IMPLICATION OF TECHNOLOGY EVOLUTION ON MOBILE DEVICES
Convergence of several industries and explosion of standards have increased the complexity.

- Interactive Multimedia (audio, video, game)
- Data (SMS, MMS, IMS, email)
- Voice

Mobile Device  Access Network  Core Network  Content / Service

Source: S. Chennakeshu
Explosion of Standards

### Telecom (e.g. 3GPP)
- GSM
- GPRS
- EDGE
- WCDMA
- HSPA
- HSPA+
- LTE/ LTE-A

### Data Communication
- SCTP, TCP, IPv4, IPv6
- UDP, SIP
- SMS, MMS
- IMS (VCC, DSC, MIH)
- MIP, PMIP, HIP

### Multimedia
- MP3, MP3 pro, AAC, AAC+, WMA, RA,
- H.263, H.264,
- MPEG4, JPEG,
- WMV, RV, VC1,
- OpenGL 2.x, DRM

### Security
- AES, 3DES, SHA, MD5, RSA,
- FSA, DH, PKCS10, PKCS11, EAP,
- IKE, SSL, TLS, A5, A8, Snow..

### Connectivity
- USB (FS, HS, OTG), BT, WLAN a/b/g/n, GPS,
- MobTV (DMB, ISDBT, DVBH, MediaFlo), Pictbridge,
- SGMII, SATA, SIM, SD, MS, NFC
High end Phone Feature Evolution

2000 – High end phone
- narrowband CS voice
- 50-150 Kbps data (DL)
- SMS, MMS
- Basic PIM
- 4K color display
- Primitive games
- 2 RF bands
- 100 MIP DSP
- 100 MIP CPU
- 8 Mbytes memory
- Basic Bluetooth, IR
- Keypad based UI

Communication Device

2008 – High end phone
- wideband CS voice
- 7 Mbps data (downlink)
- 5 Mbps data (uplink)
- SMS, MMS, IMS
- Advanced PIM
- Multi format Audio/Video
- TV out
- Basic browser/games (2D)
- DRM
- FOTA (firmware over the air)
- 5-8 MP camera
- 16M color display
- 10 RF bands
- 500 MIP DSP, 500 MIP CPU
- 8 GBytes memory
- Bluetooth, WLAN, GPS, USB
- Sensor based UI

Multi Media Device

2014 – High end phone
- wideband VoIP
- 100 Mbps data (Downlink)
- Integrated messaging/mail
- Integrated realtime PIM
- Multi format Audio/Video/3D gaming
- Full browser
- DRM + e-commerce
- FOTD A
- 8-16 MP camera
- 24M color display
- Aux display connect (HDMI)
- Universal RF bands
- 3GHz Multi-cores
- 50 GBytes memory
- BT, WLAN, GPS, USB 3.0, NFC,
- Context sensitive sensor-based UI
- Personal server

True Personal Server
Mobile Device Hardware Evolution

1. MPU
2. GPU
3. Display Processing Unit
4. Video Streaming Accelerators
5. Speech Processing
6. Image and Vision Processing
7. Sensors Processing Unit
8. Low latency Cloud and Network Processor Interfaces
10. Optimized low latency Storage
11. Advanced Security and Firewalls
12. Power and Thermal Management Systems
Mobile Device Software Evolution

API / Application Services

Cellular  Connectivity  Data Comm.  Multimedia  Security Service  OS

HAL

API / Application Services

Cellular  Connectivity  Data Comm.  Multimedia  Security Service  OS

RTOS  HAL  RTOS  Drivers  HAL
BACKUPS :
MOBILE CROSS-PLATFORM
RUNTIME ENVIRONMENT
Java

- Java is a runtime environment supported on many operating systems
  - general-purpose, concurrent, class-based, object-oriented language
  - Write once and run anywhere philosophy

- Strictly speaking, Java is three different things
  - an object-oriented programming language
  - an extensive set of class libraries as supporting environment
  - the Java Virtual Machine (JVM) for executing Java Bytecode

- Sun defines and supports four editions of Java targeting different application environment
  - Java card
  - Java micro edition (JME)
  - Java standard edition JSE
  - Java enterprise edition JEE

- Java Applets are a special kind of Java apps
  - running inside a browser (can be started from the Web)
  - running inside a sandbox (security model for limiting OS access)
JavaFX

- Running on the top of JME shedding the JME brand
  - Capable of running on multiple mobile operating systems including Symbian and Window mobile

- Java platform for creating rich internet applications (RIA)
  - Common profile across all devices (i.e. desktop & mobile)
  - Drag to install, a break away from the browser
    - Drag a JavaFx widget/web app and drop it on the desktop
  - Third party graphic integration (mainly adobe)

- More Web standards for developing content
  - CSS as a way to specify the layout of content
  - Scripting as a faster way for application development

- Success and support currently unclear

FLASH Multimedia on Web

- Started as an animation platform for content developers
- Multimedia platform for creating interactivity to web pages
  - animation + video
  - includes many of the functionality now developed in HTML5
  - traditionally used for advertising, game, and bidirectional streaming
  - target Rich Internet Application (RIA)
    - a software Framework: Web application => desktop application

- Robust scripting for desktop browsers (Actionscript)
  - good strategy for deploying Flash as a browser add-on
  - gets harder with the growing variety of browsers and platforms (iphone)

- Many accessibility issues on Flash-oriented sites
  - navigation can be completely inaccessible
  - content can be hidden from search engines
  - content is less fluid than even the simplest HTML/CSS
  - poor handling of sound
Adobe Integrated Runtime (AIR)

- Adobe AIR (desktop and mobile)
  - Cross-platform browser-less runtime environment for RIA
  - Flash, Flex (SDK), Actionscript / HTML+Javascript, Ajax

- Desktop deeper integration
  - Native process, microphone, printer APIs
  - Native document/file handler
  - UDP networking, Webkit HTML5/CSS3
  - Mass storage, drag-and-drop
  - Socket server, P2P, TLS/SSL socket
  - Large windows size, Input method editor IME API, multitouch and gestures

- Development Environment
  - HTML/Ajax Dreamwears CSx or HTML editor and AIR SDK
  - Flash Builder :
  - Flex (MXML for layout, ActionScript for application logic) : free
  - Flash CSx
## Comparison: Browser vs. Desktop

<table>
<thead>
<tr>
<th>Feature</th>
<th>RIAs in the browser</th>
<th>RIAs on the desktop</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application delivery</strong></td>
<td>Applications can be easily discovered, explored, and used.</td>
<td>Installed applications have more persistence, power, and functionality.</td>
</tr>
<tr>
<td><strong>Installation</strong></td>
<td>No application installation is necessary.</td>
<td>Applications are installed seamlessly from the browser or downloaded and installed like a traditional desktop application.</td>
</tr>
<tr>
<td><strong>Application updates</strong></td>
<td>Applications are updated by pushing new content to a website.</td>
<td>AIR provides APIs that allow applications to be updated as easily as pushing new content to a website.</td>
</tr>
<tr>
<td><strong>Multiple operating system support</strong></td>
<td>Applications run on multiple operating systems and browsers.</td>
<td>AIR applications are cross-platform, so they can be installed on and run on multiple operating systems.</td>
</tr>
<tr>
<td><strong>Programming languages</strong></td>
<td>JavaScript is provided by browsers, and ActionScript® is provided by Adobe Flash® Player software.</td>
<td>Integrated JavaScript and ActionScript virtual machines are compatible with the browser.</td>
</tr>
<tr>
<td><strong>Background capability</strong></td>
<td>RIAs can run only in a visible browser window.</td>
<td>Applications can run in the background or provide notifications like traditional desktop applications.</td>
</tr>
<tr>
<td><strong>Persistence</strong></td>
<td>Activity is limited to the browser session. When the browser is closed, information is lost.</td>
<td>RIAs are installed and available on the desktop. They store information locally and operate offline.</td>
</tr>
<tr>
<td><strong>Desktop integration</strong></td>
<td>Applications are sandboxed, so desktop integration is limited.</td>
<td>Applications can access a desktop file system, clipboard, drag-and-drop events, system tray/notifications, and more.</td>
</tr>
<tr>
<td><strong>User interface control</strong></td>
<td>RIAs run within a browser window that has its own controls, branding, and integration with the desktop.</td>
<td>RIAs have a customizable user interface and desktop integration, enabling branded experiences.</td>
</tr>
<tr>
<td><strong>Data storage</strong></td>
<td>Applications have limited local storage, which the browser can destroy.</td>
<td>Applications have unlimited local storage and access to a local database, plus encrypted local storage.</td>
</tr>
</tbody>
</table>
**Sliverlight**

- Web application framework that integrates
  - multimedia
  - graphics
  - animation
  - interactivity

- Provide a single runtime environment
  - XAML + .NET FW
  - CLI language
  - Development tools

- Support Mobile devices
  - Windows Phone 7 and Symbian S60

- Features are similar to adobe Flash and AIR
  - Both for embedded and standalone applications
  - Include many HTML5 features
  - Allow Rich Internet Application