Mobile Platforms: The Clash of Ecosystems

A critical analysis of mobile platforms and the battle for dominance

Comparing Android, BlackBerry OS, BREW, iOS, Symbian, Windows Phone and webOS across ecosystems, positioning and business strategies

November 2011
About VisionMobile

VisionMobile is a leading market analysis and strategy firm, for all things connected. We offer competitive analysis, market due diligence, industry maps, executive training and strategy, on topics ranging from the industry's hottest trends to under-the-radar market sectors. Our mantra: distilling market noise into market sense.

VisionMobile Ltd.
90 Long Acre, Covent Garden,
London WC2E 9RZ
+44 845 003 8742
www.visionmobile.com/blog
Follow us: @visionmobile

About webinos

This research was partially funded by webinos, an EU-funded project under the EU FP7 ICT Programme (#257103).

Webinos is an EU-funded project aiming to deliver a platform for web applications across mobile, PC, home media (TV) and in-car devices.VisionMobile is a member of the webinos consortium. More info at www.webinos.org

License

Licensed under Creative Commons Attribution 3.0 license.

Any reuse or remixing of the work should be attributed to the VisionMobile The Clash of Ecosystems report.

Copyright © VisionMobile 2011

Disclaimer

VisionMobile believes the statements contained in this publication to be based upon information that we consider reliable, but we do not represent that it is accurate or complete, and it should not be relied upon as such. Opinions expressed are current opinions as of the date appearing on this publication only, and the information, including the opinions contained herein, are subject to change without notice.

Use of this publication by any third party for whatever purpose should not and does not absolve such third party from using due diligence in verifying the publication’s contents. VisionMobile disclaims all implied warranties, including, without limitation, warranties of merchantability or fitness for a particular purpose. VisionMobile, its affiliates and representatives shall have no liability for any direct, incidental, special, or consequential damages or lost profits, if any, suffered by any third party as a result of decisions made, or not made, or actions taken, or not taken, based on this publication.

v.1.00

Contents

A. The New Ecosystem Economics
B. Mobile Platform Shootout
C. Platform Drill-Down: from Android to webOS

Authors

Michael Vakulenko, Strategy Director
Stijn Schuermans, Business Analyst
Andreas Constantinou, Managing Director
Matos Kapetanakis, Marketing Manager

Also by VisionMobile

The complete map of the mobile industry landscape, mapping 1,350+ companies across 85+ market sectors.
Available in wallchart and PDF format.
www.visionmobile.com/maps
Contents

Key messages ................................................................................................................................................. 4

Chapter A The New Ecosystem Economics .......................................................................................... 7
Smartphones as the new mainstream ........................................................................................................ 7
The triumph of ecosystems and network effects ....................................................................................... 8
App stores are the keys to ecosystem control ........................................................................................... 9
An app store contender emerges ................................................................................................................ 10
Mounting developer acquisition costs .................................................................................................... 11
Software players put mobile operators on the defensive ......................................................................... 12
Two horses lead the race ............................................................................................................................. 12
The rising star of HTML5 .......................................................................................................................... 13

Chapter B Mobile platform shootout .................................................................................................... 14
History and Origins .................................................................................................................................... 14
Market adoption .......................................................................................................................................... 16
The islands of application development .................................................................................................. 18
Understanding platform economics .......................................................................................................... 20
Developers as major driver for platform adoption .................................................................................... 24

Chapter C Platform drill-down: from Android to webOS ................................................................. 28
Android ......................................................................................................................................................... 28
Bada ............................................................................................................................................................... 37
BlackBerry OS ........................................................................................................................................... 41
BREW MP ..................................................................................................................................................... 47
iOS ................................................................................................................................................................. 50
webOS .......................................................................................................................................................... 61
Nokia: the story of Symbian, MeeGo and Qt .............................................................................................. 65
Key messages

Smartphones go mainstream, but the devil’s in the details. Just two years ago, smartphones were viewed as expensive toys for geeks and Apple fan boys. No longer. Smartphones have entered the mainstream in developed markets, and are taking a growing proportion of device sales in more cost-sensitive markets around the globe. In the third quarter of 2011, smartphone shipments penetration surpassed 29% globally, although this figure varies widely from nearly 65% in the USA and over 50% in Europe to 19% in Asia-Pacific, 17% in Latin America and 18% in Africa/Middle East.

iOS and Android are driven by economics of demand. Handset sales are driven not by hardware features (“what the handset can do”) but the user interface and applications available (“what you can do with the handset”). Much like any smartphone platform, iOS and Android are driven by economics of demand, where the demand generated (incl. the number of applications) has a far stronger effect on sales than pure supply chain efficiencies. As of October 2011, iOS and Android are leading the way, with over 500,000 and 300,000 applications, respectively. The rest of the platforms trail far behind with order of magnitude less applications: BlackBerry has 35,000 applications, Windows Mobile 30,000 applications and Symbian 25,000 applications.

Successful platforms are a magnet for financial investment. Application platforms like iOS and Android are able to attract huge financial investments on the part of developers, investors and brands. Taking iOS as an example, and estimating that an app costs an average $30,000 to develop, the 500,000 iOS apps represent an average investment of $15B in the iOS ecosystem. This investment directly contributes to Apple’s bottom line, and its estimated $71B iOS-powered device sales for the year ending September 2011.

App stores are about controlling ecosystems, not profiting from content. The app store business is the polar opposite of the telco content business. As such, application stores like Apple App Store and Google Android Market should not be mistaken for profit centres. Instead, Apple and Google leverage app stores as ecosystem control points. With over 85% of iOS and Android downloads coming from free apps, the 30% revenue share from paid apps subsidizes the operational cost of app intake and distribution, which runs at over $1.2B to date in the case of Apple.

The rising star of HTML5. HTML5 has the potential to become a common bridge system across smartphone platform islands and the sea of feature phones. HTML5 is the only common app technology supported by Android, iOS, new versions of BlackBerry OS and Windows Phone platforms. With 225 million Android devices and 146 million iOS devices sold to date, HTML5 is supported by over 371 million mobile devices today, albeit with mixed levels of compatibility.

Microsoft, Facebook, and mobile operators have very different motivations but are all eyeing HTML5 as a technology that could help dis-intermediate app stores as content distribution silos, reducing the power of Apple’s iOS and Google’s Android platforms.
Mobile Platforms: The Clash of Ecosystems

However, in its present state HTML5 can neither challenge nor displace the leading mobile platforms. In order to become a viable alternative, HTML5 needs to move beyond being just a development tool, and to converge around a dominant solution for web application discovery, monetisation, distribution and retailing.

**Mounting developer acquisition costs.** Platforms need apps to thrive and developers are the growth engine of the smartphone ecosystems. At the same time, developer attention is scarce; developers are very critical “platform consumers” and need to make far higher investments when adopting a new platform. We estimate that the minimum acquisition cost for a publishing developer is over $2,300 in the case of Apple. As such, Apple, Google, Nokia, Microsoft and RIM have needed to invest billions of dollars in persuading developers to write apps for their platforms.

Moreover, developers are motivated by a complex set of incentives, which includes revenue potential, user reach, ability to raise funding, and the pure coolness or utility of a platform. These incentives vary widely across different types of developers and as such call for developer segmentation as a critical cornerstone of any developer strategy.

**Software players put mobile operators on the defensive.** The app innovation unleashed by smartphones puts pressure on traditional telecom profit centres, not only around value-added services, but also on core messaging and voice services.

Apple and Google combined control the user experience of nearly 400 million users through their iOS and Android platforms. Both are strategically reducing the role of mobile operators to that of “connectivity providers”. Internet giants like Facebook and Amazon are using social-centric and retail-centric strategies to profit from mobile. Start-ups such as Foursquare and Instagram have pioneered mobile-first services. Communication companies like Skype, WhatsApp and Viber put pressure on core telecom services, notably SMS and voice.

**Incumbent mobile platforms lose to next-generation challengers.** In the last decade we’ve seen over 20 mobile platforms rise and then die not being able to achieve critical mass. Next-generation platforms (iOS, Android and Windows Phone) have achieved sustainable growth by leveraging on network effects and developer economics. Legacy platforms on the other hand (Symbian, BlackBerry OS, BREW and Windows Mobile) have been designed to handset vendor requirements rather than developer requirements; all have either been discontinued or pushed into narrow market niches. Companies with strong software DNA (common in the US) now dominate the smartphone platform landscape.

**No single winner: mobile platforms will remain a multi-horse race.** The mobile market will continue to be a multi-horse race for many years to come. iOS and Android will continue to lead, dividing the market between premium (iOS) and mass-market product segments (Android). Self-reinforcing network effects, gigantic application ecosystems and the rapid pace of platform evolution make the positions of Apple and Google unassailable. Windows Phone may only challenge BlackBerry for the third place rank.

**Patent wars.** Apple and Microsoft are trying to leverage their own patent portfolios and are paying billions of dollars in patent acquisitions in an attempt to slow down the meteoric growth of Android. Apple’s strategy is to block Android sales starting with Samsung, although with mixed, regional and temporary successes. Microsoft is using an altogether different tactic, namely “patent taxes”, to coax
OEMs like Samsung and HTC away from a higher-cost Android. At the same time, Google is planning to defend Android through the pending acquisition of Motorola Mobile Devices, and its portfolio of over 17,000 patents. We expect a culmination of the patent wars in a multi-vendor consortium designed to standardise cross-licensing agreements across Android, iOS and WP7 handset vendors.
Chapter A
The New Ecosystem Economics

Smartphones as the new mainstream

Just two years ago, smartphones were viewed as expensive toys for geeks and Apple fan boys. No longer; Smartphones have entered the mainstream in developed markets, and are taking a growing proportion of device sales in more cost-sensitive markets around the globe. In the second quarter of 2011, smartphone penetration surpassed 29% globally in Q3, 2011 according to Strategy Analytics. However, smartphone shipment penetration varies widely from nearly 65% in the USA and over 50% in Europe to 19% in Asia-Pacific, 17% in Latin America and 18% in Africa/Middle East.

Over 50% smartphone penetration in developed markets, less than 20% in emerging markets
Mobile sales volumes per region in 2011 (millions of units, estimated)

Since the introduction of the iPhone in 2007, Apple has become a staple for smartphone device specs, user experience, application ecosystem, browser capabilities and content.

Nokia, a long-time leader in mobile handset sales, suffered the most from the sudden rise of Apple.
Nokia was slow to react to Apple’s application-led and developer-led strategy, and steadily lost ground in the high-end smartphone market. Nokia had no choice but to change horses in midstream, betting its future on a partnership with Microsoft and its Windows Phone smartphone platform.

The vacuum left by Nokia was quickly filled by a large group of handset makers using the Google Android software platform. This group includes “branded” handset makers like Samsung, HTC, Motorola, LG and Sony-Ericsson, as well as aggressive, cost-leading newcomers such as Huawei, ZTE, Acer, Dell and Asus.

Today, smartphone competition is a two-horse race between Apple iOS and the growing camp of Android device makers. The Android camp is able to produce hundreds of device models at a wide range of price points, from high-end superphones like the $755 Samsung Galaxy SII with NFC support, to low-cost models such as the Huawei U8100 at $100 (pre-tax and pre-subsidies).

The triumph of ecosystems and network effects

iOS and Android are winning not only by virtue of technological sophistication, but primarily by the strength of their application ecosystems. These ecosystems comprise thousands of application developers and content providers. As of October 2011, Apple’s App Store leads the way, with over 500,000 applications. Android Market is second, with over 300,000 applications.

iOS and Android both exemplify successful application platforms carefully designed to connect two disjointed markets: users and application developers. Both application platforms offer users a broad selection of applications to satisfy a wide spectrum of needs, from keeping in touch to killing time. Meanwhile, both connect developers and content providers to an audience of platform users.

From the end-user perspective, each new application adds value to the platform. From an application developer’s perspective, the platform gains value with each and every new user.

When the number of developers, applications and users on a platform reaches critical mass, the platform begins to grow exponentially. This is because of positive feedback loops or “network effects” between users and application developers. Applications attract users, which motivates developers to create more applications, which in turn attract more users... and so on.

True application platforms like iOS and Android attract huge financial investments. If a typical app costs an estimated $10,000 to $50,000 to develop, then the 500,000 iOS apps represent an average investment of $15B in iOS. Because of network effects, this investment on the part of developers, investors, and brands directly contributes to iOS’ value, in the eyes of both users and other developers, and Apple’s estimated $71B iOS-powered device sales for the year ending September 2011.

Developers therefore drive adoption of an application platform. A platform is only as successful as the developers who build its apps. The total number of apps is frequently considered as the ultimate measure of application platform success, but it is really the long-term health and sustainability of the developer ecosystem that will determine whether developer investment in a given platform grows, stagnates, or declines.
**App stores are the keys to ecosystem control**

App stores have become the main developer-to-consumer distribution channel in recent years. Over 45% of respondents in our Developer Economics 2011 survey reported using an app store as their primary route to the market, up nearly 30% from the previous year. Meanwhile, fewer reported using other distribution channels, such as third party aggregators, operator portals, existing customers, and developers' own portals/websites (previously the top route to market).

The reason for the sudden popularity of app stores is the reach they offer. The ability to sell to more users was cited as the primary appeal of app stores by more than half of all developers who reported using the Apple, Google, Nokia or BlackBerry app stores.

Another important motivation is the fast time-to-market that app stores facilitate. Findings from our Developer Economics study suggest that app stores have reduced the average time-to-shelf by two thirds: from 68 days across traditional channels, to 22 days via an app store.

Moreover, app stores have reduced the time-to-payment by more than half; from 82 days on average in the case of traditional channels, to 36 days on average with app stores.

Native app stores have a clear advantage. With millions of downloads each day, they provide a reach far beyond what developer-owned websites or even smaller, independent app stores can provide. They do this at a reasonable cost: typically 30% of the sales price, compared to the 50% or even 90% that were common 'sales commissions' for operator portals.

With most app retailing routed through native app stores, platform owners leverage app stores as ecosystem control points, rather than revenue sources. App stores allow platform owners to control their ecosystems in five important ways.

1. **Content selection and curation.** App store owners determine which apps are presented to users, and they can therefore favour apps that contribute to their own bottom line. For example, in July 2009, Apple rejected a Google Voice app, claiming it replicated the iPhone’s functionality. This prompted an intervention by the FCC, and Google released Google Voice as a web app in January 2010. Only in November 2010, was a native Google Voice app again available on the iPhone. In addition, Apple curates applications to enforce a minimum quality of user experience.

2. **Distribution of applications.** The Apple App Store is the exclusive distributor of iOS apps for iPhone, iPad and iPod Touch devices. Since third-party application stores are not allowed on iOS devices, Apple has full control over what applications and content will be distributed to iOS users. Google has a different approach. The company uses the Android Market to force Android handsets to comply with Google’s own business requirements: only handset devices that pass Google’s CDD/CTS certification can use the Android Market. Non-compliant devices do not have access to Android Market apps, which puts those handset vendors at significant competitive disadvantage.

3. **Billing and monetization of applications.** App stores provide an opportunity for platform owners to extract a commission, in the form of a revenue share for paid apps and in-app purchases. Historically, these revenues have not been considered profit centres; instead,
they offset the cost of app store operations, which in the case of Apple amounts to over $1 billion to date.

4. **Retailing and discovery of applications.** Likewise, app stores are a channel for controlling which apps are shown to which user. Moreover, app store inventory can be used to sell ads and targeted services to developers.

5. **Consumer insights.** Finally, app stores are an opportunity to collect data about user behaviour. Such data can be used later to optimize promotion of content towards specific user segments.

**An app store contender emerges**

With web technologies such as HTML5, it is becoming more feasible to write fully functional apps with access to the phone’s capabilities. Such apps run in a browser, the only de-facto installed runtime on all handsets that is not bound to a proprietary ecosystem.

HTML5 has the potential to become a common bridge system across smartphone platform islands and the sea of feature phones. Large players – from mobile operators to Facebook and Microsoft – are using HTML5 as a technology intended to reduce the power of native platforms and dis-intermediate native app stores as distribution silos.

For example, in October 2011, Facebook announced its Platform initiative for mobile via a developer blog post. Facebook Platform (codenamed “project Spartan”) is designed to drive the discovery and distribution of HTML5 apps on mobile handsets. This is a major improvement, as web apps were previously discoverable by search or links on external websites. Facebook Platform also supports the discovery of native mobile apps.

Facebook Platform sports social channels such as Bookmarks, News feeds and Requests (friend invitations) to drive the discovery and distribution of apps. Users can pay for apps using the same Facebook credits they use on the desktop.

When announcing Facebook Platform, Luke Shepard, an engineer on the Platform team, remarked that Platform represents merely the beginning of Facebook’s push into mobile apps. Facebook reports more than 350 million active mobile users and is using mobile web as an alternative development and distribution platform to break through the iOS and Android app store silos.

Overall, the web as a development platform has several distinct advantages:

- Web apps have the potential for tremendous reach, given the web’s cross-platform nature and the growing ubiquity of WebKit-based browsers on mobile devices.
- The web allows content providers and media brands to get on board from day one with their legacy web content, and at a sustainable, low cost. More importantly mobile web avoids the need for content relicensing for mobile handsets, which is often prohibitively costly for media brands.
- As a development platform, the mobile web can attract literally millions of web developers that currently do not write mobile software.
Mobile Platforms: The Clash of Ecosystems

- The mobile web ranks third, behind only Android and iOS, in terms of developer mindshare, according to our Developer Economics 2011 research. This kind of mindshare represents the equivalent of many billions of dollars of free developer marketing.
- The web is a kind of patent haven, since web technologies are available in the public domain or under open source licensing, and are not behind corporate legal walls.

The main challenge with the web as a platform is fragmentation of both development and distribution. Unlike mobile platforms, in the mobile web there is no clear leader to push forward a single, coherent web development platform with a sufficiently large installed base of devices and a single, compelling distribution channel or app store.

**Mounting developer acquisition costs**

Platforms need apps, and app developers fuel the growth of a smartphone ecosystem. At the same time, developer attention is scarce; developers are very critical “platform consumers” and need to make far higher investments when adopting a new platform. We estimate that the minimum acquisition cost for a publishing developer is over $2,300 in the case of Apple. As such, Apple, Google, Nokia, Microsoft and RIM have needed to invest billions of dollars in persuading developers to write apps for their platforms. For example, Microsoft is believed to have spent over $1 billion in marketing budgets for promoting Windows Phone during its first year of launch.

Moreover, no two developers are alike. The developer ecosystem is a varied landscape comprised of:

- independent software vendors (ISV)
- contractors
- hobbyist developers
- moonlighting software engineers
- entrepreneurs
- integrators and in-house developers
- B2C companies
- B2B companies

Some developers are motivated by potential revenues, some by the number of users that can be reached. Yet others are motivated by the pure coolness and utility of the platform. These incentives vary widely across different types of developers and as such call for developer segmentation as a critical cornerstone of any developer strategy.

Platforms fiercely compete for developer mindshare using a complex set of motivators. These include monetisation, user reach, quality of development tools, technology characteristics, and the coolness factor.

Using a clever mix of emotional and business incentives, Apple has turned iOS into “a must” platform for most mobile developers. Android is quickly becoming a viable alternative, for developers attracted
by its wide user reach and “openness.” Microsoft is buying its way into the minds of developers by co-sponsoring the porting of many popular apps to its Windows Phone platform.

The next table demonstrates the effectiveness of Google, Apple and Microsoft in amassing ecosystems of “publishing developers” and the associated developer mindshare.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Publishers⁴</th>
<th>Average apps/user⁵</th>
<th>Mindshare ⁶ (of developers using the platform)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>70,000</td>
<td>4.1</td>
<td>67%</td>
</tr>
<tr>
<td>iOS (iPhone)</td>
<td>101,000</td>
<td>4.1</td>
<td>59%</td>
</tr>
<tr>
<td>iOS (iPad)</td>
<td>36,000</td>
<td>3.6</td>
<td>N/A</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>7,000</td>
<td>3.6</td>
<td>36%</td>
</tr>
</tbody>
</table>

Software players put mobile operators on the defensive

The lower barriers to entry of modern smartphone platforms bring a wide spectrum of new players into the mobile ecosystem. Anyone can write an app and engage smartphone users. And anyone can bypass the mobile operator “toll-booths.”

Apple and Google combined control the user experience of nearly 400 million users through their iOS and Android platforms. Both are strategically reducing the role of mobile operators to that of “connectivity providers”. Internet giants like Facebook and Amazon are using social-centric and retail-centric strategies to profit from mobile. Start-ups such as Foursquare and Instagram have pioneered mobile-first services. Communication companies like Skype, WhatsApp and Viber put pressure on core telecom services, notably SMS and voice.

Mobile operators have yet to find a success formula for dealing with the disruptions coming from software players.

Two horses lead the race

The last decade saw 20+ mobile operating system contenders, from handset makers (MotoMagx, UIQ, Palm 5/6, OpenMoko), software companies (Intrinsyc Soleus, Sasken Aria, SKY-MAP, TTPCom AJAR, Azingo Mobile, Access ALP, Openwave MIDAS, Mizi Prizm) and even operators (SavaJe) rise and eventually die not having achieved critical mass. However, today smartphone competition is a two-horse race: the smartphone OS market leadership is virtually divided between Apple iOS and Google Android. The two leading platforms share three defining characteristics:

⁴ Xyologic, USA market, August 2011
⁵ Xyologic, USA market, August 2011
⁶ [VisionMobile Developer Economics 2011 report]
1. They originate from non-telecom players, with computing and Internet platform DNA.

2. They are monetised indirectly. Instead of software license sales, they platform vendors profit by selling high-margin devices, in the case of Apple, or enhancing Google’s on-line advertising business, in the case of Android.

2. They are driven by companies able to sustain investments of billions of dollars in order to develop an OS, create engaging online services and drive an application ecosystem connecting users and developers.

Both Android and iOS are controlled by single companies. They have come to dominate over the many telecoms industry attempts at creating a standardised and commoditized platform such as LiMo, JCP, OMTP, Symbian Foundation, MeeGo and recently Tizen. The proponents of these initiatives have all failed to fully comprehend the economics of application-led platforms, and to move fast enough (at software speeds) to compete with Internet-borne players.

There isn’t and won’t be a single winner in the smartphone race. Both iOS and Android platforms reached a critical mass of hundreds of millions of users. Their long-term positions are secured by strong network effects between users and developers, making it almost impossible to displace them.

The jury is still out on whether there is a place for a third platform player, and whether Microsoft, with the help of Nokia, will be able to wrestle the third position away from Blackberry.

The rising star of HTML5

With Apple and Google gaining power, the industry is looking for alternatives. For many, the emerging HTML5 standard holds the promise to break the app distribution silos of iOS and Android platforms.

HTML5 is a set of draft standards that, along with the recently launched CCS3 and ever-faster, more efficient JavaScript engines, has the potential to greatly increase the functionality of web apps, while at the same time requiring fewer lines of code. A subset of the emerging HTML5 standard is already supported by many browsers, including the WebKit-based browsers distributed with Apple iOS and Google Android platforms.

Mobile operators hope that as web apps become more powerful, they will replace “native” applications and reinstate operator portals as the distribution channels ahead of native app stores. Facebook is eyeing HTML as a technology that can help dis-intermediate app stores as content distribution silos, reducing the power of Apple’s iOS and Google’s Android platforms. App developers, meanwhile, look at HTML5 as a technology that will bridge incompatible software platforms, reduce the effort needed to go cross platform.

In reality, however, HTML5 is still far from becoming a replacement for native applications. Multiple technical gaps still exist, such as incompatibilities between different web browsers, poor performance on mobile devices and a lack of 3D graphics. More importantly, HTML5 lacks the properties of a complete application platform. The HTML5 specification merely defines programming interfaces for web run-times (i.e., “browsers”). HTML5 does not address key areas an application platform would need to challenge iOS and Android, namely application discovery, distribution, monetisation and retailing.
Chapter B
Mobile platform shootout

For details on the platform versions being reviewed, see the appendix at the end of this chapter.

History and Origins

There are two generations of mobile platforms: “legacy” platforms and “next-generation” platforms. Legacy platforms originate from products built to the requirements of the 2000-2009 decade. These include Symbian, BlackBerry OS, BREW and Windows Mobile.

Next-generation platforms are heavily influenced by the groundbreaking iOS platform, and are designed ground-up for the requirements of the Internet age and developer economics. These include Android, webOS, Windows Phone and Bada.

None of the legacy platforms have managed to compete successfully:

- Symbian is being phased out by Nokia following an unsuccessful open-source experiment.
- RIM plans to replace BlackBerry OS with the QNX operating system.
- BREW is failing to achieve traction with either developers or handset makers, beyond a small number of device models aimed at developing markets.
Mobile Platforms: The Clash of Ecosystems

- Microsoft has stopped Windows Mobile development, and moved on to new Windows Phone platform, which is built on new software foundations.

US companies completely dominate the mobile platform landscape. All major smartphone platforms were designed in the US, which has now become the hotbed of mobile software innovation. However, none of the next-generation platforms was produced by telecom or mobile companies:

- Apple has a rich legacy in personal computers, consumer electronics and digital content.
- Google is the leading Internet player.
- Microsoft is the dominant PC software maker.
- Palm (acquired by HP) was a long-time maker of personal digital assistants (PDAs).

The next table summarises the history and origins of eight mobile platforms.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Platform owner</th>
<th>Company origins</th>
<th>OS origins</th>
<th>Initial product launch</th>
<th>Geographic origins</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbian</td>
<td>Nokia</td>
<td>Mobile phones</td>
<td>PDA</td>
<td>2000</td>
<td>Europe</td>
<td>Phasing out</td>
</tr>
<tr>
<td>BlackBerry OS</td>
<td>RIM</td>
<td>Wireless messaging</td>
<td>Two-way pagers</td>
<td>2000 (Java)</td>
<td>Canada</td>
<td>Will be replaced by QNX</td>
</tr>
<tr>
<td>BREW</td>
<td>Qualcomm</td>
<td>Wireless chipsets</td>
<td>Feature phones</td>
<td>2001</td>
<td>US</td>
<td>Weakening traction</td>
</tr>
<tr>
<td>Windows Mobile</td>
<td>Microsoft</td>
<td>PC software</td>
<td>PDA</td>
<td>2002</td>
<td>US</td>
<td>Phasing out</td>
</tr>
<tr>
<td>iOS</td>
<td>Apple</td>
<td>Personal computers</td>
<td>Personal computers</td>
<td>2007</td>
<td>US</td>
<td>Leads in tablets</td>
</tr>
<tr>
<td>Android</td>
<td>Google</td>
<td>On-line advertising</td>
<td>Acquisition of smartphone OS startup</td>
<td>2008</td>
<td>US</td>
<td>Leads in smartphone sales</td>
</tr>
<tr>
<td>webOS</td>
<td>HP</td>
<td>Consumer electronics</td>
<td>New development</td>
<td>2009</td>
<td>US</td>
<td>Orphaned after HP cancelled device plans</td>
</tr>
<tr>
<td>Bada</td>
<td>Samsung</td>
<td>Mobile phones</td>
<td>Evolution of proprietary feature-phone OS</td>
<td>2010</td>
<td>South Korea</td>
<td>Second choice for touch-screen phones</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>Microsoft</td>
<td>PC software</td>
<td>New development</td>
<td>2010</td>
<td>US</td>
<td>Unproven challenger in high-end smartphones</td>
</tr>
</tbody>
</table>

Source: VisionMobile
Market adoption

Despite the sharp rise in smartphone shipments in 2010-2011, the rate of smartphone adoption is uneven across different geographies.

Mobile markets with extensive 3G coverage and a large share of “post-paid” subscription plans show the fastest smartphone adoption rates. Mobile operators in such markets tend to subsidize smartphones heavily, in return for users committing to more expensive, longer-term subscription plans.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Cumulative sales since launch until Q2 2011</th>
<th>Low-end touch phones</th>
<th>Mid-tier smartphones</th>
<th>High-end smartphones</th>
<th>Tablets</th>
<th>Market status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>165 million</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Strong presence and growth in the US, Western Europe and other post-paid markets. Growing sales of low-cost models in pre-paid markets</td>
</tr>
<tr>
<td>Bada</td>
<td>7 million</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>Considerable growth across some geographies</td>
</tr>
<tr>
<td>BlackBerry OS</td>
<td>163 million</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>Losing market share in the US, growing in consumer pre-paid segment</td>
</tr>
<tr>
<td>BREW</td>
<td>668 million (*)</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Limited to low-cost devices in developing markets</td>
</tr>
<tr>
<td>iOS</td>
<td>222 million</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>Strong presence and growth in the US, Western Europe and other post-paid markets</td>
</tr>
<tr>
<td>Symbian</td>
<td>486 million</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>Losing ground to iPhone and Android in post-paid markets</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>4.6 million</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>Limited market acceptance mainly in the US and Europe</td>
</tr>
<tr>
<td>webOS</td>
<td>2.5 million</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>Negligible market share in the US and Western Europe</td>
</tr>
</tbody>
</table>

(*) Total sales of BREW-capable handsets. VisionMobile estimate, up until end of 2010
The battlefront, for most smartphone handset makers, is price. However, in heavily post-paid markets, premium iPhone and high-end Android models from Samsung, HTC, Motorola and Sony Ericsson are actually displacing less sophisticated smartphones from Nokia and BlackBerry.

The picture is different in mobile markets dominated by pre-paid subscriptions. In those markets, there are typically no operator subsidies for mobile handsets. As a result, these markets are much more price sensitive.

Nokia’s mid-tier Symbian phones and BlackBerry consumer models are holding onto significant shares of the smartphone market in pre-paid markets. However, both Nokia and BlackBerry are facing growing competition from inexpensive Android models manufactured by low-cost handset makers.

Compared to other mobile platforms, Android has greater price flexibility. Available Android devices sell for $100 to $750, with stops at all price points in between. As a result, Android competes with high-priced iPhones in post-paid markets, and with lower-cost Nokia and BlackBerry devices in pre-paid markets. Such pricing versatility has contributed to Android’s rapid growth.
Even though Symbian still has a large installed base, and a significant market share, Nokia’s February 2011 announcement about phasing out the platform has put an end to Symbian growth. Next to Symbian, iOS and Android have the largest installed base of customers, and are therefore the most attractive platforms for application developers and content providers.

The islands of application development

As we discussed earlier, applications play a pivotal role in the success of a mobile platform. It is critically important that a platform provides developers with efficient tools for all stages of application development, from writing the code, to publishing the app. Different platforms have varying degrees of success in this area, and their success is reflected in the health of their app ecosystems. Moreover, developers need to master new tools every time they switch to a new platform.

There is no common programming language that can be used on all platforms:

- iOS apps are written using Objective-C (a superset of C influenced by Smalltalk)
- Android and BlackBerry use different dialects of Java as their main programming languages
- Windows Phone apps are developed using C#, which is part of .NET framework
- Symbian^3, Bada and BREW support commonly-used C and C++ languages
- webOS uses JavaScript together with HTML/CSS as its primary development language.

There is no common development environment either:

- iOS apps can only be developed using Apple XCode tools running on Mac OS computers
- Windows Phone apps have to be developed with Microsoft Visual Studio tools using a Windows PC; Windows is the de facto platform for enterprise software developers.
- Symbian^3 development uses the Qt environment.
- WebOS apps are developed using the Ares browser-based development environment that is used in combination with Eclipse IDE.

The remaining platforms have settled on either the Eclipse developer environment, along with proprietary plug-ins and device emulators, or else Visual Studio.

The easier it is to begin programming for the platform, the more developers will start experimenting with that platform and eventually write and publish apps for it. Apple excelled in lowering barriers to entry for developing apps for iOS (except for the need to buy a Mac, of course). The iOS SDK sets the benchmark in how easy it is to install and configure the tools and develop a first app. It is not uncommon for people with little or no computer science training to develop apps for iOS.

Android, Windows Phone and Symbian (Qt) also offer competitive development tools, but they typically require more experience, or for new developers, more learning effort.

Rich APIs save developers lots of time, by helping them write less code. The less code developers have to write, the less code they have to debug and maintain.
Among mobile development platforms, iOS and Android offer developers the most feature-rich and flexible API frameworks. The iOS application framework is derived from the proven Mac OS API, and excels in UI-related areas. Compared to Android, however, it is less flexible, and often restricting.

The rest of the platforms still have API gaps. Windows Phone is maturing, with each platform upgrade bringing support for essential but previously missing APIs. The Symbian Qt application framework is a combination of proven Qt user interface APIs and the newly developed Qt Mobility API. The latter is still evolving, adding access to more platform features.

The majority of developers use native platform application stores for distributing their apps. Each store has its own guidelines, policies and procedures. The Apple App Store is known for its rigorous certification requirements, and unpredictable approval process. Applications can be rejected for unexpected reasons, and there is no way to predict how long it will take to pass the certification. For developers designing apps for non-US or non-English markets, there is another hurdle: Apple testers sometimes misunderstand local aspects of the app or service, and reject it for the wrong reasons.

The process for publishing apps on Android Market is exactly the opposite. Android application publishing is a self-service process foregoing any application testing or manual approval. On one hand, this makes life easier for developers: the submitted app usually shows up in the store within minutes. On the other hand, this results in large numbers of poor quality, copycat, copyright-infringing or even malware applications available in Android Market, leading to degraded user experiences for Android users.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Main language</th>
<th>Dev computer</th>
<th>Dev tools</th>
<th>Ease of entry</th>
<th>API richness</th>
<th>Certification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>Java</td>
<td>Windows</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mac OS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Linux</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bada</td>
<td>C/C++</td>
<td>Windows</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>+</td>
</tr>
<tr>
<td>BlackBerry OS</td>
<td>Java (J2ME)</td>
<td>Windows</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>BREW</td>
<td>C/C++</td>
<td>Windows</td>
<td>==</td>
<td>==</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iOS</td>
<td>Objective-C</td>
<td>Mac OS</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>-</td>
</tr>
<tr>
<td>Symbian^3</td>
<td>C/C++</td>
<td>Windows</td>
<td>+</td>
<td>=</td>
<td>=</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mac OS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Linux</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Windows Phone</td>
<td>C#</td>
<td>Windows</td>
<td>+</td>
<td>+</td>
<td>=</td>
<td>+</td>
</tr>
<tr>
<td>webOS</td>
<td>Javascript</td>
<td>Windows</td>
<td>=</td>
<td>+</td>
<td>=</td>
<td>+</td>
</tr>
</tbody>
</table>

Source: VisionMobile
The rest of the platforms attempt to strike a balance between openness and quality by combining human curation and moderate testing requirements with fair, predictable approval policies.

**Understanding platform economics**

It is important to distinguish between three very different types of mobile platforms: software platforms, application platforms and communication platforms.

<table>
<thead>
<tr>
<th>Platform type</th>
<th>Purpose</th>
<th>Primary audience</th>
<th>Network effects</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software platform</td>
<td>Sharing of software development costs and risks</td>
<td>Device makers</td>
<td>None</td>
<td>Symbian, BREW</td>
</tr>
<tr>
<td>Application platform</td>
<td>Connecting app developers and users (and handset OEM in some cases)</td>
<td>Developers</td>
<td>- Users to developers</td>
<td>Android, iOS, Windows Phone</td>
</tr>
<tr>
<td>Communication platform</td>
<td>Facilitating communication between users</td>
<td>Users</td>
<td>- Users to users</td>
<td>Telephone, fax, BlackBerry Messenger</td>
</tr>
</tbody>
</table>

Source: VisionMobile

**Software platforms** are used for building multiple variations of products, with maximum reuse of the platform technology. As such, software platforms are optimised for flexibility and sharing of development costs across multiple products. Symbian is a typical example of a software platform that has been used for building a wide range of mobile phone models. Until 2009, developers were not the focus of the platform.

In contrast, **application platforms** are designed for connecting two disjointed markets: users and application developers. Applications provide solutions to a diverse set of user needs. Since applications are locked to the platform, users must acquire the platform in order to benefit from applications. Microsoft Windows is a classic example of a successful application platform. PCs are useless without applications. Since most PC applications are developed for the Windows operating system, it is necessary to buy a license for Windows to use these PC applications.
Application platforms are a specific case of so-called two-sided markets. Two-sided markets enable interaction of two disjoint groups of participants. Some other examples of two-sided markets include credit cards, stock exchanges and digital media formats.

Successful application platforms have very strong **network effects**. Applications attract users, which attract developers to create more applications, which attract more users, which attract more developers, and so forth. Each new application adds value to the platform from the end-user perspective. And every new user adds value to the platform from the application developer perspective.

The next graph shows the relationship between the number of apps available on particular platform at a particular point in time, and the number of devices shipped for that platform in the quarter just preceding it. The number of apps available can be considered a metric for how attractive a platform is for developers. Device shipments are likewise a measure of the attractiveness of a platform for its users. The tight correlation between platform attractiveness to users and developers is consistent with Android and iOS being successful application platforms.
The positive feedback loop between developers and users dominates other aspects that might affect sales or app development, like promotions and advertisements, or the coolness of a particular technology for developers.

Consistency is key for the success of an application platform; applications must run on all implementations of the platform. For example, imagine if Microsoft Office only ran on Dell computers.

A software platform, in contrast, needs flexibility rather than consistency, allowing it to fit the requirements of a wide range of product variants, and reach supply-side economies of scale.

It is clear that Apple’s iOS is a clever implementation of a mobile application platform coming from a company with the right kind of experience and DNA. Strong network effects between users and app developers are inherent to iOS ecosystem design. These network effects have proven to be stronger than Nokia’s power over its supply chain and distribution channels, and have eventually led to the demise of Symbian.

The unprecedented success of Apple iOS is a testament to the advantages of application platforms. All new platforms introduced after the original iPhone – including Android – attempt to copy Apple’s application platform recipe.

Android comes from a company that has plenty of experience with advertising platforms that connect on-line users and advertisers. It’s hardly a surprise that Android was designed as a free application platform that makes money by driving traffic to Google’s on-line advertising services.
When plotting Blackberry, Symbian and Windows Phone on a graph similar to the network effects graph for iOS and Android, we can see the mixed results of the efforts of the platform owners to establish pronounced network effects.

Nokia has driven users away from Symbian by positioning Windows Phone as the unambiguous future for its phones, precluding any chance to kick-start a network effect.

Microsoft, on the other hand, is clearly trying to jump-start the network effects of an application platform. They have incentivised developers to produce a large amount of apps, but users so far remained unimpressed with the early version of the platform.

Application platforms are Microsoft’s bread and butter. No wonder, then, that Windows Phone has all the characteristics of an application platform, from high consistency to a strong focus on app developers.

The jury is still out, however, on whether Microsoft can turn Windows Phone into a credible competitor to iOS and Android. For that to happen, Windows Phone must reach a significant market share: the critical mass at which strong network effects begin to take hold between users and application developers.

As an application platform grows, network effects accelerate. That, combined with vendor lock-in (making it inconvenient for users to switch) makes it very difficult to displace the leading application.
Mobile Platforms: The Clash of Ecosystems

Platforms, once they are established. Merely creating a better OS is not enough, as proven, arguably, by contrasting the histories of Mac OS and webOS.

Because of its late start, Windows Phone will need to offer something radically different. Only the future will tell if Microsoft will be able to achieve that. Currently, it is working to leverage its recent acquisition of Skype, its partnership with Nokia, and its expanding relationship with Facebook.

What about BlackBerry? In platform terms, BlackBerry is a successful communication platform, connecting users though mobile email or popular BlackBerry Messenger (BBM) services. Communication platforms also have network effects, with every new user making the network more valuable for other users. A traditional telephone network is another example of a communication platform that connects users.

Because of a narrow focus on communication needs, however, the primary network effects in the BlackBerry platform are between platform users, not users and developers. These user-to-user network effects are apparently weaker than the developer-to-user network effects of iOS and Android. Wide availability of applications in iOS and Android lets these platforms serve a broad spectrum of user needs, creating value beyond what Blackberry has achieved so far. RIM is struggling to evolve BlackBerry into an application platform, but thus far has been impeded by platform fragmentation, among other issues.

Blackberry’s initial communication network effects are no longer driving sales, and while they have made progress in adding apps, they have not really created a strong new feedback loop.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Number of apps</th>
<th>Originally designed as</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>300,000</td>
<td>Application platform</td>
</tr>
<tr>
<td>Bada</td>
<td>15,000 (*)</td>
<td>Application platform</td>
</tr>
<tr>
<td>BlackBerry OS</td>
<td>35,000</td>
<td>Communication platform</td>
</tr>
<tr>
<td>BREW</td>
<td>N/A</td>
<td>Software platform</td>
</tr>
<tr>
<td>iOS</td>
<td>500,000</td>
<td>Application platform</td>
</tr>
<tr>
<td>Symbian</td>
<td>25,000</td>
<td>Software platform</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>30,000</td>
<td>Application platform</td>
</tr>
<tr>
<td>webOS</td>
<td>7,000 (*)</td>
<td>Application platform</td>
</tr>
</tbody>
</table>

Source: VisionMobile

Developers as major driver for platform adoption

Complex application ecosystems have developed around the leading mobile platforms. These ecosystems comprise a diverse set of companies and individuals. Each type of developer in these ecosystems is driven by different motivations. These motivations can be grouped into four broad categories:
Recognition and fun, typical for hobbyist developers making free apps

Software revenues, typical for independent software vendors (ISV) and software publishers looking to monetise apps by paid downloads, in-app advertising or in-app purchases

User reach, typical for players like venture-backed startups, on-line service providers, content retailers, media companies and B2C marketers

Cost of ownership, typical for enterprises deploying business solutions for their workforce

All platforms offer hobbyist developers opportunities for self-expression. Ease of entry plays a critical role. The easier it is to make the first application, more hobbyist developers will be attracted to the platform, and more free apps will be available for the platform.

iOS would be the best platform for hobbyists, if not for the requirement to use a Mac computer for app development. Mac computers are expensive, and out of reach for many people. Android tools, in contrast, can be used on Windows, Linux and Mac OS computers.

iOS offers the best opportunities for developers interested in software revenues. This is largely due to the seamless purchasing process, and the fact that Apple has stored credit card information for the majority of iOS users. In one of his famous keynote speeches, Steve Jobs stressed that Apple holds more than 200 million credit cards on file. Google lacks this important asset and, as a result, Android users are much less likely to pay for apps.

For some developers, mainly venture-backed startups, user reach is more important than direct revenues. iOS again leads in terms of user reach, running on more than 220 million iPod, iPhone and iOS devices. Android is quickly closing the gap, growing to over 190 million devices in just three years.

Traditionally, the enterprise segment has been RIM’s forte. The Canadian phone manufacturer first met with success selling to corporate accounts. iOS is growing in popularity in the enterprise, due in large part to the success of the iPad. Moreover, since version 3.0 of the platform, iOS supports most of the device management and policies features necessary for large-scale enterprise deployments.

Compared to iOS and BlackBerry, Android has higher costs of ownership for corporations. Factors here include inconsistent device management support across different models, general device fragmentation, and the extreme openness of the Android Market app store.

Our Developer Economics survey (www.DeveloperEconomics.com) showed that developer mindshare is moving away from “old guard” platforms like Symbian and Java, in favour of iOS and Android. This trend emerged in 2010, and grew in 2011. The reach that these high-momentum platforms provide seems to be the determining factor, overshadowing inconveniences like fragmentation or increased competition in the app stores.

An interesting third contender is mobile web, a low-barrier, cross-platform, mass-reach platform. With HTML5, the mobile web is advancing, technologically. Also significant: HTML6 has the potential to tap a massive existing developer pool (web developers) not previously active in mobile.

**Conclusions and Outlook**

The mobile market will continue in the coming years as a multi-horse race. iOS and Android will continue to lead, dividing the market between premium and mass-market product types. Self-
reinforcing network effects, huge application ecosystems and the fast pace of evolution make the positions of Apple and Google unassailable.

Symbian is losing its relevance quicker than Nokia hoped for, especially in the critically important Western European market. The premature announcement of the platform’s phase-out caused operators, developers and partners to withdraw much-needed support.

Windows Phone, together with Nokia, will be challenging BlackBerry for the third place. Unless BlackBerry can evolve into a fully-fledged application platform, the Windows Phone platform has the better chance to rank third, after iOS and Android. The distance between Windows Phone and the leaders will depend on Nokia’s success with its Windows Phone smartphones, and on Microsoft’s ability to innovate based on its Skype acquisition and Facebook relationship.

The lack of a single, clear winner among mobile platforms means developers will need to continue developing on multiple platforms. That entails mastering diverse and incompatible tools, APIs, programming conventions and application store policies.

HTML5 has the potential to enable shared development costs across platforms. However, in its present state, HTML5 is a software development technology, and not an application platform. It is important to know that as a result, HTML5 can neither challenge nor replace the mobile platforms themselves.

The mobile platform landscape and associated application ecosystems are expected to become ever more complex in the coming years. New companies will enter and compete in the mobile device space, as a result of smartphone and tablet hardware commoditization, and the relative openness of the Android app market. These players will use branded tablets and smartphones to drive their non-mobile businesses.

This phenomenon will be especially visible in tablets. Apple’s iPad so far has dominated the tablet market, propelled by availability of attractive content and apps in iTunes and App Store.

Out of many iPad challengers, Amazon is the most interesting one. The company recently released its own branded tablet, the Kindle Fire, which is squarely positioned as a media consumption and Internet browsing device, connecting the user to millions of books, songs, videos and TV shows.

In both cases, the combination of device and content is crucial for marketplace differentiation, and for fighting the commoditization that has plagued other tablet vendors. The difference between the iPad and the Kindle Fire is the selected use case and the business model:

- For Apple, the content offering drives device sales and profit margins.
- Amazon uses the device as a loss leader to drive its core content retail business.

Apple and Google turned the mobile industry on its head by creating vibrant product ecosystems — encompassing devices, content and on-line services. The battle of ecosystems, however, is still far from being decided. We expect continued evolution of this dynamic market driven by entry of new players, such as Amazon, Facebook and Microsoft, as well as expansion of the experience ecosystems across screens, shifting the battleground from smartphones to tablets and finally the living room.
Appendix: Platform versions reviewed in this chapter

This chapter presented a comparative analysis of the following mobile platforms:

1. Android: We refer to versions 2.3 (smartphones) and 3 (tablets) of the platform.
2. Bada: We refer to the first generation of the Bada OS.
3. BlackBerry OS: This is the version found on RIM smartphone models. Since the next-generation OS from RIM can only be found on a single tablet model, it is not discussed in this report.
4. BREW: We refer to the newer version of the platform used in touch-screen phones.
5. iOS: We refer to version 4 of the platform.
6. Symbian: Except for the discussion on market shares, we refer to the Symbian^3 version found in the most recent Nokia device models.
7. Windows Phone: We refer to Windows Phone 7.
8. webOS: Since the products designed by Palm prior to its acquisition have limited market presence, we refer to version 3 of the platform developed by HP following the acquisition of Palm.
Chapter C.
Platform drill-down: from Android to webOS

Android

The operating system originates in Google’s acquisition of Android, Inc. in 2005. The start-up company was founded by ex-Danger and T-Mobile executives that were behind development of the SideKick, a mobile device that gained popularity with teens and some celebrities.

In November 2007, Google and its partners announced the formation of the Open Handset Alliance (OHA), a PR-led effort that signalled an industry endorsement for the development of the open source mobile operating system called “Android.”

Google exerts tight control over platform development, including roadmap planning and code commit decisions. Only Google employees are allowed to commit code to the main software branch. The company also uses what we call an “open-late” licensing practice, where newer versions of the OS are only shared with a small number of carefully selected handset vendors. These versions of the OS become publicly available only after Google and its partners have begun to distribute products based on them.

The pace of Android development sets a new standard in the telecom industry, with four to five version releases each year. It stands out even in comparison with iOS, which evolves at a more measured pace. Android’s development pace has been more characteristic of Internet software development than traditional telecom developments.

The success of Android in winning the largest share of smartphone sales prompted Apple and Microsoft to leverage their rich patent portfolio in attempting to slow down Android growth. Apple initiated legal actions in the US, Western Europe and Australia against the leading Android OEM, Samsung, attempting to halt sales of its devices. Microsoft resorted to a different tactics – patent taxes – and has succeeded in extracting royalties from OEMs such as Samsung and HTC.

Google’s intellectual property rights position in Android has been weak from the onset. It attempted a fix in July 2011, when it famously bid $3.14159 (pi) Billion for a portfolio of Nortel patents that had been put out for auction. However, it lost out to a $4.5B bid from a group of adversaries, comprising Apple, Microsoft, RIM, EMC, Ericsson and Sony.

<table>
<thead>
<tr>
<th>Owner</th>
<th>Google</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launched</td>
<td>November 5, 2007</td>
</tr>
<tr>
<td>Devices sold since launch</td>
<td>225 million</td>
</tr>
<tr>
<td>Apps downloaded</td>
<td>6 billion</td>
</tr>
<tr>
<td>Apps available</td>
<td>300 thousand</td>
</tr>
<tr>
<td>Device price range</td>
<td>$99 to $754 USD (handsets) $89 to $1059 USD (tablets)</td>
</tr>
</tbody>
</table>

VisionMobile estimates, as of Q3, 2011. Device prices refer to unlocked models, pre-tax.
In August 2011, Google announced the planned acquisition of Motorola Mobility, in a deal that included its hardware business and portfolio of over 17,000 patents.

Google’s pending acquisition of Motorola creates a dilemma: Google must choose between staying true to its core business, or reshaping into the new vertical giant that will challenge Apple at its own game.

We see two possible scenarios of how Google may solve the dilemma:

1. Google stays true to its core ad business and divests most Motorola assets apart from patents to an Asian OEM wanting to break into the North America market.

2. Google keeps Motorola’s hardware design and device production capabilities, to allow Android to compete head-to-head on every Apple price point, both the current high-end and the rumoured mid-tier pricing. To compete favourably with Apple, however, Google would have to take a hit on its cash flows and profitability, by putting its hand deep in its pocket (both in terms of CAPEX and OPEX).

It is worth noting that Google is the biggest Internet company, with quarterly revenues surpassing US $9.7B, a net quarterly income of $2.7B and cash reserves of over $42B, as of Q3, 2011.

**Positioning**

Android is positioned by Google as an open Internet platform, allowing unrestricted mobile access to Internet cloud services. Naturally, Google’s own cloud services play an important role in the platform offering. Google services like Search, Gmail, Maps, Android Market and YouTube are pre-integrated, and as a result are prominently featured in most products running Android. Activation of the phone usually involves entering a Google Account ID, which links the device to a wide spectrum of Google services personalised for the user. Examples include:

- Android Market (the Android app store)
- Gmail, Calendar and Contacts
- Search enhanced by user’s location information
- YouTube
- Google Maps, with turn-by-turn navigation and street view in some countries
- Google +
- Google Voice (US only)
- Photos (Picasa)

The initial versions of Android targeted smartphones, including high-end, mid-range and low-end devices. Android began expanding rapidly into tablets, connected TVs, and set-top boxes in early 2011. Android’s open source nature has enabled numerous vendors to work on parallel initiatives independently from Google. These initiatives have aimed to adapt Android for use in car entertainment systems, DECT handsets, video conferencing terminals, embedded devices and other consumer electronic products requiring a rich, programmable user interface and Internet connectivity.
It is important to note that many products produced from these independent initiatives will eventually be very different from the platform profiles supported by Google. As a result, these products will not be eligible for running official Android Market and other Google proprietary applications (Gmail, YouTube, Maps, etc.).

Android is extremely versatile. It scales across a range of (mostly ARM-based) application processors, including ARM11 528MHz at the low end (HTC Magic), ARM Cortex-A8 1GHz at the mid-range (HTC Desire, Nexus One), and dual-core ARM Cortex 1.2 GHz (Samsung Galaxy Nexus) at the high end.

Current prices span from the unlocked Huawei U8100, which sells for around $100 pre-tax, to Samsung’s Galaxy S II with NFC and 16GB (unlocked), which sold for $755 pre-tax as of October, 2011.

Android bears a royalty-free license, so sales of Android handsets do not bring direct revenue to Google. Instead, Android is monetized indirectly, through search-driven ads served to Android devices. The Android Market application store is pre-installed on Android devices and offers a wide selection of free and paid applications. Google takes a 30% share of revenues from application sales. This revenue is not considered to be an important part of Google’s revenue mix, due to its small size compared to Google's core business.

**Handset vendor adoption**

Google has successfully created a broad and diverse ecosystem of Android handset and tablet makers. In May, 2011, Google announced at the Google IO event there were 310 Android device models from 36 manufacturers in 112 countries.

HTC was the first handset maker to build Android smartphones. HTC made its name as a manufacturer of Windows Mobile smartphones growing into one of the top smartphone OEM. HTC remains committed to Android, and makes a full range of Android smartphones and tablets.

Except for Nokia, most top handset makers offer Android smartphones and tablets:

- Motorola put Android in the centre of its recovery plans, abandoning a home-grown Linux stack and several in-house phone RTOSe to focus solely on Android-based smartphones. Motorola produces a full range of Android devices, from mid-range models (Citrus) to high-end smartphones (Droid X and Droid RAZR), as well as tablet computers (Xoom). Note that Google announced plans to acquire Motorola Mobility in August 2011.

- Samsung produces a number of mid-range and high-end Android models, including its highly successful Galaxy product line, which includes Galaxy S smartphones and Galaxy Tab tablets.

- Sony-Ericsson entered the market of Android phones with X10 and X10 mini devices, which found success in the Japanese and European markets.

- LG offers a number of mid-range models, from Etna to Ally.

In addition, many low-cost manufacturers and ODMs (“original design/manufacturers,”) many from the PC market, have announced or are already selling Android smartphones and tablets. These include...
Dell, Huawei, Asus, Acer, ZTE, Kyocera, and Lenovo. Finally, many small Asian ODMs and OEMs (original equipment manufacturers) make white-label Android devices available for operator branding.

As of October 2011, Android was installed on an estimated base of around 190M handsets, according to Google. The sales base is growing rapidly, with 0.6M handsets sold in 2008, 6.8M in 2009, 70M in 2010 and 88M in the first half of 2011 alone.

Android found its initial success in the US market. Its high rate of adoption there was influenced by strong support from top US carrier Verizon Wireless, as well as the wide proliferation of Internet services and broad deployment of 3G data services in the US. Android gained popularity in Western Europe, and is showing strong growth in Asia and Latin America.

Android’s success in winning over handset makers is rooted in the desire of OEMs to compete with lucrative iPhone smartphones, and the support by leading chipset vendors such as Qualcomm, Texas Instruments and NVIDIA.

OEMs initially saw a high differentiation potential in the platform, due to the availability of Android source code. HTC pioneered customization (skinning) of the Android user interface; compared to the stock Android user interface, its “Sense” UI offers more aesthetic graphics, animations and widgets, among other “experience” improvements. Motorola and Sony Ericsson followed suit with their respective MOTOBLUR and Timescape customizations.

However, few OEMs can keep up with the pace of innovation set by Google with its “Google Experience” handsets. The Google Experience handsets are optimized for use with Google mobile services, and are designed by Google in partnerships with selected OEMs. Examples include the Motorola Droid (Milestone), HTC Nexus One and Samsung Galaxy Nexus. As a result, key OEM customisation efforts such Motorola MOTOBLUR and Sony Ericsson Timescape are being phased out.

**Operator adoption**

Android handsets are offered in most markets where operators deploy 3G networks, including North America, Latin America, Western Europe, Japan, Eastern Europe, Russia, and China. In many cases, Android is positioned as the alternative to popular iPhone smartphones.

Due to deep conflicts of interest between Apple and mobile operators, operators initially grew very enthusiastic about Android and the possibilities offered by the platform in the context of their “competition” with Apple. The differentiation potential of Android was viewed as an important aspect of operator’s ability to attract new subscribers and reduce churn to competition.

The Android platform has been strongly embraced by OEMs, many of whom are looking to diversify beyond heavily commoditized PC and laptop markets. Android device availability from multiple manufacturers, and at multiple price points, places operators in a convenient negotiating position when procuring handsets. Examples of first-tier operators strategically supporting Android are Verizon Wireless and T-Mobile in the US, Orange in France and DoCoMo in Japan.

China Mobile took a rather radical approach in leveraging the Android platform. China Mobile forked Android and created the new Open Mobile System (OMS) platform, promoted under the oPhone brand. OMS is developed in partnership with large Chinese software integrator Borqs.
The OMS platform was stripped of all Google services, which were then replaced by China Mobile proprietary solutions for messaging, location-based and content distribution services. The platform has improved support for the Chinese language, iPhone-like home screen, browser-based run-time (BAE) and support for a Chinese-developed mobile broadband technology – TD-SCDMA. The initiative is actively supported by Marvel Semiconductor, which is a primary supplier of chipsets and smartphone reference designs for OMS.

**Developer adoption**

Android Market lists over 300,000 applications as of October, 2011, making it the second largest platform application store behind Apple. The feature-rich and easy-to-learn application framework and powerful SDK, combined with growing number of handsets, have driven a mass arrival of developers into Android from both the PC space and the mobile space (Symbian, Java Platform Micro Edition [J2ME], Windows Mobile and PalmOS).

Android is the mindshare leader according to our VisionMobile Developer Economics 2011 research. Based on the research, two thirds (67%) of mobile developers have recently developed on Android, irrespective of the platform they are currently developing for.

There are concerns about platform fragmentation along five dimensions:

1. **Release speed.** With three major version releases between Q2 2010 and Q2 2011, the platform is changing too fast for OEMs.
2. **Dis-incentives for OEMs to update.** Android phone OEMs lack commercial incentives to keep updating handsets that have already been sold.
3. **OEM-specific UI fragmentation.** HTC’s Sense UI differs from Sony Ericsson’s Rachel, Motorola’s MotoBLUR, Samsung’s TouchWiz and LG’s S-Class user interface. All OEM additions create traces of fragmentation for developers.
4. **Screen fragmentation.** Android v2 and Android v3 address different ‘screens’ and are optimised differently for each. Screen types might include smartphones (different sizes), tablets, in-vehicle screens and TVs.
5. **Codebase forking.** China Mobile’s oPhone, China Unicom’s Wophone, Cyanogen and MiuiAndroid are ‘forked’ versions that no longer follow the same update trajectory as the main Android code base.

Android Market is the primary distribution and monetization channel for the absolute majority of developers. There are, however, a few independent application stores carrying paid content, which do not require Google Checkout. These include Amazon Appstore, SlideME and GetJar. Additionally, operators such as Vodafone, France Telecom/Orange and AT&T are launching Android app stores of their own, with application pre-qualification, and support for operator billing.

There are many concerns around the true monetization potential of Android Market:

- **High number of free apps.** In September 2011, Xyologic reported 66% of free apps in the US Android market, versus 42% of free apps in the US iPhone market.
Mobile Platforms: The Clash of Ecosystems

- Very few localized apps. According to Distimo’s September, 2011, report, only 4.9% of Android apps are published in one country only (indicating locally focused content).
- Rampant copyright infringement makes it difficult for legal apps to succeed (e.g. Spotify)
- Google is more interested in the ad-supported model of app monetization, i.e. in creating inventory for its apps, rather than seeing developers monetise through direct payment means.

Licensing & Governance

Google has been very successful in using open source to its business advantage. Android source code is available both in public and private source branches. The private branch contains advanced versions of the operating system, which are usually 6-8 months ahead of the public branch. The private branch is available only to a small number of OEMs working on high-profile “Google Experience” handsets.

While the base platform is available in open source, Google’s closed applications (Android Market, Google Maps, and Gmail) are proprietary to Google, and are not available in open source. These applications are licensed to OEMs under a commercial agreement, and are subject to platform compliance. Compliance is verified using CTS (Compatibility Test Suite) and CDD (Compliance Definition Document), which ensure consistency of the system API implemented in the device, and its hardware specs.

Visibility and transparency of Android open source development is severely restricted.

Google does not maintain public visibility into the Android roadmap, which is tightly controlled by Google, with limited visibility by select OEM and chipset partners. Development of new functionality is largely done in private code branches, which are not visible to public.

The vast majority of development is done by Google employees without calls for contributions, visibility into the contribution approval process, or open discussions on development priorities.

The majority of third-party contributions are done in the public tree, which is behind the version Google is currently developing. For example, there have been around 1,000 contributions from 40 companies into the Froyo (v2.2) public tree.

Contributors are requested to sign a contributor license agreement, containing a copyright license and a patent license (for licensable patents). Google is allowed to re-license the contributed code under any license they see fit.

Android maintainers (all of which are Google employees) decide which contributions will be integrated into the advanced private branch, and thus which will become part of the platform.

Tight control over the contributions and roadmap is seen as necessary for rapid platform evolution, which otherwise can be restricted by the conflicting interests of participating parties.

Android is the ‘role model’ in terms of the number and diversity of control points in open source projects. The Android control points are:

- Advanced private branches, which are ahead of public source for 6-8 months. Private branches are only disclosed to select group of OEM and chipset vendors, which gain significant competitive advantage by early access to new versions of the OS.
Mobile Platforms: The Clash of Ecosystems

- Android Market is only available subject to a commercial license, which is Google’s way of gating the availability of over 300,000 applications on handsets.
- Google Maps, Gmail and other Google proprietary applications are only allowed to be installed on the device (pre-load) if the device conforms to Google specifications, as verified by CTS maintained by Google.
- The Android trademark is held by Google, and licensed under commercial terms.
- Exceptionally fast pace of version release, combined with full control of Google over contributions
- No roadmap visibility outside a small group of select partners

The filings of the Skyhook legal suit against Google alleging non-competitive practices reveal details of how Google controls the ecosystem of OEMs. One of the most characteristic pieces of evidence is an email from Dan Morril, of Google, dated Aug. 6, 2010, which says, “We are using compatibility as a club to make them [handset vendors] do things we want”. For a deeper analysis into Android’s governance model, the reader is referred to VisionMobile’s Open Governance Index, a free report published in July 2011.

Technology Foundations

Android is based on the Linux kernel. It leverages a well-established Linux process model, driver architecture and security framework. Although Android is based on the Linux kernel, it should not be confused with the Linux operating system itself. Due to a different C library, UI framework and application life-cycle model, Android cannot directly run applications developed for Linux.

The platform introduces an extensive layer of Android-specific middleware on top of the Linux kernel. The middleware provides support for audio and video multimedia, 2D and 3D graphics, wireless networking, location services, sensors, Bluetooth, and more.

Multimedia middleware was initially based on the open source OpenCore framework by PacketVideo. In recent versions of the platform, it was replaced by a new and less complex multimedia framework based on Google’s Stagefright open source project. Graphics is supported by use of industry standard OpenGL ES 1.x and 2.0 APIs for 3D hardware acceleration.

Higher layers of the operating systems are implemented in a dialect of the Java language. Java software is executed in the Dalvik Java virtual machine, which was designed by Google from scratch. The Dalvik virtual machine is heavily optimized for a mobile environment, including processor load, run-time memory use and battery drain. Android version 2.2 introduced a JIT (just in time) compiler, further improving performance of Java-based applications on the mobile platform.

The platform includes an extensive set of built-in applications, including dialler, messaging, contacts, calendar, email, media player, and more. The applications are written in the Java language, and sometimes have access to restricted platform APIs that are not available for third-party applications.

Android offers a relatively spartan stock UI supporting multitouch, gestures and animations. It is often enhanced by OEMs’ own user experience layer (HTC Sense, LG S-Class, Samsung Touch Wiz, Sony Ericsson Rachael).
The Android user interface has gone through significant improvements in versions 3 (Honeycomb) and 4 (Ice Cream Sandwich) of the OS. The changes are focused on the support of larger displays and different tablet ergonomics. The version 3 of the OS (Honeycomb) only supports tablets, and was never released as an open source project. Changes introduced in this version were merged into the main branch in Android v4.x (Ice Cream Sandwich).

The Android Web browser is based on the WebKit open source engine, and is capable of rendering standard web pages. Google makes continuous improvements to the browser's JavaScript performance and HTML5 support. Applications can invoke HTML/CSS/JavaScript components, which are rendered using the Webkit browser engine.

Android version 2.2 introduced support for Adobe Flash, through a downloadable extension provided by Adobe.

Android application security builds on top of Linux's multi-user infrastructure. This infrastructure enforces the separation of third-party applications from the operating system, and from other applications.

Just as in iOS, at the OS level, each installed application runs as a separate ‘user’. Application access rights are governed by the permissions of that user ID. Third-party applications run in a sandboxed environment where each application runs in a separate Linux process space and has access to a dedicated file system area. When the application needs to access sensitive API (e.g. contacts, location, SMS), the developer declares the need in the application package manifest. At the time of installation, the OS requests user permission to install the application with the required rights. If permission is granted by the user, the application is installed on the device with access to the API.

The OS typically comes pre-integrated with Google cloud services. This includes Google Search with voice, Google Maps, Gmail, Google Accounts, Picasa, etc. Applications can interface and invoke the functionality of these Google services as part of the application flow.

Application development

The Android SDK is based on the popular Eclipse IDE, extended with a device emulator and Android-specific plug-ins. The plugins provide all the necessary tools to design, debug, optimize and deploy mobile applications for the platform.

Android uses a Java-based application framework for running third-party applications. The application framework uses a dialect of the Java language, but employs a unique application life-cycle model and system API. This makes Android incompatible with standard Java applications written for Java SE and ME (formerly Java SE and Java ME, respectively).

Android allows developers to embed components developed in C/C++ within Java applications. C/C+ development is done using Native Development Kit (NDK). NDK lets developers compile, build and package application components written in C/C++.

Android application frameworks are composed of multiple API groups, including web services based on the WebKit engine, 2D and 3D graphics, SQLite for structured data storage, mobile telephony,
Bluetooth, Wi-Fi, camera, location and sensors, and media support for common audio, video, and still image formats.

Android uses a novel application framework architecture that allows loose coupling of applications and services. This is based on the concept of Intents, which communicate the need to perform certain action (e.g. display a map). The Intent is forwarded by the OS to the most appropriate component that can service the request. The serving components can be easily replaced or reused.

Android applications can be installed from the Android Market. Alternatively, if the user explicitly allows “unknown sources” option, applications can be installed from other sources, such as from an email attachment, or downloaded file. Install from “unknown sources” is disabled by default, and must be explicitly allowed by the user.

The Android Market on-device client is the primary method of application discovery. The client also manages application updates. It displays notifications when updated versions of installed applications become available, and also lets the user set some applications to update automatically.

The Android Market web store is accessible using any web browser. It offers features such as device compatibility check and web-initiated download of apps (the user click an install button in the web browser, and the application is automatically downloaded to the Android handset registered for the user).

In July 2011, Google announced significant improvements to the Android Market client. These included better application discovery, as well as ability to purchase e-books and video (US only).

**Viewpoint**

Google developed Android to ensure unrestricted mass-market access to Google services from mobile devices, in other words as a platform to bridge consumer eyeballs and Google ads.

Google monetizes Android indirectly. The wide proliferation of smartphones allows Google to increase turnover from its ad inventory by driving traffic to its online properties and tapping into new mobile-specific advertising use cases. For Google, Android is also a means to commoditise intermediary markets which stand in the way of eyeballs and ads, namely handsets, networks and browsers.

The Android ecosystem faces multiple patent threats from companies trying to slow down Android momentum. Apple challenges largest Android OEM, Samsung, and Microsoft imposes a patent tax on multiple Android OEMs. Google is expected to become more active in protecting the ecosystem using extensive patent portfolio it gained with the pending acquisition of Motorola Mobile Devices.

Unless Google fails to protect the ecosystem from patent attacks, the Android platform will continue its fast evolution, improving the user experience, expanding into tablets and developing support for connected TVs. Moreover, the growing number of Android-based smartphone and tablets will help Google to lock large parts of the mobile ecosystem into Google cloud services.
**Bada**

Samsung Bada is a proprietary mobile application platform developed by Samsung for mid-range and low-range smartphones. The company announced the platform on Nov. 10, 2009, and unveiled its first Bada-based product, the Wave S8500, at the Mobile World Congress, in February 2010.

In May 2010, Samsung released a beta version of the Bada SDK. It also launched the Bada Developer Challenge, featuring prizes of US $2,700,000. Version 1.0 of the SDK was released in August 2010. The Samsung Wave S8500, the first Bada-based phone, was released on June 1, 2010.

In February 2011, Samsung announced Bada 2.0. Bada 2.0 is expected to reach markets in second half of 2011. Bada 2.0 introduces many API enhancements, including support for multitasking, NFC and the WAC API.

**Positioning**

Samsung Bada is positioned as a mobile application platform for mid- to low-end touch-screen smartphones. The platform offers service-centric capabilities to applications, including content management, location-based services, social networking and commerce services.

The current lineup of phones running Bada software includes models like the S8530 Wave II, S8500 Wave, S5250 Wave 525, S7230E Wave 723, S5330 Wave533, S5780 Wave 578, S5750 Wave575 and S7250 Wave 725.

On the low end, we find the Samsung Wave 525, which is sold for $144. The Bada flagship is the Samsung Wave II, which sells for $390 (unlocked models and pre-tax).

The platform is monetized indirectly by Samsung, through sales of mobile handsets running the OS.

**Ecosystem adoption**

Samsung is the only company selling devices based on the platform. Our estimates place Bada shipments between 9M and 14M units in the first five quarters following the platform’s launch.

Current Samsung Wave designs use Samsung’s own application processor: an S5PC110 application processor integrating a 1 GHz ARM Cortex A8 core with a POWERVR SGX540 3D graphics engine.
The Samsung Wave initially launched in Europe, on June 1, 2010. As of early 2011, it was being sold by operators in 80 countries. Samsung presents the platform as ‘operator friendly.’ It allows handset customization, and a share of revenues based on application sales.

Concurrent with the release of the Samsung Wave, Samsung opened its international Bada application store, Samsung Apps. In March 2011, Samsung Apps had approximately 13,000 applications, and had surpassed 100 million downloads. Samsung now reports a total of 40,000 items in the store, including wallpapers and ringtones. Samsung Apps is the only application store distributing Bada applications.

Some major publishers showed their support for the Bada platform at launch. These included Twitter, EA, Capcom, Gameloft, and Blockbuster. The platform and the Samsung App store offer to developers the possibility to sell their applications in a wide selection of global markets, based on the industry-standard 30/70 revenue share model.

Independent developers, however, have adopted a ‘wait and see’ approach to the platform. They are waiting for the platform to prove its monetization potential before making a significant investment.

**Technology foundation**

Samsung Bada is built as a four-layer architecture comprising kernel, device, service, and framework layers.

The kernel layer can be the Linux kernel or a proprietary Samsung real-time OS derived from Samsung’s internal SHP OS.

The device layer provides core device functions, including system and security management, the graphics and windowing system, data protocols and telephony, and audio-visual and multimedia management.

The service layer provides service-centric functions via application engines and web-service components interconnecting with the Bada Server.

According to copyright notices, the Samsung Wave S8500 software stack draws from the FreeBSD, NetBSD and OpenBSD projects.

The Samsung Bada UI framework supports multi-touch, an advance over earlier closed-OS (SHP-based) Samsung products. The platform integrates the WebKit web browser engine. Native C/C++ Bada applications can embed the Adobe Flash Player, WebKit browser, and map controls, creating “hybrid” apps that incorporate web app technologies into the application’s interactive flow.

Samsung Bada includes a Social Hub application that delivers real-time updates, using “push” technology. It supports popular email services such as Hotmail, Gmail, Yahoo! mail and others, instant messaging, and social networking sites including Facebook, Twitter, MySpace and others.

Samsung phones that run Bada have the Samsung App client pre-installed. That allows the user to discover and purchase applications from the device. In addition, the application store allows access from a PC computer, so applications can be browsed and purchased using a larger screen.

Samsung Bada only allows installation of applications from the Samsung App store. The application package is subject to certification, and signed. The OS verifies the signature at install-time, and at run-
time. Samsung Bada security is based on permissions granted according to the developer membership status, i.e., Basic or Partner.

Privileges are granted based on a mechanisms controlling application access to sensitive API and system resources. There are three privilege levels for API and resources: Non-privileged, Normal and System. The System privilege level is only available to developers having Partner membership status. Application privileges are declared by the developer in the manifest file, verified at the time of the application certification and are checked by the service layer at runtime.

**Application development**

Samsung Bada application development is based on C++. The framework layer C++ API consists of an application framework, and functions exported by the underlying layers.

There are two application types: Base applications are stored in the ROM, and cannot be removed, while Bada applications are installable and removable via the Application Manager.

Samsung Bada is designed as a service-centric platform. It incorporates various online service enablers supported by back-end Bada servers that are run by Samsung. The service enablers are exposed to developers through an open API. Social networking applications can manage user profiles, including buddy information for application users, and can integrate users on most popular social networking services. Content management applications can store and search contents on devices and servers. Location applications can use landmark stores, maps, and geocoding services. Commerce applications can sell products, such as game items in applications.

Multiple device resolutions are supported by the auto-scaling feature. Auto-scaling avoids having to rewrite applications for different screen sizes. As long as the application base resolution and the target device resolution have the same aspect ratio, the UI is automatically scaled up or down to match the target device resolution.

Bada provides multiple UI widgets for application developers, including such UI controls as listbox, colour picker, tab, etc. The UI supports multi-touch pinch-to-zoom, as well as cut, copy, and paste.

Version 2.0 of the platform will introduce multi-tasking, 3D audio based on OpenAL API, NFC, WiFi-Direct, HTML5, support for 80+ WAC APIs, speech recognition, as well as a new SDK for Linux and Mac and a faster simulator.

Samsung Bada native applications are developed in C++ with the Bada SDK, and the Eclipse based IDE. GNU-based tool chains are used for building and debugging applications on ARM processors. The IDE also contains UI Builder, with allows developers to design the app interface by dragging and dropping UI controls into forms. For testing and debugging, the IDE includes a phone simulator.

The platform also supports development with Flash Lite and web technologies (HTML, CSS, JavaScript).
Viewpoint

Samsung is close to capturing global mobile device leadership from Nokia in terms of unit sales. The company maintains a broad portfolio of devices aimed at different user segments and geographies.

The introduction of Bada is Samsung's attempt to create differentiation around its mid- and low-tier touch-screen products, and create barriers to entry for lower-cost device makers. Bada is also a strategic means for Samsung to sustain its negotiation position towards Google and Microsoft when discussing use, price and support for these platforms.

Although Samsung put significant resources in promoting Bada in Western Europe, Bada will remain focused on markets where iOS and Android have not yet gained strong positions, i.e., such countries as India, South Asia and Russia.

Long term, we believe Bada will succeed more as a negotiating card for Samsung towards Google and Microsoft, rather than a viable platform or application ecosystem.
BlackBerry OS evolved over several generations. It traces its origins to the BlackBerry 850 device introduced by RIM in 1999. BlackBerry 850 was a two-way pager that used the DataTrac data network. In 2002, RIM introduced its second generation of devices, which were more reminiscent of modern smartphones. These devices had physical QWERTY keyboards and supported push e-mail, mobile telephone, text messaging, Internet faxing and basic web browsing.

BlackBerry devices found their initial market success in providing a solution for enterprise mobile email. In recent years, BlackBerry expanded into the consumer segment, providing messaging solutions for text-addicted users. In 2009, RIM reported that more than 50% of its subscribers are consumers, and 80% of the additions to the subscriber base are coming from the consumer (non-enterprise) sector.

The latest versions of BlackBerry OS combine traditionally strong support for messaging functions (email, SMS, instant messenger) with support for personal information management (calendar, contacts), multimedia, location services, downloadable applications, access to enterprise data systems and Internet services. BlackBerry OS version 6 supports universal search, and features an embedded YouTube application, a Social Feeds application, and a full-featured browser based on the open-source WebKit engine.

Despite the wide success of the BlackBerry platform, it is widely accepted that the OS is outdated, making it difficult for RIM to compete with products based on the much more advanced iOS and Android. In an effort to modernise its handset platform, in April 2010, RIM announced the acquisition of QNX Software Systems from Harman International. QNX Software Systems is a long-time maker of the QNX operating system, used in embedded applications, including in-car infotainment systems. In October 2011, RIM announced its next generation BBX platform, which is based on QNX OS. BBX is expected to replace the legacy BlackBerry OS in future smartphone and tablet models made by RIM.

BlackBerry OS and BBX are proprietary to RIM, and are not licensed to third parties. RIM is expected to continue following the path of basing its products on a proprietary OS.
Positioning

Similar to Apple, RIM follows an integrated approach in designing BlackBerry products: RIM makes the devices, the operating system, and the embedded applications. RIM also makes messaging servers, and operates the service infrastructure for its messaging and location-based services.

RIM sells a wide selection of device models at different price points. The two main targeted product segments are:

- Enterprise messaging services, where in addition to BlackBerry devices RIM sells BES messaging servers
- Consumer email, text and instant messaging (BBM) services operated by RIM

Most RIM BlackBerry devices fall into the category of midrange smartphones. In April 2011, RIM launched its first tablet, the BlackBerry Playbook, aimed at the enterprise market.

At the time this report was written, BlackBerry devices sold for between $140, for the entry-level BlackBerry Pearl 8100, and $700, for the high-end BlackBerry Bold 9900 and BlackBerry Torch 9850 touch-screen model (all prices are for unlocked models, pre-tax).

RIM derives its revenue from multiple sources: sales of BlackBerry devices, revenue share on the service subscription with network operators, and selling BES licenses to enterprises.

Ecosystem adoption

BlackBerry OS is used solely by RIM in BlackBerry devices, and is not licensable to other OEMs.

As of September 2011, RIM’s subscriber base exceeds 70M BlackBerry users worldwide, growing at 40% year-over-year. RIM sold 10.6M BlackBerry devices in the quarter that ended August 27, 2011. The company’s revenue in this quarter was $4.2B, 73% of which came from hardware, and 24% from services. Quarterly net income was $329M.

The majority of BlackBerry sales come from the North American market, with traditionally strong support for BlackBerry by one of the leading US carriers, Verizon Wireless.

In the US, BlackBerry is traditionally strong in the enterprise market. In Western Europe, (particularly UK and France), the Gulf, Latin America and the Pacific, RIM BlackBerry found success among text-addicted consumer users (from teens to 30s), who use the platform for socializing. The strong adoption by consumer users is due in part to the successful integration of Facebook and BlackBerry Messenger (BBM). BBM is exclusive to BlackBerry OS and is only available on RIM devices.

BlackBerry OS enjoys a wide and well-developed operator ecosystem. Over 335 operators in 130 countries sell BlackBerry devices and services.

The relatively low price of entry-level BlackBerry models places these devices in the range of products sold for pre-paid subscribers. For example, Carphone Warehouse in UK sells a range of pre-paid BlackBerry models.
Operators are generally positive toward BlackBerry devices, and actively promote RIM products. This is due to the additional service revenue generated by BlackBerry subscribers, which increases the operators’ ARPU metrics.

As of September, 2011, BlackBerry App World, the official application store for the platform, contained over 35,000 applications. Although BlackBerry App World has far fewer applications than iOS and Android, the BlackBerry application portfolio offers a fair selection of utilities, games and cloud-based applications.

BlackBerry App World does support paid applications. However, BlackBerry applications are typically priced higher than applications in Apple’s App Store or the Android Market. Furthermore, monetizing BlackBerry applications can be challenging, due to the relatively poor usability of the App World client, and the client’s lack of availability for a large number of legacy devices.

Development of Enterprise BlackBerry applications is known to be more difficult and time-consuming than development on iOS and Android. This is due to a less capable application framework, inconsistent connectivity API, and significant fragmentation between device models and OS versions. VisionMobile’s 2011 Developer Economics report indicated Blackberry as the third most fragmented platform, after Symbian and Java ME.

**Technology foundations**

BlackBerry OS is a proprietary operating system. In current models, it runs on ARM11-based application processors from Qualcomm and Marvell.

BlackBerry OS supports the following:

- Push email, which can be integrated with enterprise messaging systems like Microsoft Exchange, IBM Lotus Notes, and Novell Groupwise
- Personal information manager (PIM), including address book, calendar and to-do list, which can be synchronized with enterprise collaboration systems
- Office compatibility suite supporting Microsoft Office file format for documents, spreadsheets and presentations. Office compatibility suite was developed by DataWiz, which was acquired by RIM in September 2010.
- BlackBerry Messenger – instant messaging integrated with unique BlackBerry identity (PIN)
- Unified messaging application aggregating push notifications from multiple applications, including third party applications
- Phone dialer supporting integration with enterprise PBX systems
- Media player supporting video and music playback
- Video and picture capture
- Extensive support for device policies, allowing enterprise IT to manage devices and protect the data residing on the device
- Internet-based services, including Facebook and YouTube
Mobile Platforms: The Clash of Ecosystems

- WiFi networking
- Multitasking
- Location services with navigation capabilities

Recent BlackBerry devices have a pre-installed App World client. The client allows discovery and purchasing of applications. The applications can be updated from the client, or using a PC-based utility.

BlackBerry OS is designed to meet stringent enterprise security requirements. It affords control over what applications may be installed on a device, and what data and device capabilities applications may access.

Many application control settings can be managed by end-users. If the user is connected to a BlackBerry Enterprise Server, the IT administrator may establish network-wide policies. Controls and policies establish whether applications can make network connections, play media and access the BlackBerry Calendar.

The BlackBerry user interface is designed to support efficient messaging, with a wide array of messaging features. Most models have a physical QWERTY keyboard and track-ball, and are optimized for one-handed operation. RIM’s attempts to reproduce the effectiveness of the user interface on touch-screen devices have met with moderate success.

In 2009, RIM acquired Touch Mobile, which specializes in WebKit development. Starting from BlackBerry 6 OS, the operating system provides a full-featured web browser based on WebKit open-source engine. The browser supports HTML5 extensions, including storage, location and multi-threading.

In order to reduce the amount of transmitted data and minimize power consumption, the BlackBerry browser works with a network-based proxy server. The proxy server aggregates information from multiple Web servers, repackages and compresses the contents before sending it to the device.

In September 2010, RIM announced the introduction of the Playbook tablet, based on the QNX OS. The tablet is designed as a companion device for BlackBerry smartphones, and employs device-sharing capabilities originally developed for QNX-based in-car infotainment systems.

In October 2011, RIM announced plans to replace its legacy BlackBerry OS in smartphones with a more modern BBX platform based on QNX OS.

Application development

BlackBerry OS offers developers multiple options for application development, including Java and web development.

BlackBerry OS provides a Java-based application development framework based on the J2ME MIDP 2 specification. RIM extended the framework with proprietary APIs that built-in applications can use: background threads with server push; communication with Bluetooth peripherals; multimedia capable of audio playback, recording and RTSP streaming, as well as video playback and streaming; location-
Mobile Platforms: The Clash of Ecosystems

based services; access to PIM data, UI widgets, layouts and transitions; cryptography and security functions with PKI and elliptic curve support.

The Java-based application framework provides an API for discovery of device capabilities, including cradle, physical keyboard, virtual keyboard, rotation capability and touch-click capability.

BlackBerry Mobile Data System (MDS) is a flexible framework for application development that enterprises can use to add and manage applications. The OS supports discovery and discovery policies for MDS applications in an enterprise environment.

BlackBerry Java applications can be developed and debugged using an Eclipse-based IDE and the BlackBerry Java Plug-in for Eclipse. The latter provides tools to create, debug, optimize and localize Java applications.

The BlackBerry Java Development Environment (BlackBerry JDE) provides similar capabilities, but is a legacy, standalone development environment. Both SDKs include BlackBerry Smartphone Simulators that enable developers to test applications without the need for a physical device.

Installable web applications can be developed using the Widget Development Architecture. BlackBerry widgets are standalone web applications that use HTML, CSS and JavaScript, along with the BlackBerry widget APIs. The widget APIs provide device-specific information, and access to on-device data stores such as the SD Card file system, SQLite database, UI widgets, and secure remote servers.

The BlackBerry Widget SDK consists of the Widget Packager for packaging web assets into installable BlackBerry Widget format, BlackBerry Smartphone Simulator, HTTP Proxy to simulate BlackBerry email, documentation and code samples. Widget development can be done using the widely used Eclipse or Microsoft Visual Studio IDEs, extended with a BlackBerry Web Plugin.

**Viewpoint**

The traditional BlackBerry strengths have been diminishing in importance, as the market reacts to the success of iOS and Android.

RIM faces increasing competition in both the fast-growing consumer segment and its core-strength enterprise markets. Apple’s iPhone and iPad products are extremely successful in the consumer market, and have gained popularity in the enterprise, especially with the iPad tablet. Android is also showing strong growth. Android-based smartphones overtook BlackBerry in Q2, 2010, in terms of the number of devices sold per quarter in the US market. Moreover, there are multiple reports of decreasing satisfaction levels by current BlackBerry users. Such reports hint at the possibility users could migrate to competing platforms.

In order to maintain its leading position and financial results, RIM must keep increasing device shipment volume without compromising its gross margins.

RIM is slow to respond to the challenges posed by the competition. Attempts to introduce a touch-screen device (BlackBerry Storm, Storm 2 and Torch models) were largely disappointing. New devices with physical keyboards are basically variations on the same product concept, lacking the
significant innovation needed to restore BlackBerry’s differentiation. The much-anticipated PlayBook tablet failed to impress, and showed dismal sales.

RIM is torn between two very different market segments: enterprise mobile messaging, and text-addicted consumers. Amidst troubling signs for RIM’s future, the company needs to reconcile its dual personality. Unless RIM creates separate product experiences for business users and consumers, it will continue to struggle in the face of competition from Apple and Android-based OEM.

Investors are pushing RIM towards a radical strategic restructuring. However, such transitions can be lengthy and painful. It is not clear that investors will have patience to wait for the new platform strategy to bear full fruit.
Mobile Platforms: The Clash of Ecosystems

BREW MP

The original BREW platform was introduced in September 2001, and was deployed mainly by mobile operators in US CDMA markets.

The BREW platform consists of an application runtime and an end-to-end software distribution system. It enables developers and mobile operators to deploy and monetize applications to BREW-enabled handsets. BREW stands for Binary Runtime Environment for Wireless.

Qualcomm BREW provides a vertically integrated solution for developing, distributing and monetizing applications on handsets based on the company’s chipsets. The solution includes compatible chipsets, device runtime, application delivery servers, as well as application catalogue accompanied by application vetting and authorization services.

The solution is aimed at feature-phones and low-end smartphones. It is supported on a wide range of chipsets, from single processor core (MSM6xxx series) to dual processor core (MSM7xxx) chipsets to high-performance QSD8xxx Snapdragon chipsets.

Positioning

VisionMobile estimates, as of Q3, 2011. Device prices refer to unlocked models, pre-tax.

<table>
<thead>
<tr>
<th>Owner</th>
<th>Qualcomm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launched</td>
<td>September 2001</td>
</tr>
<tr>
<td>Devices sold since launch</td>
<td>725 million (250 million BREW-enabled)</td>
</tr>
<tr>
<td>Apps downloaded</td>
<td>N/A</td>
</tr>
<tr>
<td>Apps available</td>
<td>18 thousand</td>
</tr>
<tr>
<td>Device price range</td>
<td>$139 to $399 USD (handsets)</td>
</tr>
</tbody>
</table>

In October 2010, Qualcomm acquired iSkoot, which specializes in enabling mobile access to Internet services such as social networks, consumer email, and IM products. It is expected that iSkoot technology will be integrated with BREW MP, adding Internet service capabilities to the platform.
Mobile Platforms: The Clash of Ecosystems

As of June 2011, the INQ Chat 3G handset based on BREW was available for US $162. The higher-end HTC F5151 Freestyle was offered for $399.99.

BREW and BREW MP are monetized indirectly, by driving sales and increasing the competitiveness of Qualcomm’s chipset offering. BREW and BREW MP are provided for free to licensees of Qualcomm’s chipsets.

Ecosystem adoption

As of October 2011, Qualcomm reported that over 70 device OEMs had made over 1,400 handset models supporting BREW. The list includes HTC, ZTE, LG, Samsung, INQ Mobile, and Pantech. Qualcomm reports that the global installed base of BREW-enabled devices is over 250 Million.

BREW-enabled handsets can run third-party applications and access a catalogue of ready-available applications. Qualcomm reports that as of Q1, 2010, about 18,000 applications were available for the BREW platform, excluding ringtones and screen savers.

BREW is traditionally strong in the US, and was subsequently deployed in Latin America, China, India and Asia Pacific. Qualcomm reported that as of June 2010, over 45 mobile operators around the globe support BREW in their offerings.

The BREW operator ecosystem originally formed around CDMA carriers, but Qualcomm has actively pursued GSM/UMST operators, albeit with limited success. AT&T has released a series of BREW-based handsets aimed at messaging, like the HTC Freestyle (released February 2011) and the Pantech Pursuit II (released July 2011).

As of June, 2010, Qualcomm reports over $3B+ in cumulative developer earnings over nine years. This has benefited mostly established software houses, although BREW is much less appropriate for small independent developers.

Developers are overall dissatisfied with the overall cost of taking a BREW application to market. The costs include charges to get the SDK and associated development tools, as well as charges for testing and submitting applications to the application store managed by Qualcomm. In our Developer Economics 2011 survey, 28% of developers using BREW are planning to abandon the platform.

Each application must pass through two acceptance gates, i.e., both Qualcomm certification, which is very stringent, and individual operator certification, with operator certification requirements varying widely.

The complex and expensive certification process is perceived as an advantage at times by established software developers. Difficulties associated with testing and development costs create a high barrier to entry for developers with low budgets and little time, resulting in less market dilution.

AT&T’s BREW program has attempted to address these challenges by lowering certification costs from $500 to $100, and cutting application time-to-market by half, it claims.

BREW applications are available through the BREW Delivery System, which offers operator in-network and Qualcomm-hosted deployment options. BREW managed services is a turnkey solution that provides a Qualcomm-hosted shared catalogue for operators without any dedicated staff for the BREW service.
Mobile Platforms: The Clash of Ecosystems

Technology foundations

The BREW platform is based on a proprietary Qualcomm operating system called AMSS or REX. The middleware includes APIs for functions as connectivity, databases, languages, media, networking, security, telephony and UI. BREW by itself provides a set of enablers for building UI, but does not contain the UI itself.

BREW is focused on feature-phones and low-cost smartphones, and is not directly suitable for other device types, such as tablets and connected TVs.

BREW by itself does not provide embedded applications – these need to be developed by the handset maker or licensed separately. For example the Opera Web browser is pre-integrated with the platform, but is not part of the platform.

BREW third-party applications are discovered using operator-branded operator stores or on-device client (depending on the model).

Most of the BREW application development is done using C/C++, though Java or Flash Lite can be supported as well.

Viewpoint

By developing BREW, Qualcomm aims at increasing sales of its chipset designs and improving their competitiveness, especially towards low-cost chipset competitors such as Mediatek.

With new versions of the platform (BREW MP), Qualcomm pushed to increase completeness, with the aim of reducing time-to-market and development cost for BREW MP designs. It partnered with third party vendors to pre-integrate additional software components, such as web browsers, messaging software and navigation solutions - more than 25 companies to date.

Although BREW devices are still shipping, there are indications from both OEM and operators that the platform is close to its expiry date. BREW’s main advantage was its ability to run on low-cost chipsets. As “smart” devices are increasingly associated with touch-screen UIs, this advantage becomes less important, in comparison to the high BOM (bill of materials) cost of a large display and touch-screen components.

Moreover, Qualcomm has not been able to modernise its decade-old application platform and streamline the certification and go-to-market route following the standards set by Apple, Google and Microsoft. As such the BREW MP platform has been decaying for the last year in terms of both developer mindshare and OEM adoption.

Since Android and Windows Phone 7 are much more effective in driving Qualcomm chipset sales that BREW, we see the vendor soon discontinuing BREW platform development.
iOS

iOS is a proprietary platform owned by Apple, Inc. The platform is not licensed to third-party device makers, and can be found only on devices designed and sold by Apple.

Apple announced iPhone, the first device based on the platform, in January, 2007, at the MacWorld Conference & Expo. Initially the platform was called "iPhone OS," but it was renamed to iOS with the introduction of iPad tablet in January 2010. iOS is a key factor in the unprecedented success Apple has enjoyed with iconic iPhone, iPod Touch and iPad products.

iOS provides a full application platform “value-stack” for mobile and portable products. The iOS stack includes the operating system itself, comprehensive middleware, an application framework, web services framework and application store, as well as integration with iTunes content delivery services and iCloud on-line services (announced in June, 2011).

Apple makes most of its revenues and profits selling iPhone, iPad and iPod devices at a premium price. The unprecedented success of these products helped Apple become the world’s second-richest public company, behind only Exxon Mobil. Apple reported in October 2011, that the company earned $6.6B in Q4, FY2011, on revenues of over $28B. Including short- and long-term marketable securities, it has to date amassed “cash” reserves of more than $81 Billion (US).

Positioning

Current iOS versions target high-end mobile and portable devices, with strong emphasis on multimedia and gaming. The price range for devices running iOS starts at the low end, with iPod Touch portable media players, and extends to iPad tablet computers at the high-end.

iOS is monetized by Apple indirectly by being pre-installed on Apple iPod Touch, iPhone and iPad devices, and included in the device price. Platform updates are provided by Apple with no additional charge to existing users. Advanced functionality and the premium user experience provided by the platform allow Apple to sell iPod touch, iPhone and iPad devices at a significant premium.

As of October, 2011, the least expensive iPod Touch (4th Generation) has 8GB of storage retails for $230. The iPad 2 with most expensive configuration, supporting Wi-Fi, 3G and having 64GB of storage, is sold by Apple for $999.99.*
Mobile Platforms: The Clash of Ecosystems

Ecosystem adoption

As of October 2011, (fiscal Q4, 2011), Apple has sold 250M iOS units since the launch of the first iPhone in 2007.

Apple’s iPhone smartphone is sold by many operators around the world. Until February, 2011, AT&T enjoyed an exclusivity agreement with Apple in the US, and was the only US mobile operator to offer the device. In February, 2011, Apple introduced iPhone 4 versions supporting CDMA networks, and offered by Verizon Wireless.

In other countries (e.g. France), regulators prevented such exclusivity agreements, and the iPhone was offered by multiple operators.

Operator subsidies boosted iPhone penetration in markets dominated by post-paid subscriptions. Operators typically subsidize the iPhone in order to increase ARPU (average revenue per user), taking a hit on device costs in order to sell customers into higher-priced, longer-term subscription plans and data packages.

Despite the high ARPs generated by iPhone sales, network operators express mixed attitudes towards Apple. The Californian consumer electronics company has been very stringent in blocking any attempts by operators to customize or co-brand Apple devices. This limits the differentiation opportunities for operators, and increases the subsidies that operators have to spend in order to attract users to their voice and data plans.

Apple has established a golden standard in building a large and thriving developer ecosystem around iOS. As of October 2011, there are over 500,000 apps available in the Apple App Store.

iOS developers are generally very positive about the platform, praising the ease of creating compelling applications. They also praise iOS’s user demographics: the iPhone has attracted an affluent group of users that tends to be favourably inclined toward purchasing applications.

The Apple App Store is the only official store where iPod/iPhone/iPad users can buy apps. The store supports iTunes-based billing, with a 70 percent revenue share for developers.

Some independent developers achieved multi-million dollar revenues by selling their software in the Apple App Store. However, aside from a relatively small number of success stories, it is difficult for a new application to rise above the noise, gain user popularity and achieve significant revenues.

Technology foundations

iOS is based on a scaled-down version of MAC OS X, the operating system used by Apple in its personal computers. Compared to Mac OS X, iOS is optimized for touch-screen user interfaces, ARM-based processors and hardware configurations lacking hard disks and physical swap partitions.

iOS is a UNIX-like operating system that, like Mac OS X, was derived in part from BSD Unix. Like Mac OS X, iOS uses a “hybrid” kernel combining elements of both microkernels and monolithic kernels. As with microkernels, device drivers run in userspace. As with monolithic kernels, though, core OS
features like the networking stack, process model and users/groups security model run in kernel space. Apple distributes open source portions of the kernel and OS via the Darwin project that it sponsors.

The iOS operating system offers a rich set of middleware components, including multimedia, 3D graphics, networking, web services, UI and more. The middleware functionality is exposed to application developers via a rich set of public APIs, allowing the creation of engaging applications and games.

Devices running iOS come preloaded with an extensive set of applications, including:

- A phone dialler and SMS app (iPhone-only)
- Email, contacts and calendar applications
- iPod media player with iTunes integration
- Apps Store for buying and installing applications
- YouTube and Google Maps applications for using Google cloud services
- The Safari Web browser

The iOS user interface is based on direct manipulation of 3D screen objects using touch controls, together with touch and multi-touch. The gestures include swiping, tapping, pinching, and reverse pinching. There is single application window visible to users at any given time. The foreground application takes the full area of the display.

The OS supports the use of device accelerometer sensors to create additional UI functionality. For instance, rotating the device can switch the display from portrait to landscape mode.

iOS has succeeded on several different form-factors: from iPhone/iPod Touch devices with 4.5-inch displays, to iPad having larger 9.56-inch displays.

The Safari Web browser in iOS is based on WebKit, an open source browser engine. It is capable of rendering standard web pages. Safari has best-in-class support for HTML5 extensions. Adobe Flash is not supported on iOS.

Native iOS applications can invoke WebView screens defined using HTML5/CSS/Javascript. Such screens are rendered by the same WebKit engine underlying the Safari browser.

With the introduction of iCloud and iOS 5, Apple began addressing a platform weakness – the dependency of iOS devices on PCs for device activation and synchronization.

iCloud online services, announced in June, 2011, support storing music, photos, apps, calendars and documents on remote servers managed by Apple. Any changes to the information are wirelessly pushed to all devices registered for that user.

Application development

iOS applications are developed using proprietary Apple tools available only for Apple Mac OS computers. The tools are based on the same “XCode” development suite used to build Mac OS applications.
Mobile Platforms: The Clash of Ecosystems

The iOS application framework allows developers to access platform functionality using a rich set of APIs (application programming interfaces). The iOS APIs are organized in the following layers:

- Cocoa Touch
- Media
- Core Services
- Core OS

Core OS and Core Services layers contain the fundamental interfaces for iOS, including those used for accessing files, low-level data types, Bonjour services, network sockets, and others. These interfaces are mostly written in the C language, and include API groups such as Core Foundation, CFNetwork, SQLite, and access to POSIX threads and UNIX sockets, among others.

The Media layer provides API for graphics, audio, and video functionality of the device used for multimedia applications.

The high-level frameworks in the Cocoa Touch layer mostly use the Objective-C language. The Foundation framework provides object-oriented support for collections, file management, network operations, and more. The UIKit framework, which is part of this layer provides the visual infrastructure for the application, including classes for windows, views, controls, and the controllers that manage those objects. Other frameworks at this level give access to the user’s contact and photo information and to the accelerometers and other hardware features of the device.

The OS supports Bonjour zero-configuration networking architecture for publishing and discovering services on an IP network on the same LAN.

iOS application security leverages UNIX-based, multi-user infrastructure for separation between applications and the OS itself. At the OS level, each installed application runs as a separate ‘user,’ and the application access rights are governed by the permissions granted to the corresponding user ID.

Third-party iOS applications run in a sandboxed environment. Each application runs in a separate UNIX process with restricted permissions, and has access to a dedicated file system area. If the application attempts to access a sensitive API (e.g. contacts), the OS requests user permission. If it is granted, the application gains access to the API.

iOS only allows applications be installed from Apple App Store. The application package is signed and the signature verified during installation and at run-time.

Each iOS device comes with a pre-installed App Store client, allowing users to discover and purchase applications while using their device.

Each application undergoes App Store certification before appearing in Apple App Store. An on-device client checks for updated versions of installed applications, and notifies the user when updates are available.
Viewpoint

Since its launch in 2007, Apple's iOS has emerged as one of the most successful mobile platforms, acting as a benchmark for its competitors. It combines strong technical foundations, unique product experience, and a sophisticated application platform strategy.

iOS software will continue to be central to the unique product experiences characteristic of Apple products. These unique product experiences are the key factor allowing Apple to sell its products at premium prices.

Apple’s platform strategy has created a strong application ecosystem around iOS, which facilitates user lock-in and drives platform expansion, based on both strong network effects between users and developers, and strong production economics from very few product SKUs.

Apple operates as a vertically integrated company, where it controls the end-to-end product experience, from hardware design, to operating system, services and content. This integrated model allowed Apple to create new markets and change the basis of competition in the existing markets.

It would take a complete reversal of the rules of the game to challenge Apple iOS’s leadership position. However, such reversals are not unheard of, in the fast-moving mobile industry, as shown by Nokia’s downfall since its 2007 glory days.
Windows Phone

Windows Phone 7 is a new mobile platform by Microsoft. It represents a radical departure from Microsoft's earlier platform, Windows Mobile OS.

Windows Phone 7 was first announced in February 2010, at Mobile World Congress in Barcelona. Several tier-1 mobile operators took part in the announcement, including AT&T, Deutsche Telekom, Orange, SFR, Sprint, Telecom Italia, Telefónica/O2, Telstra, T-Mobile USA, Verizon Wireless and Vodafone.

Microsoft began development of its mobile products back in 1998, with the first device introduced to the market in April 2000, under the name of 'Pocket PC 2000'. The platform evolved and eventually became the Windows Mobile operating system. It was traditionally viewed by Microsoft as a companion product line for its core Windows and Office products. Until the introduction of Android, Windows Mobile was the primary open licensable operating system for smartphones.

Microsoft began working on the new Windows Phone mobile platform at the end of 2008, when it became apparent that the legacy Windows Mobile OS could not compete with the more modern iOS and Android platforms. The project started from scratch, building on designs of the Zune HD portable media player, the Silverlight UI engine and the Xbox gaming console.

In February 2011, Microsoft and Nokia announced an extensive partnership. The deal calls for Nokia to base its future smartphone devices exclusively on Windows Phone.

Similar to other Microsoft software products, Windows Phone 7 is proprietary to Microsoft and available under a "closed source" model. The rights to the operating system are owned by Microsoft. There is no backwards compatibility with Windows Mobile: legacy applications developed for Windows Mobile will not work on Windows Phone 7.

Positioning

Windows Phone 7 offers a complete value stack. The stack includes the core operating system, an extensive set of built-in applications, application store, and integration with Microsoft's rich set of online services. Phone 7 is also accompanied by suite of sophisticated development tools.
Initial versions of Phone 7 targeted the high-end smartphone market dominated by iPhone and high-end Android models. The platform called for a WVGA (800x480) touch-screen, 1GHz ARM-based application processor, a GPU with Direct X hardware acceleration, a large amount of memory, WiFi, sensors, GPS and an optional physical keyboard. A subsequent update (Mango) brought the platform down-market, relaxing CPU requirements to a minimum of 800MHz.

In the end of October 2011, an unlocked HTC Titan Windows Phone 7.5 was selling for $675, while on the low end a Dell Venue Pro Windows Phone 7 could be had for $265 (unlocked models, pre-tax).

Microsoft monetizes Windows Phone 7 through a combination of a licensing fee and indirect revenue streams from its Windows, Office and Xbox product lines.

The direct licensing fee has not been publicized by Microsoft, and varies from OEM to OEM. An estimate of the fee places it at $8-$10 per handset, i.e., in the lower range of previous licensing fees Microsoft was reportedly charging for Windows Mobile ($8-$15). Based on information from Microsoft, the license includes patent litigation protection by Microsoft.

Ecosystem adoption

The initial launch of the Windows Phone 7 platform, in October 2010, was supported by HTC (four models), Samsung (two models), LG (two models) and Dell (one model). All these OEMs are also making Android-based models.

Additional OEMs reported to be working on Windows Phone 7 models include Sony Ericsson and Asus.

At the platform’s launch, handsets running Windows Phone 7 software were available in the Americas (USA, Canada, and Mexico), Western Europe (France, UK, Germany, and Spain) and Asia Pacific (Singapore and Australia).

Besides Android, Windows Phone 7 is the only mainstream alternative to OEMs for producing high-end touch-screen smartphones with a competitive app ecosystem. Compared with Android, Windows Phone 7 offers two main advantages for the OEMs: less effort in software development, and patent infringement indemnification.

Windows Phone 7 smartphones all have very similar hardware specs. They are based on Qualcomm QSD8250 (1GHz), QSD8650 (1.3GHz) or MSM7x30 (0.8-1GHz, Mango version) chipsets integrating the “Snapdragon: application processor. Windows Phone 7 is pre-optimized for these chipsets, leaving little work for the OEMs besides software integration.

Microsoft tightly controls the user experience of Windows Phone 7 handsets. OEM are limited in ways they can customize devices. The only possibility for OEM to differentiate is to pre-install limited set of add-on applications on the device. Changes the OS functionality and UI are not allowed. Compared to Android, this significantly limits the ability for OEM differentiation.

Windows Phone 7 was met with a positive reception from mobile operators. At the initial launch, the platform was available from 30 mobile operators around the world, including the Americas (AT&T, T-Mobile USA, TELUS Canada, América Móvil Mexico), Western Europe (O2 UK and Germany, Orange France and UK, SFR France, Movistar Spain, Deutsche Telekom Germany, Vodafone UK, Germany and Spain) and Asia Pacific (SingTel Singapore, Telstra Australia, Vodafone Australia).
Operators are looking at Windows Phone 7 as a possible source of differentiation to compensate against the extraordinary strength and closeness of Apple’s iPhone brand. Operators expect that Microsoft will bear most promotional costs for the new platform. This compares favourably with Android, as Google makes little effort to promote the Android brand to consumers.

The far-reaching partnership between Nokia and Microsoft that was announced in February 2011, is expected to bolster the position of the platform. Nokia’s device design expertise and distribution reach has the potential to significantly accelerate adoption of the Windows Phone platform.

Microsoft targeted PC and Xbox developers when designing Windows Phone 7. These developers can use the same tools, languages and API frameworks as on their respective platforms.

At the BUILD conference in September 2011, Microsoft reported there were 50,000 registered Windows Phone 7 developers, while the Windows Phone Marketplace had over 30,000 applications.

The platform was met with considerable enthusiasm from developers currently developing for other Microsoft platforms, using the .NET and XNA frameworks. Active development of apps began before availability of real hardware. The Windows Phone SDK Beta was downloaded 200,000 times in its first two days of general availability.

Naturally, mobile developers currently developing for iOS and Android are more sceptical about the platform.

Windows Phone Marketplace is the only channel for application distribution for Windows Phone 7. Microsoft manages the application vetting and approval process, according to a published set of policies (e.g., PG-13). It allows developers the industry-standard 70% revenue share.

Microsoft is acting aggressively to attract developers to the platform. The company leverages its established network of software development and training partners to educate potential developers and provide them with code and tools for the platform.

Developers of popular titles for iOS and Android were actively persuaded by Microsoft to release versions of their apps for Windows Phone. In some cases Microsoft covered the development costs (e.g. Foursquare).

In an unusual step, LG and Microsoft teamed up to provide free applications. Via the LG Application Store, accessible from LG Windows Phone 7 models, they will allow free downloads. Ten free apps will be promoted at a time. Many would ordinarily cost $30 or more. They will be available for free for 60 days, subsequently replaced by a new set.

Microsoft has put considerable effort into ensuring that the Windows Phone Marketplace offers sufficient monetization opportunities for developers. The store supports multiple billing channels (operator billing, credit cards), trial APIs, and controlled Beta procedures.

Microsoft can leverage considerable Xbox monetization experience, having created over $1B in Xbox sales in 2009-2010. For example, Microsoft reports that based on its Xbox experience, trial APIs can lead to three- to four-fold revenue increases.
Mobile Platforms: The Clash of Ecosystems

Technology foundations

Windows Phone 7 is built on top of the Microsoft proprietary Windows CE 6 kernel. The kernel provides fundamental system services such as memory management, process abstraction and scheduling. Compared to the Windows CE 5 kernel used on previous generations of the Windows Mobile product line, the Windows CE 6 kernel is touted as significantly more advanced.

Windows Phone middleware supports 3G data, Wi-Fi networking and Bluetooth connectivity. A built-in connection manager handles Wi-Fi and cellular data connections according to a pre-configured policy. The new graphics and rendering engine of the platform is based on the Direct 3D 11 API, with hardware acceleration. The OpenGL ES API is not supported.

The platform comes with an extensive set of built-in applications typical of a modern smartphone. This includes email clients, calendar, and contacts with Facebook integration. It supports full multimedia capabilities, including a camera, image viewer, video player, music player and video recording (with 720p resolution on some models). There’s also Bing Maps, a Twitter client, a Zune multimedia player with integration with Zune content services, mobile versions of Office applications, and a full web browser.

The Windows Phone 7 user interface is based on Microsoft’s Zune HD portable media player. User interaction flow is based on the concept of tiles and hubs. Somewhat similar to widgets, tiles are fixed-size areas on the device home screen that display application information and allow quick access to device functions. Hubs group device functions by type. For example, there’s a People hub, Messages hub, Music and Multimedia hubs, Games hub, Office hub, and so on.

The Windows Phone 7 web browser is based on IE7 and IE8 rendering engines, and is capable of displaying full Internet pages. The ‘Mango’ platform update has introduced an IE9 web browser. Contrary to its PC counterparts, the Windows Phone 7 browser does not support ActiveX plugins, or Adobe Flash. The browser user interface is adapted for mobile device screen sizes and touch-based controls, including multi-touch.

Windows Phone provides a capabilities-driven security model, where a user must opt-in to certain functionality within the application. For certain scenarios, Windows Phone APIs invoked by applications may require specific security permissions or user disclosure when run.

Windows Phone 7 supports separation of applications at the process level, as well as at the file-system level (every application only has access to its own files). The application security is enforced by the Windows CE 6 kernel.

Windows Phone 7 applications are discovered using pre-installed Windows Marketplace client. The client notifies the user that there is a new version of the installed application. Applications can only be installed from Windows Marketplace after formal submission, verification and vetting process by Microsoft.

Application development

The application framework is built upon Microsoft Silverlight and XNA software frameworks. It is focused on supporting consumer entertainment experiences.
The Silverlight UI framework supports event-driven applications based on XAML. It is also used for desktop and Web development. The XNA framework is used for development of loop-based games. This framework is also used for Xbox game development.

In September 2011, Microsoft released a significant update to Windows Phone called “Mango”. One of the important features added in Mango is multitasking. Multiple applications can reside in memory simultaneously, while fast switching between those applications will be done using a card-based UI motif inspired by webOS. Background applications are able to generate audio and download files. However, the multi-tasking model puts a number of restrictions on what applications can do in the background. There is no support for long-running background network connections, for example. This will make such applications as VoIP and instant messaging difficult to implement.

The notion of Live Agents introduced in Mango allows developers to write small background tasks running on the phone itself. These tasks pull notifications and respond to events such as moving to a Wi-Fi connection, or plugging into AC power.

Beyond multitasking, Mango introduced direct access to the camera, access to the sensors, the ability to interact with the calendar or contacts, network sockets and numerous other API improvements (some 1,500 new API calls were introduced).

The Windows Phone application development environment is based on the widely popular Visual Studio integrated developer environment. The Free Visual Studio 2010 Express for Windows Phone SDK contains the Windows Phone Emulator, Silverlight for Windows Phone, XNA Game Studio 4.0 and Expression Blend 4 for Windows Phone.

**Viewpoint**

The licensing fee is not the sole or even the main reason for Microsoft to develop Windows Phone 7. The key driver is the need to protect the company’s core businesses – Windows and Office suite PC software licensing – from “ecosystem churn”.

The explosion of mobile applications and the extraordinary success of Apple iOS and Google Android ecosystems put Microsoft’s core products and businesses under direct threat. Users exposed to Apple or Google products are more likely to abandon Windows-based PCs in favour of Apple computers, or abandon Office products in favour of Google Docs and Google Apps productivity suites.

In addition, Windows Phone will serve as a mobile platform for new online products capitalising on existing Microsoft cloud services, such as Bing Search, Bing Maps, Windows Live, Azure and Xbox Live.

Windows Phone has a good chance to become the third app ecosystem, behind Apple iOS and Google Android. The big question is, how far will it be from the two leaders?

In order to succeed in the shadow of well-entrenched leaders, Windows Phone needs to offer unique and groundbreaking capabilities. Microsoft must innovate quickly, to advance the state of the platform, and jump-start network effects between users and developers.
Nokia will play decisive role in Windows Phone endgame by bringing Nokia’s brand and distribution power to the service of the platform. The partnership between Nokia and Microsoft seems natural at the first glance, but there are many internal conflicts in it. Nokia has chosen Windows Phone over Android to avoid competing in this quickly commoditising segment where multiple device makers compete with each other on price. However, Microsoft is interested in spreading Windows Phone to as many devices as possible through as many OEM as possible. Windows Phone is not exclusive to Nokia - Microsoft is signing up multiple OEMs, including Nokia’s arch-competitor Samsung, which will make everything it can to cut off Nokia in its quest to become #1 handset maker in the world.
Palm announced its webOS smartphone operating system at CES in January 2009. The first device, Palm Pre, was launched in June 2009, with Sprint in the US.

Palm has had a long history of developing mobile and portable devices, starting from the iconic Palm Pilot PDA introduced in 1996, which was based on Palm OS software. Over the years, the company introduced a line of successful Palm OS smartphone products under the Treo brand. With multiple changes in the company ownership, it eventually lost rights to the Palm OS operating system, which now belongs to ACCESS Company, Ltd.

In 2007, following investment by Elevation Partners fund ($325M initially), Palm began efforts to refresh its product line under the leadership of John Rubinstein, the former Apple executive who led iPod hardware development. WebOS was conceived to become an “iPhone killer” by exceeding the capabilities of iPhone OS.

WebOS was developed by Palm from the ground up. It was designed to address the shortcomings of the initial version of the iPhone OS, such as the lack of multitasking, lack of cut and paste, and the proprietary application development environment.

WebOS introduces novel UI concepts and allows efficient multitasking interactions. It let Web developers use familiar tools and languages (HTML/CSS/JavaScript) to create apps for the device.

WebOS has also pioneered a deep integration of social networking. The "Synergy" feature embed feeds from multiple social networks into built-in applications in the device UI.

Despite high expectations, Palm and webOS failed to make a significant impact in the smartphone market. Palm was acquired by HP in April, 2010, for $1.2B. Following the acquisition, HP holds all the rights for webOS.

Following acquisition by HP, Palm became the basis for the Palm Global Business Unit responsible for webOS development. In October 2010, Palm/HP announced webOS 2, with UI enhancements, support for Flash, readiness for Skype, and more. This version first appeared on Palm Pre 2 devices running on 1 GHz processors.

In February 2011, HP announced webOS 3, running on the company’s TouchPad tablet computers.
Following leadership changes, HP announced plans to discontinue its webOS device business, including TouchPad tablets and Pre smartphones. Despite HP’s attempts to present an alternative strategy for the OS, it is almost certain that webOS has no future.

**Positioning**

Originally, Palm was focused on making smartphones running the webOS operating system. Palm produced two lines of devices: Palm Pre and Palm Pixi. At the time of introduction, Palm Pre was a high-end smartphone, whereas Palm Pixi fell into the category of mid-range smartphones.

Following acquisition by HP, the platform was repositioned to address a wider range of devices, including tablets.

Before being discontinued by HP, the HP TouchPad sold for between $480 for a 16GB model and $575 for a 32GB model (unlocked models, pre-tax)

**Ecosystem adoption**

WebOS is used solely on devices made by Palm/HP, and is not licensed to third parties.

The estimated installed base of Palm Pre and Palm Pixi devices was about 2.5M, as of mid 2010. The sales base for 2009 was estimated at 1.2M devices. Due to the acquisition by HP, sales figures in 2010 are not clear.

WebOS was designed to allow Palm to sell high-margin smartphone devices that are competitive with iPhone. Palm devices based on webOS were initially introduced with Sprint in the US (CDMA model) and later in Western Europe (UMTS models).

Palm Pre and Pixi devices were available in the US from Sprint, AT&T and Verizon Wireless, and from O2 in Germany, Italy and UK, SFR in France and Vodafone Spain. Palm devices based on webOS were initially well received by operators seeking differentiation vis-à-vis the iPhone. Disappointing sales and the fast ramp-up of Android-based smartphones caused operators to lower the priority of Palm devices. This resulted in reduced promotion budgets allocated for Palm products.

There are about 10,000 webOS apps in the Palm App Catalog, 1,000 of which have been optimized specifically for the HP Touchpad.

Initial enthusiasm among old-time Palm developers changed to disappointment with weak platform sales. Many Palm developers have now moved to iPhone and Android platforms. Compared to the leading smartphone platforms, the webOS development environment offers less functionality, and needs improvement.

Palm’s App Catalogue is the official application store available on Palm devices. Due to limited installed base and spotty global footprint, the current monetization potential of the webOS App Catalog is rather poor compared to the competition.
Technology Foundation

WebOS is built atop a Linux 2.6 kernel. It integrates a number of Linux packages, such as gstreamer for multimedia, ALSA for audio, libpurple for instant messaging, Simple DirectMedia Layer (SDL) for game programming, PulseAudio for audio management, and WebKit for HTML/CSS/JavaScript rendering.

WebOS offers a full suite of built-in applications, including email client, web browser, music player, messaging, calendar, camera, photo viewer, video player, Google Maps, memos, document viewer, tasks and contacts.

The webOS UI is designed for efficient multitasking. It introduces the metaphor of "cards" to manage multitasking. The user switches between running applications by clicking the front-face button to bring up the cards, and then flicking left and right on the screen. Applications are closed by flicking a card up and off the screen. WebOS supports multi-touch gestures, allowing most of the UI navigation to be performed using the touch-screen.

WebOS introduces a feature called Synergy that integrates feeds from multiple Internet sources into the native OS UI. Synergy supports Gmail, Yahoo!, Facebook, LinkedIn, and Microsoft Outlook (via Exchange ActiveSync).

The webOS web browser is based on the WebKit rendering engine, and is capable of displaying full Internet pages. WebOS 2 introduced support for Adobe Flash in the browser.

The WebOS application framework (Mojo) is based on web technologies (HTML/CSS/JavaScript). It introduces modified application lifecycles and an extended API providing access to platform services. WebOS applications run in a “sandboxed” environment to limit unauthorized access to data and services. The applications can interact with each other only through specific APIs and services offered by the OS. Application security is enforced at the kernel level.

The OS only installation of applications from the official application store only. The OS validates the package signature prior to installing the application.

Application development

The webOS API is essentially a JavaScript framework called Mojo. It supports common application-level functions, including UI widgets, access to built-in applications and their data and native platform services. The framework supports HTML5 features such as audio and video tagging, and database access.

Later versions of the OS introduce native application development using C/C++, and aimed primarily at game development requiring 2D and 3D graphics.

HP announced during Palm Developer Day in November 2010, that the current Mojo application framework will be replaced by the more capable Enyo framework. Enyo is expected to better support multiple form factors, screen sizes and aspect radios, and offer native hardware acceleration and improved application load times.
The webOS SDK is based on the widely used Eclipse IDE, and a webOS plugin. Its Mojo JavaScript framework allows UI development using DOM (document object model) with web technologies (HTML/CSS/JavaScript languages). Plugin Development Kit (PDK) allows access to 2D graphics API (SDL) and 3D graphics OpenGL ES 1.x and 2.x from native applications (written in C/C++).

The webOS SDK also supports the Ares IDE, which is a mobile development environment hosted entirely in a browser. Ares aims to enable a next-generation mobile development workflow, in which developers move quickly and seamlessly from editing in a browser, to debugging on a device, to selling applications in Palm’s App Catalog or on the web.

**Viewpoint**

HP has indicated at multiple occasions that it acquired Palm for its webOS software. HP is a consumer electronics company envisioning that users will be using a multitude of different device types in their daily life. In HP’s view, “successful companies control the user experience stack from end to end”. With the webOS acquisition, HP planned to deliver a “unified HP experience” on a wide range of devices, spanning smartphones, tablets and printers.

While HP’s acquisition made Palm part of large and financially solid company, it didn’t provide a solution to webOS weaknesses. In fact, there were very few real synergies between HP and Palm. Clear product differentiation, a competitive developer mindshare and operator subsidy commitment were all critical for the success of the webOS, but remained unaddressed by HP regardless of its financial assets. HP did not complement Palm in any of these critical areas, which eventually led to the platform’s demise.
Nokia: the story of Symbian, MeeGo and Qt

Over the years, Nokia has heavily invested in the development of operating system software for its smartphones, attempting to transition from its legacy Symbian OS to the more modern, Linux-based MeeGo platform and Qt application framework.

Symbian

Symbian OS is rooted in the EPOC operating system developed by Psion for its PDA products. In 1998, EPOC was spun out as a separate company with the name Symbian. The new company was co-owned by Psion and several leading mobile phone makers, including Nokia, Motorola and Ericsson.

The motivation behind the creation of Symbian was two-fold: first, to establish an alternative to possible domination of Microsoft of the mobile OS market, and second, to ensure the partner companies shared both risk and control over the new operating system.

In 2001, in order to enable independent differentiation by the Symbian handset vendor partners, the user interface framework was technically separated from the core operating system. Three main variants of the user interface were supported by Symbian for most of its lifetime: Series 60, mostly used in Nokia phones (also used in a few Siemens and Samsung models); UIQ, jointly developed by Sony Ericsson and Motorola; and MOAP (Mobile Oriented Applications Platform), used in the phones designed for NTT DoCoMo network in Japan.

With Nokia Series 60 effectively being the only successful Symbian variant outside Japan, Nokia emerged as the driving force behind Symbian. In 2008, Nokia acquired Symbian, absorbing around 1,200 employees and devoting 4,000 staff to the development of the OS.

In the hope of sharing development costs among multiple licensees, Nokia migrated the OS into an open source project called Symbian Platform. Nokia’s Series 60 user interface became the basis for the platform’s user interface.

In February 2010, the Symbian Platform was officially made fully available in open source, under EPL licensing.

Symbian Foundation released two versions of the operating system: Symbian^2 and Symbian^3, the former being a reiteration of the original Series 60 variant of the OS.

In Q3, 2010, following disappointing sales of their Symbian-based models, Samsung and Sony Ericsson announced they no longer had Symbian products in their roadmaps. The move made the structure of the Symbian Foundation redundant.

In November 2010, Nokia took over the governance of Symbian, scaling down the foundation to IP licensing. Shortly after the announcement, developers were notified of symbian.org web site being shut-down.

Nokia attempted to position Symbian as an alternative for high-end, Internet-centric smartphones, like iPhone and Android. Despite these attempts, Symbian’s market performance shows that the true comfort zone of the platform was in low-cost smartphones and high-end featurephones. Symbian
Mobile Platforms: The Clash of Ecosystems

indeed failed to compete with Apple and Google on the strength of the applications ecosystem (both in terms of apps available and apps downloaded), which has been the key driver of platform and device sales in the smartphone business.

Based on the adoption of the Windows Phone 7 platform, Nokia was convinced by Microsoft to discontinue Symbian development which has since wound down and the OS moved into maintenance mode. Nokia has since said that the OS will be used as a stop-gap solution until introduction of smartphones based on Microsoft’s Windows Phone 7 platform. In April, 2011, and in preparation for the transition, Nokia announced plans to transition thousands of employees responsible for the development of the OS to Accenture, which has taken over maintenance of the platform.

MeeGo

MeeGo started its life as an open source Linux project announced by Intel and Nokia in February 2010, at Mobile World Congress. MeeGo combined Intel’s Moblin project (focused on netbooks) and Nokia’s Maemo project (focused on handhelds) into a single open source software platform.

The operating system was designed to support a number of standardized open source UX (User eXperience) profiles. The initial UX profiles supported by the platform are netbook, smartphone, IVI (In-Vehicle Interface) and media center/TV.

In October 2010, MeeGo announced the MeeGo 1.1 release. It includes Core OS 1.1, together with UX stacks for netbooks, in-vehicle infotainment and handsets.

In November 2010, MeeGo released MeeGo SDK 1.1 Beta, based on Qt Creator and Qt Quick development tools.

MeeGo was driven by two large players with conflicting agendas. Nokia saw MeeGo as the application platform for its high-end smartphone products. Nokia’s goal was to share the burden of software development with industry partners. Intel’s goal was to increase penetration of its Atom processors into netbook, smartphone and other device types.

In June, 2011, Nokia launched the one and the only MeeGo handset: the N9. The company stated that there will be no more smartphone models running the OS. MeeGo’s fate was sealed after Nokia withdrew its support following the partnership announcement with Microsoft.

Intel had to search for new industry partners, and in September 2011, announced a partnership with Samsung for the development of a new software platform called Tizen. Tizen will be based on both MeeGo and SLP (Samsung Linux Platform), selectively merging components from both and basing its application framework on HTML5.

Tizen seems set to be another victim of misaligned incentives across several industry partners. Samsung is bringing SLP to the “standards” table simply to find a new home for it, now that LiMo is winding down. Intel is seeking another marriage of convenience, trying to tempt a major OEM to support its x86 processors.
Qt

Qt is a cross-platform application and UI framework used in many embedded devices, ranging from coffee machines to medical equipment. Qt has been used in a diverse set of commercial applications since 1995, including products of such companies and organizations as Adobe, Boeing, Google, IBM, Motorola, NASA, Skype, AutoDesk, KDE project and DreamWorks.

The Qt framework was originally developed by Norwegian company Trolltech, which was acquired by Nokia in the beginning of 2008. Following the acquisition, Nokia has held all rights to the platform, which is distributed under a dual commercial and open-source licensing scheme.

Nokia has extended the framework by introducing Qt Mobility APIs that expose mobile platform capabilities to Qt applications through a set of standardized Qt APIs.

In addition to C++, Qt supports a hybrid development model where applications are developed using a combination of Web technologies (HTML, CSS, JavaScript) and C++. For that purpose, Qt integrates the WebKit web engine, responsible for rendering of HTML/CSS and running JavaScript code.

The Qt version 4.7 introduces the Qt Quick framework, which allows developers to describe user interface layouts by means of a new declarative scripting language called QML. QML is a declarative language designed to describe the user interface of a program: both what it looks like, and how it behaves. In QML, a user interface is specified as a tree of objects with properties. QML can be used seamlessly with JavaScript as glue code for controlling the overall behaviour of user interface transitions.

Initially, Nokia planned to use Qt as a primary application framework on all its smartphone products. Qt was meant to provide a unified application framework on top of its two separate software platforms – legacy Symbian and the more modern MeeGo.

Symbian and MeeGo are being discontinued. However, Nokia is rumoured to be featuring Qt within an upcoming operating system code-named Meltemi. An evolution of S40, the new platform is based on Linux, and will target feature phones.

Given the very large volumes behind S40 and the sub $100 price range, Nokia’s Qt may indeed allow Nokia to create a platform that is competitive to both Android and iOS, but addressing the feature phone segment of the market that neither platform can target. The challenge remains whether Nokia can integrate Qt to its phone hardware to optimise performance in low-spec feature phones and whether the Qt app ecosystem can effectively attract developer mindshare away from the mainstream smartphone platforms.

Lessons from Nokia’s platform strategy

Nokia’s wide product portfolio pushed the company to develop multiple incompatible operating systems, which proved to be disastrous. The proprietary S40 OS was aimed at low-cost featurephones, Symbian at high-end featurephones and mid-range smartphones, and MeeGo was targeting high-end smartphones.
In contrast Apple uses two variants of the same operating system that share a common architecture, OS core, and large portions of their application frameworks and development tools. iOS is used for mobile and portable devices with touch-based UIs, while Mac OS is used for personal computers with UIs based on keyboards and pointing devices.

While Nokia was focused on harmonizing its multiple software platforms, both iOS and Android were building successful application platforms driven by strong network effects. These network effects proved to be stronger than Nokia’s distribution power and supply chain efficiency, forcing Nokia to abandon its own software platforms in favour of Windows Phone 7.

The history of Nokia’s software efforts presents two learnings. First, that building a successful application platform requires a strong “software DNA”. Both present-day leaders (Apple and Google) as well as the credible challenger (Microsoft) are all companies with considerable experience in building application and Internet platforms. Second, that network effects (economies of demand) are far stronger than supply chain efficiencies (economies of supply), which has allowed disruptors (Apple/Google) to challenge and win over Nokia’s internal smartphone platforms.