A Rapidsoft Systems' White Paper

© September, 2008

Mobile Games Development – Options, Market Directions and Technologies

Mobile phones' primary purpose was to make phone calls, but that role has now transformed into a personal tool that combines a lot of other functionalities that mobile people need. Recently, mobile phones have emerged as personal entertainment device specially in youth and teenagers. Today's mobile Phones can play music, browse Internet, record and watch videos, and of-course play games.

Among the many activities possible on phones, mobile games have been the first to get wide acceptance as they cater to much wilder segment of populations. The tremendous popularity of mobile gaming is playing a pivotal role in revenue generation for the cellular carriers, game publishers, and handset makers, while generating new opportunities for game developers and game professionals. With the number of mobile gamers around the world expected to reach 250 million by 2009, the mobile gaming business is definitely on upswing and will constitute even a bigger portion of the profit pie for the cellular carriers and handset makers. This is an opportunity that Rapidsoft team can help you, if you are a publisher of a game or a developing house.

Categorizing Games

Just like console games, mobile games span several major categories.

Every imaginable game genre is covered on mobile, and most of them have good game play and good depth. Here is a breakdown for the most popular ones (source: IGD White paper):

- **Sports:** Bowling, Golf, Football, Basketball, Hockey, Soccer, etc. Big sports brands are well represented covering all the major sports categories.
- **Racing:** All types of tracks (off road, drag, circuit) and all types of vehicles are represented. Understandably this is a very popular category as little instruction is necessary for a good racing game and it attracts consumers easily.
- **Action**: Numerous brands are represented in this category, dominated by side-scrolling platform games that are easy to play on mass-market devices. There have been several attempts at twitch fighting games which have met with some success. It is very difficult to generate the fast action needed from the current handsets for real fun game play.

- **Adventure:** Most of the games in this category have been adapted from PC adventure games and can be quite complex. Thus far the adventure category has been only a moderately successful category in the US.
- **Word/Trivia:** A very popular category which works well across all devices as it is not dependent upon phone engine speeds and screen limitations to deliver fun game play.
- **Arcade/Classics**: Always a favorite, this category does well. Populated by the long term favorites
- like Ms. PacMan, Galaxian, and Asteroids.
- **Logic/Puzzles:** Some of the highest revenue generating mobile games are in this category. Tetris type (falling block) games are perpetually at the top of the charts and are being joined by a great selection of titles found on casual game portals such as Pogo.com, Yahoo! Games, MSN Zone, RealArcade, Shockwave.com, etc.
- **Strategy/Simulation:** This category has been slow to take off in the US, but as handset capabilities continue to improve, active gamers are taking the mobile medium more seriously. Opportunities exist to create a more meaningful link between mobile and PC versions their favorite strategy game.
- **Casino:** Card games traditionally sell well on line and mobile follows suit. These games provide opportunity to establish a longer-term relationship with the game through subscriptions. This genre is very crowded however, and getting space on operator decks requires a unique concept to be considered.
- **Parlor:** Much like casino games, parlor games is top sellers, but this is an extremely competitive category.

Unlike console games, mobile games scope are much limited due to limitation of handsets. Also, mobile games have only a few levels unlike to PC or console games that can support complex scenes and many levels of increasing complexity. That makes developing mobile games a far more manageable compared to their desktop or console counter parts.

Mobile games can be classified into three broad categories:

- **Embedded games:** Games that are hardcoded into the mobile handset's system and shipped with it. Nokia and other manufacturers like Motorola, shipped many simple embedded games on its symbian phones. These were simple rather games with low level of challenge or entertainment value.
- **SMS games:** Some vendors and carriers tried to create SMS based games. Games are played by sending text messages—for example, SMS to game server—that process them and sends back the result through SMS. Often in the form of live contests and polls. Not very popular because the cost of gaming increases with each SMS sent to the game server.
- Browser Based or Downloadable games: These games are played using mobile phone's built-in **microbrowser** (net browser for mobile devices), either in online or offline mode. Players can play such games online through their cellular carrier's or a third-party game provider's game Web site, or download them for offline gaming. This category includes a wide range of games, such as solo or multiplayer games, network games, offline games, arcade games, and so forth

Among these three categories, downloadable games are today's most popular type of mobile games for their innovative and multimedia-rich content, appealing presentation, and lower cost of gaming compared to SMS games. This white paper discusses the development of downloadable games and henceforth, the term "mobile games" will refer to "browser games" in this article.

In this article, we concentrates on 2D games. Because a large number of mobile phones in circulation today have very limited resources (small screen, limited memory and graphics support, cumbersome key input), the most suitable as well as commercially feasible games for these devices are 2D games. But, as capabilities of mobile phones are bound to increase over time, 3D games will become commonplace in the near future.

Just like all other Mobile applications, there are many options for game developers. These are:

- C/C++ For Brew Phones
- J2ME (Java 2 Micro Edition)
- Flashlite from Adobe

J2ME Java for Mobile Game Development

Those folks that have developed games for Desktop or console environment the use of C/C++ is natural for games development. Unfortunately, the paradigm does not that easily works on the mobile phones. Although C++ has the advantage of being compiled into native code and provide direct access to various system resources, it may not be the best option for mobile environment. BREW platform, a primary environment for C/C++ based development, provides end-to-end solutions to mobile game developers while allowing them to work with any desired language (including C++, Java, XML, and Flash), Java is the most popular choice for game development. J2ME has emerged as the primary language for game development as it combines a number of powerful features for mobile phones, ease of portability and free powerful development environment. , is identified as the most convenient for developing mobile games. The main reasons behind J2ME's popularity for mobile games and application development are:

- J2ME enjoys the status of an industry standard backed by all major handset makers, with most of the present day mobile phones being Java-enabled.
- J2ME is a free and open platform. This helps keep the development costs low and provides the necessary flexibility with ample support freely available for developers using it.
- Its highly portable nature ("Write once run anywhere") ensures that a game application written for one brand/type of handset will work with all other brands/types of Java-enabled handsets.
- It is especially optimized for small devices, is lightweight, and is highly secure because applications written on it cannot access or affect other applications running on the phone/device.
- J2ME consists of the **Mobile Information Device Profile (MIDP) API** that is designed specifically for developing applications for mobile devices including mobile phones, keeping in mind their limitations and constraints. Furthermore, the latest **MIDP version 2.0** itself dedicates a whole API to game development, making game development simpler and quicker.

MIDP 2.0 in Game Development

MIDP 2.0 API is a set of feature-loaded APIs used for developing secure, rich-content multimedia applications, including games, for mobile devices. MIDP 2.0 builds upon its predecessor MIDP 1.0 to provide a better development platform for building efficient and fast mobile applications.

MIDP 2.0 further refines the features and functionalities provided by MIDP 1.0. One of the important additions made to MIDP is the **Game API**, or the javax.microedition.lcdui.game API package to be precise. Through the Game API, MIDP 2.0 provides game developers with the readymade building blocks that were to be developed from scratch in the case of MIDP 1.0. These building blocks are classes for creating and controlling various game elements such as game canvas, sprites, layers, and so forth (these are explained in the next section). Thus, MIDP 2.0 significantly reduces the time involved in game development.

The other two MIDP 2.0 API packages essential for game development, also explored by this article, are javax.microedition.midlet and javax.microedition.lcdui.

The **javax.microedition.midlet** API package provides a base for developing all mobile applications. It contains the **javax.microedition.midlet.MIDlet** class, which is the base class of all J2ME-based mobile applications (also known as **midlets**) and must be extended by the main classes of all mobile applications. Quite similar to the **java.applet.Applet** class, the **MIDlet** class provides resources necessary to create midlets.

The **javax.microedition.lcdui** API package is necessary to develop a user interface (UI) for all types of mobile applications. This API provides classes to create and control UI components (such as screen, form, text box, radio buttons, and so on) and processing input for mobile applications, including games. Developers who have GUI development experience with **AWT** and **SWING** will find that the elements of the javax.microedition.lcdui package are similar to elements from these APIs.

It is easy to build games in the emulator and test them. The figure below shows a sample game that is running in the emulator.

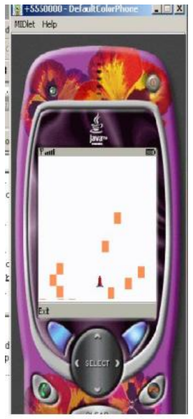

Figure 1: Emulator running the example game app during development

Market Directions

This section is largely based on the data published and provided by IGDA.

1. Traditional video game publishers

Over the past few years, video game publishers such as THQ, Electronic Arts, Activision, Disney have begun to spend on mobile gaming as a way of increasing revenue. This in addition to the many smaller pure-play wireless companies that are developing mobile games and content, which provides an indication that revenues are increasing. All this activity has led to a significant expansion in the number of games available and the quality of the games.

2. Increased availability of games

From a small number of mobile games available a few years ago, today there are hundreds. In the US, most of the major carriers offer a variety of games in many different categories. In some cases, the wireless games pattern themselves after the console titles, such as *Tony Hawk's Underground* and *Tiger Woods PGA Tour 2008,* and the "classics" such as *Galaga* or *Pitfall.* There are also unique movie games and adaptations of parlor and card games to the mobile space like *Texas Hold'em, Pub Pool* and *Darts*. There are even unique attempts at multiplayer and community based products.

3. Enhanced quality of games

In addition to the availability of games, one of the biggest changes in the last year has been the quality of the games. This has been driven by increased competition, mobile devices with greater capability (better graphics, more memory, etc.), and the growing revenue streams from mobile gaming sales. Continuing advancements in device capability and growing revenues should help drive even better quality games going forward. Major developments on the horizon include increasing numbers of 3D rendered and multiplayer games. In the early stages of the mobile game business the products were little more than novelties with very simple graphics and limited sound capabilities. As quality and capabilities expand, consumers will be much more likely to view mobile games much more as entertainment products in their own right and propel mobile gaming into the mainstream.

4. Improved network capability

Carriers are continuing to pour money into upgrading their networks to allow high-speed data. As this transition is occurring and more mobile users have access to data enabled handset with higher download speeds and more on board memory, the quality of games should continue to improve. It will be easier and faster to download larger, higher quality games that have more content. While mobile devices may not be as powerful as internet-connected desktop computers, the advantage they have is that **mobile content has never been free** – the for-pay model that has been implicit in the mobile market is a healthier, more powerful, and faster driver of growth than the historically free desktop internet.

5. Solid growth projection

While mobile phones do not offer the deep, rich gaming experience a dedicated console does, they do offer a ubiquitous, ever-present, connected platform that is growing faster than any other medium.

Mobile Games Develeopment Value Chain

The mobile value chain is similar to the traditional video game business where developers, console manufacturers, distributors, publishers, retailers, and consumers are all engaged in the purchase of game products. However, even though there are similar players, they are engaged in a very different business.

1. Game Developer

Relative to the traditional video game business, the developer’s role in the mobile game business is the least different of all the members of the value chain. They remain the creators and producers of the initial game concept to the final playable and released gold master code. Their mission remains to create fun games. As you look closer, it becomes clear that the nature of the challenge is different. Mobile game developers must not only create and develop great games but they must make sure those games run on a wide variety of mobile phones.

The obvious impact is that games must support whatever the native API, graphic format or audio format is available. In addition, screen sizes and processor power variances cause additional design challenges.

What worked really well on an LG 8600 in 3D must be completely re-designed to work on a Nokia N95 . It’s easy to argue that these are two totally different devices and potentially, two totally different games. Finally, multiply all of this complexity by two (or more) because there are two dominant software platforms that must be supported to please the various carriers: Qualcomm’s BREW (“Binary Runtime Environment for Wireless) and Sun’s J2ME (“Java 2 Mobile Edition”).

The end result is that developers normally deliver several different baseline “reference builds” that are used to port to ever increasing number of handsets on the market.

For developers there is an unfortunate reality to the mobile marketplace today, as you will see by continuing to read this section: All of the real power and opportunity is in the hands of the carriers, publishers and license

holders. For most developers, the only opportunity to get involved is in work-for hire development, which is becoming commodity and is increasingly being outsourced overseas. Where
some smaller developers are finding more upside is in establishing intellectual property in other channels,
namely with downloadable games, and then licensing newly established properties into mobile, becoming
licensors instead of or in addition to being developers.

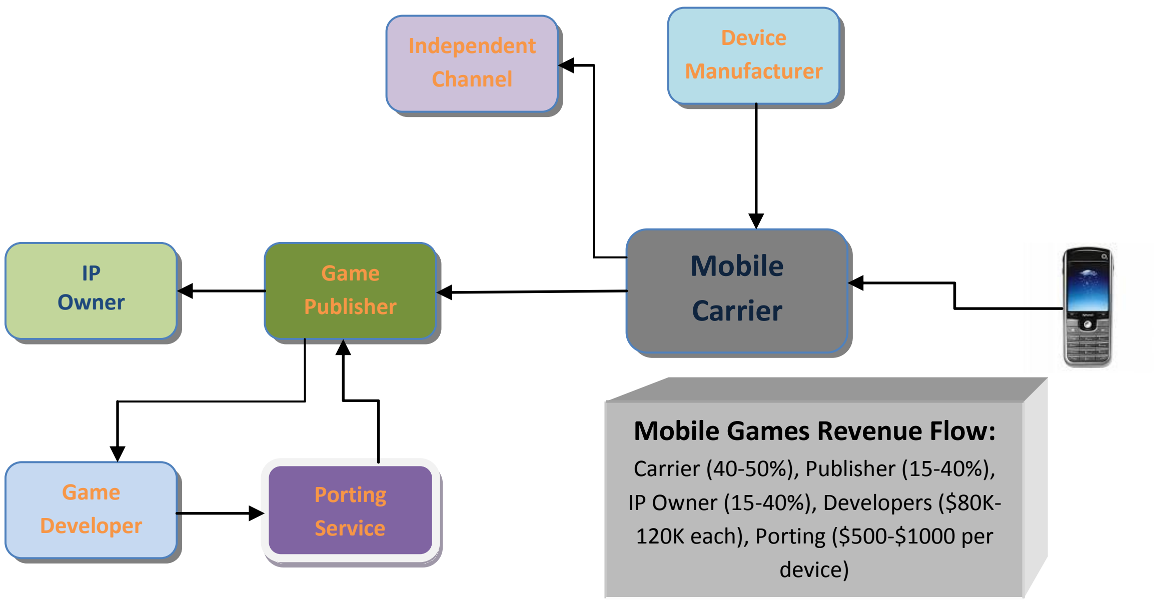
Independent Channel
Device Manufacturer
IP Owner
Game Publisher
Mobile Carrier
Game Developer
Porting Service
Mobile Games Revenue Flow:
Carrier (40-50%), Publisher (15-40%), IP Owner (15-40%), Developers ($80K-120K each), Porting ($500-$1000 per device)

Figure 2 : Showing Role of Various Entities and Revenue Flow

3. Game Publisher/Aggregator

The mobile publishing business is more a mirror of the traditional video game business. Publishers plan a slate of titles based on IP they either own, create or plan to acquire and then match that IP to in-house or 3rd party talent to create a game. There are over 100 devices across the major carriers in the US market alone. Publishers want their game on as many of these devices as possible in order to realize the full sales potential of any given title. This is not only driven by the publisher desire to support many platforms – carriers will favor games that support the widest selection of handsets, even though the newer models will have a very low installed base (and thus might not otherwise be targeted by publishers).

The basic business model of the mobile game publisher is the same as in the PC and console video game industry. They acquire the rights to different IP and typically fund development of the game through internal or external studios to bring it to market. They also typically fund the porting process and manage the delivery of the game to the carriers.

4. Handset Manufacturer

The handset manufacturers are the mobile industry equivalent of game console manufacturers. As consoles, these handset manufacturers embed different run-time environments into their handsets which include virtual machine or byte code environments like Java.

BREW, Symbian and Pocket PC OSs provide binary runtime environments similar to Windows where applications are usually in a binary form. Further fragmenting the market, or at least complicating the porting process, is the addition of proprietary API's that may be added by the handset makers themselves or their operator customers.

Handset manufacturers play an important part in setting market direction of the technology that enables games and, in some markets can even play the role of distribution partner to publishers and studios alike in the form of pre-loaded demos or by purchasing games that show off the latest and greatest technology. The leaders, in terms of global market share are Nokia, Motorola, Sony Ericsson, Siemens, Panasonic and Samsung.

5. Carriers

Carriers wield significantly more power in the mobile gaming space than retailers do in the traditional game business because they have a monopoly over their very large customer base – and game selection is simply not a criterion for most people in choosing their carrier.

Carriers don't only provide the storefront but also drive pricing, technology specifications (e.g., 3D API's, digital rights management, community infrastructure), determine/enable various business models (e.g., subscription, one-time download, micro-payments), and they provide the network that connects it all together. No other player in the value-chain touches so many aspects of the user experience. Without strong carrier partnerships a publisher and its products simply will not make it.

The reason the carriers wield so much power is that today there is no meaningful alternative method of distribution to "the deck", which is the small, text-based shopping interface completely programmed by the carrier. But for game developer seeking to bring their art to mobile phones, it is important to realize that carriers are the gate-keepers to customers, and the established publishers are gate-keepers to the carriers.

6. Independent Channels

This category refers to all web, WAP ("wireless application protocol") and SMS ("short message service") sales channels not owned by the carriers. This includes portals run by device manufacturers and publishers, mobile content offerings on major fixed line portals.

Device manufacturers have to make sure that content is available for all their handset models to make them attractive to end-users, and ensure content is available even when a model has just a small market, or limited penetration such as at launch.

Most device manufacturers license content which is embedded into handsets as well. Device vendors also support devices through the web, and through this medium also supply a range of content to ensure subscribers have a choice. Devices sold through non-operator channels are often set-up to drive users to the device maker's portals for content. Examples include My-Siemens, Club Nokia and Hellomoto.

It is important to understand that the value of embracing these channels is very limited, as they have very limited traffic in comparison to the official carrier channels. The European market is far more open to alternative distribution (i.e., non-carrier) than the U.S. market. It is safe to say that everyone in the value chain other than the carriers eagerly awaits the day when non-carrier channels become viable for mobile content.

Porting Services

Many developers aren't prepared for the job of creating individual game builds for the hundreds of devices in the marketplace. Just acquiring the handsets themselves is a challenge, not to mention the large commitment to testing and QA required. Then you need to consider localization normally into the five primary EFIGS languages (English, French, Italian, German, and Spanish). This makes the number of SKU's potentially staggering. Depending on the complexity of the game, the porting process usually exceeds the initial development costs. There are a few companies which have technology that helps to ease this process through automated tools. This is likely to become the trend due to the high cost of porting without a process that addresses handset diversity.

Wireless developers and publishers spend an enormous amount of time, energy and money creating top notch games. But these costs can never be recouped unless the games are available on multiple phones across multiple carriers. Game purchases are directly proportional to the number of handsets and carriers that are supported.

In addition to the obvious benefit of providing more potential customers, there are also many other advantages of supporting multiple devices. A more prominent market presence provides greater game visibility, consumer recognition, and opportunities for word of mouth recommendations. Carriers are also more inclined to promote and market games that support more of their phones. Finally, the game is much more likely to be placed in a 'top ten' category if it is available on many devices. It's clear to see why porting to multiple handsets therefore provides a competitive advantage.

Since it is so important, why isn't every application already ported to every available handset? The reason is porting can be time consuming, expensive, and requires access to all the handsets as well as intimate knowledge of their strengths and weaknesses.

These porting challenges can be minimized by developing a porting strategy and producing code that is port-friendly.

Developing a Porting Strategy

A porting strategy is an important part of the overall plan to bring product idea to profitability. Product success will be very difficult if the product plan only focuses on developing to a few baseline handsets. A common mistake is to spend most of the development schedule and budget on targeting the few high end handsets and then hoping that the more common lower-end handsets can easily be supported. The fact is: they cannot.

Before writing a single line of code, a company should first determine:

- distribution channel (carrier deck, third party portals, direct download, etc)
- the target carriers (it's a good idea to first ensure they will accept the title)
- target handsets (be explicit and identify each model, not just manufacturer families)
- timeline and budget for reference builds
- timeline and budget for ports
- what resources will produce the reference builds and ports

This last item is the one most often overlooked. The best resources for developing the baseline builds are not necessarily the best resources for producing the ports. Creating the baseline build can be exciting for developers due to the raw creativity involved, but doing the porting and re-sizing can be a tedious, technical project that bores a creative design team. Porting that initial build to over fifty handsets requires a different kind of creativity in which the programmer must work out how to fit the original features into phones with different memory, speed, and graphical capabilities. The two development efforts are as different as a sports car and a sport utility vehicle. Don't assume that the same team can do both well.

There is also considerable cost to purchase and maintain live network coverage for all of the targeted handsets. It’s one thing to code a game, another to build and host a server side environment with 99.9% uptime to handle server requests from your mobile game. More importantly, there is a steep cost associated with learning the idiosyncrasies of each device and how to achieve the best performance for each.

For these reasons, developers may wish to consider using a professional porting company to produce the ports. If developers consider using a porting partner, don't jump to a hasty decision. Ask the companies for their list of actively supported carriers and handsets, ensure that the work is performed at a location where the target carrier network coverage is available, and ask about their QA procedures and for references. It may even be wise to tour their facilities and meet the staff who will be performing the porting work.

Regardless of who performs the porting, the time line and costs will depend on how the original code was structured. The product will come to market faster if the reference code is “port-friendly”.

a) Planning Port-friendly Code

At the risk of interfering with the lead developer's creativity, the following are some tips to facilitate the porting.

- Don't write the baseline only to high-end phones. Make at least reference builds for low, medium and high capability devices.
- Avoid or encapsulate use of device-specific graphical APIs.
- Encapsulate sound functions as they will likely be replaced for different devices
- Don't overuse the Object-Oriented paradigm. (Enterprise developers moving down to small
- device programming will have issues with this.)
- Use threads carefully and do not overuse them.
- Don't hard-code key input values into the code; use constants.

- Don't hard-code graphical positions; place graphics dynamically based on screen size.
- Use a standard build process.

b) Ensuring Success: Port Testing

The importance of testing cannot be overstated. A porting company must have well-defined, reproducible testing procedures in place that follow specific carrier test plans. Even so, the application producers must be involved in the testing and overall quality assurance process for the ports. Producers should review ports to ensure that they meet the guidelines of the application. It may be necessary to make certain judgment calls during the porting process in order to support certain devices, e.g., to accommodate a smaller screen, the porting programmer might split a copyright notice onto two screens.

Producers should review these changes in order to ensure that contractual and aesthetic guidelines are being met. Finally, some aspects of testing are necessarily subjective in nature. One tester may feel that sound volumes are too loud whereas another might think they are too soft. And some might think that three key presses to activate a special game action are too many, whereas others might think it is just right. The producer should ultimately make judgment calls about these subjective issues and be prepared to defend them.

Conclusions

Mobile game development has become a lucrative industry with a manifold increase in mobile subscriptions and the number of avid mobile gamers around the world. However, success depends on reducing the cost of development and porting and timely execution of the development and application porting.

Use of external porting can cut down the cost and time to market applications. While it is possible to create an internal porting capabilities for most companies that option will be quite expensive since it requires significant man power efforts that are cyclical in nature. This means using a trusted partner like Rapidsoft Systems to develop and port your applications can be the best solution for you.

For more information and specific questions, please contact us at:

Rapidsoft Systems, Inc,
Mailing Address: 7 Diamond Court, Princeton Junction,
New Jersey 08550, USA

www.rapidsoftsystems.com

Phones: **1-609-439 9060 (US East Coast, NJ Office)**
1-408-829-6284 (US West Coast, San Jose Office)
Toll Free (North America): 1-800-946-5490
Fax: 1-831-855-9743

Email: info@rapidsoftsystems.com